REINFORCING LARGE VISION LANGUAGE MODELS FOR MULTI-HOP REASONING IN LIFELONG KNOWLEDGE EDITING

Anonymous authors
Paper under double-blind review

ABSTRACT

Lifelong multimodal knowledge editing aims to correct outdated or incorrect knowledge in large vision language models (LVLMs) without retraining. Recent studies have significantly improved editing quality in terms of locality, generality, and reliability. However, the ability of post-edit models to perform multi-hop question answering over sequentially injected facts remains critically overlooked. This challenge stems from the confinement of existing supervision signals to the edited fact itself, with no dedicated mechanism for coherent multi-hop reasoning chains. To bridge this gap, we propose **MoE-GRPO**, a reinforcement learning framework built on GRPO for lifelong multimodal knowledge editing over MoE-based editors. Using two-stage singular value perturbation via full SVD decomposition and carefully designed outcome-level rewards, MoE-GRPO provides direct supervision over multi-hop reasoning, bypassing the need to explicitly model intermediate parameter disruption. Extensive experiments on BLIP2-OPT, LLaVA-v1.5, and MiniGPT4 demonstrate that MoE-GRPO achieves substantial relative gains in multi-hop reasoning, while accepting a small, deliberate trade-off on single-hop editing metrics (Reliability, Locality, Generality) that is intrinsic to the multi-hop reasoning objective. Ablation studies reveal practical trade-offs between editing quality and reasoning ability induced by key design choices. Furthermore, MoE-GRPO generalizes consistently across different MoE-based editors, yielding robust reasoning improvements.

1 INTRODUCTION

Benefitting from the prosperity of large language models (LLMs) (Touvron et al., 2023a; Achiam et al., 2023; Guo et al., 2025; Touvron et al., 2023b), large vision language models (LVLMs) (Dai et al., 2023; Lu et al., 2023; Liu et al., 2024; Bai et al., 2025) have attracted growing attention for their wide applications. Meanwhile, concerns have emerged regarding outdated facts, inherent biases, or offensive content arising from training data contamination. As real-world knowledge is inherently grounded in both language and vision with complex relationships, lifelong multimodal knowledge editing (MKE) (Chen et al., 2025a;b; Jiang et al., 2025b; Han et al., 2026) provides a principled alternative to costly retraining by directly modifying or injecting specific facts into pre-trained models.

Recent advances in knowledge editing have coalesced around two canonical paradigms: locate-then-edit (Meng et al., 2022a; Mitchell et al., 2021; Fang et al., 2024; Pan et al., 2025; Chen et al., 2025b) and extra-parameter methods (Mitchell et al., 2022; Huang et al., 2023; Dong et al., 2022; Wang & Li, 2024b; Zhang et al.), among which MoE-based editors (Chen et al., 2025a; Wang & Li, 2024a;b) have emerged as one of the strongest paradigms for lifelong MKE owing to their modular architecture. Despite these achievements, as shown in Figure 1, existing knowledge-editing approaches (Han et al., 2026; Chen et al., 2025b) still struggle with multi-hop question answering, also called portability (Huang et al., 2024). While approaches specifically designed for multi-hop reasoning (Gu et al., 2024; Wu et al., 2025; Lu et al., 2025; Shi et al., 2026) of post-edit models try to tackle these challenges, they have been largely confined to pure LLMs, leaving multimodal knowledge reasoning unexplored.

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.

We attribute those challenges in the post-edit model multi-hop reasoning to a fundamental supervision gap in existing editing paradigms. The supervision signal in current methods, including locate-then-edit and adding parameters paradigms, is confined to the edit fact itself, with no coverage of the downstream effects on knowledge interdependencies. While such side effects are unavoidable and necessary, the mechanisms of how parameter shifts propagate to the existing knowledge graph, disrupting the multi-hop reasoning chains, remain opaque in lifelong knowledge editing. However, the disruption can be observed phenomenologically through degraded performance on tasks requiring multi-hop inference over edited facts. Reinforcement learning naturally offers a principled alternative: by optimizing outcome-level rewards derived directly from such observable phenomena, it provides supervision without requiring explicit modeling of the underlying disruption mechanisms.

A natural alternative to parametric editing is in-context memory augmentation, where each edit is stored in an external buffer and retrieved at inference time. However, parametric editing offers three principled advantages in the lifelong multimodal setting. **(i) Capacity:** in-context memory is bounded by the context window, which is quickly saturated under 1000+ sequential edits, whereas parametric experts scale linearly with edit count. **(ii) Integration:** parametric edits are fused into the model's representation space, enabling implicit reasoning over edited facts that cannot be achieved by retrieve-and-concat operations. **(iii) Inference-time stability:** parametric editing preserves the original inference pipeline, avoiding the latency and error propagation of retrieval-augmented decoding. These advantages motivate our focus on parametric editing, consistent with prior lifelong knowledge editing literature (Chen et al., 2025a; Wang & Li, 2024a; Chen et al., 2024; Wang & Li, 2024b).

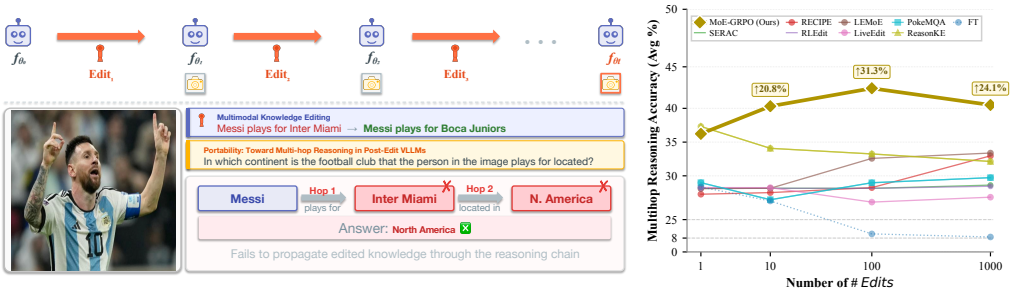


Figure 1: Lifelong multimodal knowledge editing in LVLMs. As a multimodal knowledge pair is injected into the LVLM, it should be capable of leveraging this fact to perform inference across related reasoning chains, while the unrelated knowledge should be preserved. The left shows this multi-hop reasoning process, while the right shows the current KE method performance on the multi-hop question answering task.

Motivated by these desiderata, we propose **MoE-GRPO**, a reinforcement learning framework based on GRPO (Shao et al., 2024) for MoE-based lifelong multimodal knowledge editing towards multi-hop question answering. It bridges the critical *supervision gap* between isolated editing operations and coherent multi-hop reasoning over sequentially edited multimodal facts. During the process of generating experts, the feature extractor provides refined embeddings and information about the editing sample to the expert generator, and the routing mechanism is determined by the output of the feature extractor. We treat the joint distribution of these two as the policy and formalize lifelong multimodal knowledge editing as a Markov decision process (MDP). To encourage exploration during RL, we introduce SVD-based structured perturbation into GRPO, with noise injected into the singular values of expert parameter matrices. We deliberately adopt an inverted lower-bound clipping to preserve large positive advantages to compensate for the gradient signals that would be suppressed by the frozen LVLM backbone under MKE settings. Furthermore, we update old policy networks with an *asymmetric normalized gradient*: the feature extractor receives a regularized gradient to maintain stable routing, while the gradient of the expert generator is less constrained to aggressively apply newly learned facts.

Extensive experiments show that MoE-GRPO achieves strong multi-hop portability, which sustains even after 100 sequential edits. MoE-GRPO achieves a relative portability improvement of 30.48% and 24.23% over the second-best baseline at 100 and 1000 edits on BLIP2-OPT, respectively, while maintaining near-comparable locality, generality, and reliability (within a few percentage points of the baseline editor) during the lifelong editing procedure. It also reveals a potential trade-off between

portability and other metrics. Ablation studies empirically demonstrate the asymmetric gradient normalization: the feature extractor demands the reward-driven regularization of gradients, while the expert generator needs unrestricted gradients. Diving down into the different settings of callback depth in sequence consistency and perturbation noise level helps us further balance multi-hop reasoning and the editing performance. We further visualize the reinforced experts at the inference stage and apply attention analysis to show that MoE-GRPO reinforces experts to learn a new marginal distribution associated with new knowledge restoration.

2 PRELIMINARIES

2.1 LIFELONG MKE PROBLEM DEFINITION

We adopt the parametric editing formulation: an editor computes parameter updates $\hat{\mathcal{E}}_t$ that are fused into the LVLM via residual injection at the edit layer, distinct from in-context memory approaches that store facts in an external buffer. Given a pretrained LVLM $f_{\mathcal{W}} : \mathbb{X} \mapsto \mathbb{Y}$, let $\mathcal{D}_{\text{edit}} = \{(\boldsymbol{I}_e, \boldsymbol{x}_e, \boldsymbol{y}_e) \mid f_{\mathcal{W}}(\boldsymbol{I}_e, \boldsymbol{x}_e) \neq \boldsymbol{y}_e\}$ denote the set of multimodal knowledge samples. The editing objective is to derive a post-edit model $f_{\mathcal{W}'}$ from $f_{\mathcal{W}}$ such that

$$f_{\mathcal{W}'}(\boldsymbol{I}_e, \boldsymbol{x}_e) = \begin{cases} \boldsymbol{y}_e, & \text{if } (\boldsymbol{I}_e, \boldsymbol{x}_e, \boldsymbol{y}_e) \in \mathcal{D}_{\text{edit}}, \\ f_{\mathcal{W}}(\boldsymbol{I}_e, \boldsymbol{x}_e), & \text{otherwise.} \end{cases} \tag{1}$$

When $\boldsymbol{I}_e = \varnothing$, the multimodal knowledge editing problem reduces to pure LLM editing. As knowledge arrives in a sequence $\{(\boldsymbol{I}_t, \boldsymbol{x}_t, \boldsymbol{y}_t)\}_{t=1}^{T}$, the lifelong editing can be represented as:

$$f_{\mathcal{W}_t} = \mathrm{ME}(f_{\mathcal{W}_{t-1}}, \boldsymbol{I}_t, \boldsymbol{x}_t, \boldsymbol{y}_t), \quad t = 1, 2, \ldots, T, \tag{2}$$

where $\mathrm{ME}(\cdot)$ denotes the editor updating calculated from the t-th target knowledge $(\boldsymbol{I}_t, \boldsymbol{x}_t, \boldsymbol{y}_t)$.

2.2 MIXTURE-OF-EXPERTS BASED MKE

MoE-based knowledge editing paradigm (Chen et al., 2025a; Wang & Li, 2024a;b) maintains an expert pool, dynamically growing via $\mathcal{E}_{t+1} = \mathcal{E}_t \cup \{\boldsymbol{e}_k\}$. Each expert $\boldsymbol{e}_k$ is rank-r parameterized and can be decomposed into two low-rank matrices $\mathbf{C}_k, \mathbf{R}_k \in \mathbb{R}^{r \times d_m}$, which together produce a residual correction $\Delta \boldsymbol{h}_k = \rho(\boldsymbol{h}^{l_e} \mathbf{C}_k^{\top}) \mathbf{R}_k$ to the intermediate representation at the edit layer, where $\rho(\cdot)$ is an activation function. The MoE-based editor consists of an *expert generator* $\mathcal{G}_\psi$ and a *feature extractor* g_ϕ, where the *feature extractor* generates the corresponding gating probability under a deterministic routing mechanism, while the *expert generator* leverages $\boldsymbol{h}^{l_e}$ and the feature extracted by g_ϕ to generate $\boldsymbol{e}_k$. A new expert and its gating probability distribution are jointly produced:

$$\begin{cases} \boldsymbol{e}_t \triangleq \mathcal{G}_\psi\Big(\boldsymbol{h}^{l_e}, g_\phi\big(\mathrm{Emb}(\boldsymbol{I}_t^e, \boldsymbol{x}_t^e, \boldsymbol{y}_t^e)\big)\Big), \\ \boldsymbol{p}_t = \begin{bmatrix} p_1 & \cdots & p_t \end{bmatrix} \triangleq \mathit{Route}\Big(\mathcal{E}_t, \boldsymbol{h}^{l_e}, g_\phi\big(\mathrm{Emb}(\boldsymbol{I}_t^e, \boldsymbol{x}_t^e, \boldsymbol{y}_t^e)\big)\Big) \end{cases} \tag{3}$$

where $\mathrm{Emb}(\cdot)$ is the embedding layer, Route denotes the routing strategy, expert set $\mathcal{E}_{t+1} = \mathcal{E}_t \cup \{\boldsymbol{e}_k\}$. Note that each expert only contains feature parameters, while routing is handled by a shared gating network. The active expert set $\hat{\mathcal{E}}_t = \{\boldsymbol{e}_k \mid p_k > 0\} \subseteq \mathcal{E}_t$ contributes to the edited representation via a weighted residual connection, given by

$$\boldsymbol{h}_{\text{out}}^{l_e} = \boldsymbol{h}^{l_e} + \sum_{\boldsymbol{e}_k \in \hat{\mathcal{E}}_t} p_k \cdot \Delta \boldsymbol{h}_k. \tag{4}$$

Our method is based on a pretrained MoE-based editor.

3 MOE-GRPO

In this section, we introduce the MoE-GRPO framework, which refines a pretrained MoE editor (LiveEdit (Chen et al., 2025a)) via reinforcement learning to improve multi-hop reasoning while preserving its foundational editing capabilities. We begin by modeling a Markov decision process in MoE-based lifelong knowledge editing. Next, a GRPO-inspired online RL framework is established with SVD-based structured perturbation for exploration. Finally, we introduce the rewards to provide intermediate supervision. The overview of MoE-GRPO is shown in Figure 2.

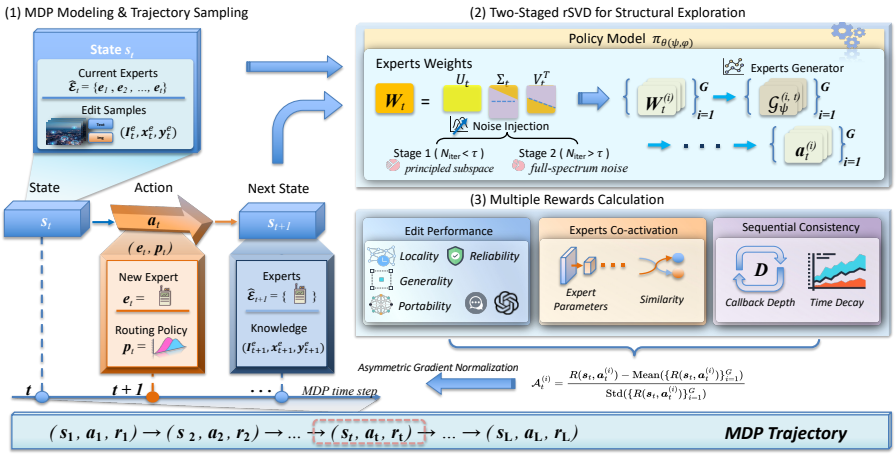


Figure 2: The overview of MoE-GRPO. We model the MoE-based auxiliary editor involved lifelong knowledge editing as an MDP, and leverage SVD with diminishing noise injected to singular values for structurally preserved random exploration. Advantages of each group are calculated via the edit performance reward, expert co-activation reward, and sequential consistency reward we devised, providing outcome-level supervision.

3.1 Modeling MoE-based MKE as an MDP

Rank-one hypernetwork-based lifelong knowledge editing can be formalized as a Markov decision process (MDP) (Li et al., 2025b), defined by $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \pi, \gamma)$. We extend this paradigm to the MoE-based lifelong multimodal knowledge editing. The t-th sample $(\boldsymbol{I}_t^e, \boldsymbol{x}_t^e, \boldsymbol{y}_t^e)$ is regarded as the transition trigger from time step t to $t+1$. The state at time step t is defined as $\boldsymbol{s}_t \triangleq \big(\hat{\mathcal{E}}_t, (\boldsymbol{I}_t^e, \boldsymbol{x}_t^e, \boldsymbol{y}_t^e)\big)$. The corresponding action, defined as $\boldsymbol{a}_t = (\boldsymbol{e}_t, \boldsymbol{p}_t)$, is deterministically computed through Eq. 3 and creates $\mathcal{E}_{t+1} = \mathcal{E}_t \cup \{\boldsymbol{e}_t\}$. The active expert set becomes $\hat{\mathcal{E}}_{t+1}$ via $\boldsymbol{p}_t$ selection. The edit target ends with the arrival of the next sample $(\boldsymbol{I}_{t+1}^e, \boldsymbol{x}_{t+1}^e, \boldsymbol{y}_{t+1}^e)$. Thus, the transition from $\boldsymbol{s}_t$ to $\boldsymbol{s}_{t+1}$ can be formalized as

$$\big(\hat{\mathcal{E}}_t, (\boldsymbol{I}_t^e, \boldsymbol{x}_t^e, \boldsymbol{y}_t^e)\big) \xrightarrow{\boldsymbol{a}_t = (\boldsymbol{e}_t, \boldsymbol{p}_t)} \big(\hat{\mathcal{E}}_{t+1}, (\boldsymbol{I}_{t+1}^e, \boldsymbol{x}_{t+1}^e, \boldsymbol{y}_{t+1}^e)\big). \tag{5}$$

The reward function $r_t = R(\boldsymbol{s}_t, \boldsymbol{a}_t)$ offers the outcome-level estimation. The initial state is represented as $\boldsymbol{s}_1 \triangleq \big(\hat{\mathcal{E}}_1, (\boldsymbol{I}_1^e, \boldsymbol{x}_1^e, \boldsymbol{y}_1^e)\big)$, where $\hat{\mathcal{E}}_1 = \varnothing$ for $\mathcal{E}_1 = \varnothing$. We treat the joint parameter set $\theta : \theta(\psi, \phi)$ as the policy, thus gaining the policy model represented as $\pi_{\theta(\psi,\phi)}$. For a sequence $\mathcal{D}_{L_T} = \{(\boldsymbol{I}_t^e, \boldsymbol{x}_t^e, \boldsymbol{y}_t^e)\}_{t=1}^{L_T}$ of length L_T randomly collected from $\mathcal{D}_{\text{edit}}$, the corresponding trajectory of the MDP can be formalized as:

$$\Big\{(\hat{\mathcal{E}}_1, (\boldsymbol{I}_1^e, \boldsymbol{x}_1^e, \boldsymbol{y}_1^e)), (\boldsymbol{e}_1, \boldsymbol{p}_1), r_1, \cdots, (\hat{\mathcal{E}}_{L_T}, (\boldsymbol{I}_{L_T}^e, \boldsymbol{x}_{L_T}^e, \boldsymbol{y}_{L_T}^e)), (\boldsymbol{e}_{L_T}, \boldsymbol{p}_{L_T}), r_{L_T}\Big\} \mapsto \Big\{\boldsymbol{s}_1, \boldsymbol{a}_1, r_1, \cdots, \boldsymbol{s}_{L_T}, \boldsymbol{a}_{L_T}, r_{L_T}\Big\} \tag{6}$$

3.2 GRPO for Lifelong Multimodal Knowledge Editing

Modeling GRPO in Knowledge Editing. RL methods such as PPO (Schulman et al., 2017) and DPO (Rafailov et al., 2023) require an additional reward model for advantage estimation. In contrast, group relative policy optimization (GRPO) (Shao et al., 2024) encourages a group-relative comparison-based normalization for advantages, learning a policy model π_θ by maximizing the following objective:

$$\mathbb{E}_{\{\boldsymbol{o}_i\}\sim\pi_\theta}\left[\frac{1}{G}\sum_{i=1}^{G}\left(\frac{1}{|\boldsymbol{o}_i|}\sum_{t=1}^{|\boldsymbol{o}_i|}\min\Big(\boldsymbol{\rho}_{i,t}\hat{A}_i,\ \text{clip}\left(\boldsymbol{\rho}_{i,t}, 1-\epsilon, 1+\epsilon\right)\hat{A}_i\Big) - \beta\,\hat{\mathbb{D}}_{\text{KL}}\left[\pi_\theta \,\|\, \pi_{\text{ref}}\right]\right)\right], \tag{7}$$

where $\boldsymbol{\rho}_{i,t} = \dfrac{\pi_\theta(\boldsymbol{o}_t^i | \boldsymbol{q}, \boldsymbol{o}_{<t}^i)}{\pi_{\theta_{\text{old}}}(\boldsymbol{o}_t^i | \boldsymbol{q}, \boldsymbol{o}_{<t}^i)}$ is clipped into a lower-bound trust region for conservative policy improvement. However, under MKE settings, the LVLM backbone is frozen, and gradients must propagate through a narrow interface layer. This leads to systematic underestimation of policy gradients, making conservative clipping overly restrictive. Instead of the conservative lower-bound clipping

used in PPO/GRPO, we therefore adopt an upper-biased clipped surrogate objective, which amplifies gradient signals when policy updates are under-constrained. We provide a formal derivation and a side-by-side comparison with the standard min-based objective in Appendix J.

$$\mathcal{J}_{\text{MoE-GRPO}}(\theta) = \mathbb{E}_{(\boldsymbol{I}_t, \boldsymbol{x}_t, \boldsymbol{y}_t)\sim\mathcal{D}_{(L_T)}\subset\mathcal{D}_{\text{edit}},\ \{\boldsymbol{a}_t^{(i)}\}_{i=1}^{G}\sim\pi_\theta(\cdot|\boldsymbol{s}_t)}$$
$$\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{L_T}\sum_{t=1}^{L_T}\left(\max\left(r_\theta^{(i,t)}\mathcal{A}_t^{(i)},\ \text{clip}\left(r_\theta^{(i,t)},\ 1-\epsilon,\ 1+\epsilon\right)\mathcal{A}_t^{(i)}\right) - \beta\,\hat{\mathbb{D}}_{KL}\left[\pi_\theta||\pi_{\text{ref}}\right]\right)\right], \quad (8)$$

Gradient Compensation

where $\delta_L \leq L_T \leq \Delta_L$, δ_L and Δ_L are the minimum and maximum sequential editing lengths, respectively. Advantages are given by the normalized group-relative rewards as $\mathcal{A}_t^{(i)} = \frac{R(\boldsymbol{s}_t,\boldsymbol{a}_t^{(i)}) - \text{Mean}(\{R(\boldsymbol{s}_t,\boldsymbol{a}_t^{(j)})\}_{j=1}^{G})}{\text{Std}(\{R(\boldsymbol{s}_t,\boldsymbol{a}_t^{(j)})\}_{j=1}^{G})}$. We apply asymmetric normalization across the two parameter groups, where ψ receives raw gradients while ϕ undergoes norm-based clipping. We empirically find that this consistently outperforms the other four possible configurations (Tab.3), described as:

$$\psi \leftarrow \psi + \eta\cdot\nabla_\psi\mathcal{J}_{\text{MoE-GRPO}}, \quad \phi \leftarrow \phi + \eta\cdot\nabla_\phi\mathcal{J}_{\text{MoE-GRPO}}\cdot\min\left(1,\ \frac{1}{\|\nabla_\phi\mathcal{J}_{\text{MoE-GRPO}}\|_2}\right). \quad (9)$$

Two-stage Structured Knowledge Exploration via SVD Perturbation. The policy model $\pi_{\theta(\psi,\phi)}$ is deterministic and is structurally sensitive, while Eq. 8 requires stochastic action sampling. To bridge this gap, we introduce a structured perturbation mechanism based on full singular value decomposition (SVD) with noise injected into the singular values of the expert parameter matrices. We distinguish three levels of indexing: the time step t in the RL trajectory, the sample index $i \in \{1,\ldots,G\}$ in GRPO, and the singular value index $j \in \{1,\ldots,r\}$. Each MoE expert in the LiveEdit backbone is parameterized by *both* a down-projection $\mathbf{C}_{e_k}^t$ (key-like, mapping hidden states into the low-rank expert subspace) and an up-projection $\mathbf{R}_{e_k}^t$ (value-like, mapping back to the residual stream). SVD-based perturbation is applied to *both* matrices independently, not to either one in isolation. Given the expert parameter matrices $\mathbf{C}_{e_k}^t, \mathbf{R}_{e_k}^t \in \mathbb{R}^{r\times d_m}$ at time step t, we apply SVD independently to each:

$$\mathbf{C}_{e_k}^t = \mathbf{U}_{\text{C},t}\,\mathbf{\Sigma}_{\text{C},t}\,\mathbf{V}_{\text{C},t}^\top, \qquad \mathbf{R}_{e_k}^t = \mathbf{U}_{\text{R},t}\,\mathbf{\Sigma}_{\text{R},t}\,\mathbf{V}_{\text{R},t}^\top, \quad (10)$$

where $\mathbf{\Sigma}_{\text{C},t} = \text{diag}(\sigma_{1,\text{C}}^t,\cdots,\sigma_{r,\text{C}}^t)$ and $\mathbf{\Sigma}_{\text{R},t} = \text{diag}(\sigma_{1,\text{R}}^t,\cdots,\sigma_{r,\text{R}}^t)$. We adopt a two-stage exploration schedule over training iterations with a threshold of τ. In the early stage ($N_{\text{iter}} \leq \tau$), we inject normalized, magnitude-aware noise into the singular values of both matrices:

$$\sigma_{j,\mathcal{K}}^{(i,t)} = \sigma_{j,\mathcal{K}}^t + \frac{|\sigma_{j,\mathcal{K}}^t|}{\max_{1\leq k\leq r}|\sigma_{k,\mathcal{K}}^t| + \mu}\,\eta_{j,\mathcal{K}}^{(i,t)}, \quad (11)$$

where $\mathcal{K} \in \{\text{C},\text{R}\}$, $i \in \{1,\cdots,G\}$ and $\eta_{j,\text{C}}^{(i,t)}, \eta_{j,\text{R}}^{(i,t)} \sim \mathcal{N}(0,\alpha^2)$. This confines perturbations within the principal subspace, preserving dominant knowledge components while enabling controlled exploration. In the later stage ($N_{\text{iter}} > \tau$), we switch to full-spectrum perturbation with fixed noise scale $\sigma_{j,\mathcal{K}}^{(i,t)} = \sigma_{j,\mathcal{K}}^t + \alpha\,\eta_{j,\mathcal{K}}^{(i,t)}$, $\mathcal{K} \in \{\text{C},\text{R}\}$. The corresponding perturbed parameters are reconstructed as $\mathbf{C}_{e_k}^{(i,t)} = \mathbf{U}_{\text{C},t}\,\text{diag}\big(\sigma_{1,\text{C}}^{(i,t)},\ldots,\sigma_{r,\text{C}}^{(i,t)}\big)\,\mathbf{V}_{\text{C},t}^\top, \mathbf{R}_{e_k}^{(i,t)} = \mathbf{U}_{\text{R},t}\,\text{diag}\big(\sigma_{1,\text{R}}^{(i,t)},\ldots,\sigma_{r,\text{R}}^{(i,t)}\big)\,\mathbf{V}_{\text{R},t}^\top$. Repeating this process yields G perturbed parameter pairs $\{(\mathbf{C}_{e_k}^{(i,t)}, \mathbf{R}_{e_k}^{(i,t)})\}_{i=1}^{G}$ as a group of actions $\{\boldsymbol{a}_t^{(i)}\}_{i=1}^{G}$. This stochastic process addresses the structural issue of how to effectively explore the parameter space while preserving the original knowledge structure.

3.3 Reward Design for MoE-GRPO

We argue that effective post-edit reasoning depends not on uniform knowledge distribution, but on whether salient experts produce compatible residual updates, enabling stable and coherent composition across reasoning steps. Thus we design rewards consisting of *edit performance reward*, *expert co-activation reward*, and *sequential consistency reward* to provide outcome-level supervision.

Edit Performance Reward. Since the policy model is initialized from a pretrained MoE-based auxiliary editor, we first ensure the basic editing performance, including *locality*, *reliability*, and *generality* Li et al. (2025b). For an edit sample $(\boldsymbol{I}_t, \boldsymbol{x}_t, \boldsymbol{y}_t) \sim \mathcal{D}_{L_T}$ at the time step t, we select the unrelated knowledge $(\boldsymbol{I}_t^{\text{loc}}, \boldsymbol{x}_t^{\text{loc}}, \boldsymbol{y}_t^{\text{loc}})$ and the rephrased knowledge $(\boldsymbol{I}_t^{\text{gen}}, \boldsymbol{x}_t^{\text{gen}}, \boldsymbol{y}_t^{\text{gen}})$ to compute

$$\begin{aligned} &\mathcal{L}_{\text{loc}} = \text{KL}[f_{\mathcal{W}}(\cdot \mid \boldsymbol{I}_t^{\text{loc}}, \boldsymbol{x}_t^{\text{loc}}) \parallel f_{\mathcal{W} \cup \hat{\mathcal{E}}_t}(\cdot \mid \boldsymbol{I}_t^{\text{loc}}, \boldsymbol{x}_t^{\text{loc}})], \\ &\mathcal{L}_{\text{rel}} = -\log f_{\mathcal{W} \cup \hat{\mathcal{E}}_t}(\boldsymbol{y}_t \mid \boldsymbol{I}_t, \boldsymbol{x}_t), \qquad \mathcal{L}_{\text{gen}} = -\log f_{\mathcal{W} \cup \hat{\mathcal{E}}_t}(\boldsymbol{y}_t^{\text{gen}} \mid \boldsymbol{I}_t^{\text{gen}}, \boldsymbol{x}_t^{\text{gen}}), \end{aligned} \tag{12}$$

where $f_{\mathcal{W} \cup \hat{\mathcal{E}}_t}(\cdot)$ denotes the original LVLM with the pretrained parameters $\mathcal{W}$ and the currently selected experts $\hat{\mathcal{E}}_t$. We introduce coefficients λ_{rel} and λ_{gen} to balance the sub-rewards as

$$\mathcal{R}_{epr} = -\left(\mathcal{L}_{\text{loc}} + \lambda_{\text{rel}}\mathcal{L}_{\text{rel}} + \lambda_{\text{gen}}\mathcal{L}_{\text{gen}}\right) \tag{13}$$

We empirically observe that adding an explicit portability reward (e.g., $\mathcal{L}_{\text{port}}$ computed on auxiliary multi-hop QA pairs) does *not* further improve—and can slightly hurt—multi-hop reasoning performance compared with the current three-term formulation, as discussed in Appendix D.

Expert Co-activation Reward. As the residual correction signal $\Delta \boldsymbol{h}_k^t = \rho(\boldsymbol{h}^{l_e}\mathbf{C}_{\boldsymbol{e}_k^t}^{\top})\mathbf{R}_{\boldsymbol{e}_k^t}$ from expert $\boldsymbol{e}_k^t$ $(k = 1, 2, ..., E_t)$ is injected into the backbone with a gating probability $p_k^{(t)}$ at the time step t, degenerate experts with low-magnitude parameter matrices may contribute negligibly to the residual update. More critically, routing misalignment, where the gating network g_ϕ assigns high probability to degenerate experts with low-magnitude parameter matrices, can corrupt the residual injection. This motivates the need to quantify the relative contribution of experts based on their parameter capacity. Let $\{\mathbf{R}_{i_1}, \ldots, \mathbf{R}_{i_{E_t}}\}$ denote the $\mathbf{R}$-matrices of all experts in $\hat{\mathcal{E}}_t$ (cf. Eq. 4). We define a normalized capacity distribution over experts to capture their relative contributions based on parameter magnitude:

$$\nu_{\boldsymbol{e}_t}^{(u)} = \frac{\exp(\|\mathbf{R}_{i_u}\|_2)}{\sum_{j=1}^{E_t} \exp\big(\|\mathbf{R}_{i_j}\|_2\big)}, \quad u = 1, \ldots, E_t. \tag{14}$$

to penalize the situation where expert parameter matrices degenerate toward sparsity. As discussed above, we compute the pairwise product of the normalized sparsity weighted by similarity:

$$\mathcal{R}_{ecr} = -\sum_{1 \le i < j \le E_t} \Big(\nu_{\boldsymbol{e}_t}^{(i)} \cdot \nu_{\boldsymbol{e}_t}^{(j)} \cdot \text{Sim}(\Delta \boldsymbol{h}_i^t, \Delta \boldsymbol{h}_j^t)\Big), \tag{15}$$

To quantify expert pairwise similarity, we define the normalized cosine similarity $\text{Sim}(\boldsymbol{u}, \boldsymbol{v}) = \frac{1}{2}\Big(1 + \frac{\langle \boldsymbol{u}, \boldsymbol{v} \rangle}{\|\boldsymbol{u}\| \cdot \|\boldsymbol{v}\|}\Big) \in [0, 1]$, which maps the standard cosine similarity from $[-1, 1]$ to $[0, 1]$ to prevent negative rewards caused by obtuse angles.

Sequential Consistency Reward. Under MKE settings, editing new knowledge may cause expert degradation or knowledge attenuation (Li & Chu, 2024), manifested as abrupt shifts in the *sparsity distribution vector* across consecutive edits, which indicate the inaccessibility of previously edited knowledge. To reduce such abrupt shifts while tolerating gradual structural evolution, we maintain a callback buffer of depth D that records the most recent *expert parameter sparsity vector* $\{\mathcal{P}[\boldsymbol{e}_{S-D+1}], \ldots, \mathcal{P}[\boldsymbol{e}_S]\}$, where $\mathcal{P}[\boldsymbol{e}_t] = \begin{bmatrix} \nu_{\boldsymbol{e}_t}^{(1)} & \cdots & \nu_{\boldsymbol{e}_t}^{(E_t)} \end{bmatrix}^{\top}$ use normalized sparsity to indicate the relative activation level of the parameters. Given the current step S (i.e., editing the S-th sample), the *sequential consistency reward* is calculated by

$$\mathcal{R}_{scr}^{(S)} = \frac{\sum_{t=S-D+1}^{S-1} \gamma^{S-t} \cdot \text{Sim}(\mathcal{P}[\boldsymbol{e}_S], \mathcal{P}[\boldsymbol{e}_t])}{\sum_{t=S-D+1}^{S-1} \gamma^{S-t}} = \frac{1-\gamma}{(1-\gamma^{D-1})} \sum_{t=S-D+1}^{S-1} \gamma^{S-t-1} \cdot \text{Sim}(\mathcal{P}[\boldsymbol{e}_S], \mathcal{P}[\boldsymbol{e}_t]), \tag{16}$$

which measures how consistently the model routes across consecutive edits, with exponential temporal decay favoring more recent routing patterns. We then organize all rewards with coefficients α_{ecr} and α_{scr}:

$$R(\boldsymbol{s}_t, \boldsymbol{a}_t) = \mathcal{R}_{epr} + \alpha_{\text{ecr}}\mathcal{R}_{ecr} + \alpha_{\text{scr}}\mathcal{R}_{scr}^{(t)}. \tag{17}$$

This group of rewards $\{r_t^{(i)}\}_{i=1}^G = \{R(\boldsymbol{s}_t, \boldsymbol{a}_t^{(i)})\}_{i=1}^G$ produces advantages $\mathcal{A}_t^{(i)}$ in Eq. 8.

Baseline	Editors	1 Edit							100 Edits							1000 Edits						
		2-hop		3-hop		4-hop		Avg	2-hop		3-hop		4-hop		Avg	2-hop		3-hop		4-hop		Avg
		Orig	Reph	Orig	Reph	Orig	Reph		Orig	Reph	Orig	Reph	Orig	Reph		Orig	Reph	Orig	Reph	Orig	Reph	
BLIP2-OPT (2.7B)	FT	27.68	27.78	27.82	27.98	30.63	30.59	28.73	11.13	11.42	11.71	11.72	12.54	12.59	11.85	7.69	7.68	9.57	9.47	9.87	9.77	9.01
	MEND	0.96	0.91	0.25	0.26	0.37	0.28	0.51	0.96	1.01	1.06	1.03	1.20	1.22	1.08	1.23	1.22	0.47	0.53	6.24	0.64	1.72
	SERAC	25.96	25.65	27.90	27.68	32.17	31.99	28.56	25.92	25.61	27.97	27.75	32.20	32.01	28.57	27.34	27.22	26.18	27.55	32.87	32.49	28.94
	RECIPE	25.54	25.51	26.61	25.87	31.55	31.12	27.70	25.98	25.92	27.25	27.14	32.72	32.64	28.61	30.04	29.86	31.19	30.97	35.61	35.45	32.19
	RLEdit	25.96	25.65	27.90	27.68	32.17	31.99	28.56	25.92	25.61	27.97	27.75	32.20	32.01	28.57	26.18	25.77	27.97	27.55	32.87	32.49	28.81
	PokeMQA	26.84	26.63	27.94	27.70	32.17	31.99	28.88	27.07	26.78	28.21	27.86	32.22	32.06	29.03	27.72	27.33	27.98	27.71	32.94	32.63	29.39
	ReasonKE	36.19	35.85	34.73	34.82	35.65	**36.00**	**35.55**	31.83	31.57	31.61	31.54	32.07	31.94	31.76	31.09	31.50	31.82	31.60	30.04	30.53	31.10
	LEMoE	25.95	25.65	27.90	27.67	32.16	31.98	28.55	31.54	31.74	31.46	31.08	33.14	33.02	31.99	30.75	30.84	32.77	31.45	35.59	35.77	32.60
	LiveEdit	26.85	26.95	27.52	28.63	29.95	32.07	28.66	27.00	27.25	27.53	28.65	30.19	31.98	28.76	27.56	27.15	28.27	29.56	30.44	31.96	29.16
	MoE-GRPO	**36.61**	**35.11**	**35.64**	**33.33**	**36.49**	35.10	35.38	**42.67**	**39.07**	**42.22**	**38.71**	**44.63**	**43.16**	**41.74**30.48%	**40.45**	**39.00**	**40.47**	**38.84**	**42.65**	**41.56**	**40.50**24.23%
MiniGPT4 -vicuna (7B)	FT	21.27	21.93	20.30	20.67	21.27	21.84	21.21	5.50	5.67	6.28	6.15	6.96	7.10	6.28	4.78	4.64	7.88	7.74	8.26	8.08	6.90
	MEND	0.45	0.44	0.21	0.24	0.22	0.28	0.31	0.00	0.00	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	SERAC	38.81	37.06	41.25	41.05	39.81	39.44	39.57	39.22	39.38	41.07	41.20	44.01	43.87	41.46	40.02	39.64	40.81	40.27	41.33	42.76	40.81
	RECIPE	39.01	38.86	39.83	39.74	40.12	40.10	39.61	38.66	38.59	39.96	39.87	40.91	40.79	39.79	38.92	38.33	40.17	40.05	41.30	41.19	39.99
	RLEdit	0.21	0.21	0.03	0.03	0.00	0.00	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	PokeMQA	38.02	38.18	40.51	40.60	42.59	43.04	40.49	38.02	38.18	40.62	40.71	42.61	43.07	40.53	38.26	38.52	40.73	40.88	42.71	43.12	40.70
	ReasonKE	**40.81**	**40.94**	40.43	**40.85**	41.22	42.45	41.12	**42.56**	**42.49**	42.65	**43.04**	44.22	43.24	**43.25**	**42.64**	42.69	**42.03**	**42.47**	41.00	41.32	42.02
	LEMoE	37.12	37.18	39.54	40.38	41.45	41.06	39.46	29.59	29.43	28.77	28.99	30.63	30.37	29.63	24.72	25.83	26.87	26.15	24.43	23.79	25.30
	LiveEdit	37.66	38.19	39.01	39.96	41.63	41.75	39.69	37.73	38.24	39.12	40.10	41.66	41.91	39.79	38.35	38.66	39.52	40.08	41.75	42.53	40.14
	MoE-GRPO	39.82	40.33	**40.61**	40.76	**46.08**	**44.58**	**41.94**1.99%	40.53	40.81	**43.63**	42.87	**44.83**	**44.63**	42.88	42.37	**42.83**	41.24	42.26	**44.35**	**44.74**	**42.93**2.17%

Table 1: Portability (multi-hop reasoning) evaluation across different numbers of edits on BLIP2-OPT and MiniGPT4-vicuna. Results are shown for original (*Orig*) and rephrased (*Reph*) multimodal knowledge pairs. **MoE-GRPO** achieves strong portability and robustness under rephrasing.

# Edit	Ablation Settings	T-Loc	M-Loc	T-Gen	M-Gen	Rel	1-*hop*		2-*hop*		3-*hop*		4-*hop*		Avg
							Orig	Reph	Orig	Reph	Orig	Reph	Orig	Reph	
100	LEMoE	**45.54**	**48.85**	46.16	**45.99**	**46.09**	29.61	29.38	31.15	31.20	**31.71**	**31.57**	33.59	**33.65**	31.48
	+ **MoE-GRPO**	45.33	47.77	**46.33**	45.64	45.76	**29.79**	**29.70**	**31.35**	**31.57**	31.22	31.31	**33.77**	33.55	**31.53**
1000	LEMoE	**44.34**	44.51	**38.71**	**36.02**	**35.74**	29.83	29.83	29.95	30.00	31.08	31.42	35.17	34.84	31.52
	+ **MoE-GRPO**	43.98	**45.42**	37.66	35.97	35.68	**31.54**	**31.59**	**33.36**	**33.23**	**32.62**	**32.46**	**35.21**	**34.89**	**33.11**

Table 2: Reinforcing LEMoE with MoE-GRPO.

4 Experiments

4.1 Experimental Setup

Baseline LVLMs & Baseline Methods. Our experiments are conducted on three scaled LVLMs: BLIP2-OPT (2.7B) (Li et al., 2023), LLaVA-v1.5 (7B) (Liu et al., 2023), and MiniGPT4-Vicuna (7B) (Zhu et al., 2023). We compare the MoE-GRPO method with the following knowledge editing baselines. Fine-tuning: *FT* (Zhu et al., 2020). Editor-based: *RLEdit* (Li et al., 2025b). Parameter-based: *RECIPE* (Chen et al., 2024), *LEMoE* (Wang & Li, 2024a), *LiveEdit* (Chen et al., 2025a). RAG-based reasoning: PokeMQA (Gu et al., 2024), ReasonKE (Wu et al., 2025). Unless otherwise specified, MoE-GRPO initializes the policy model $\pi_{\theta(\psi,\phi)}$ from the pretrained weights of LiveEdit (Chen et al., 2025a) and updates ψ, ϕ via the proposed RL objective; the LVLM backbone $\mathcal{W}$ remains frozen throughout. We report the settings in Appendix H.

Datasets & Metrics. We mainly train and evaluate MoE-GRPO on VLKEB (Huang et al., 2024), E-VQA (Cheng et al., 2023), and MQuAKE (Zhong et al., 2023). For editing performance and multi-hop reasoning evaluation, we train MoE-GRPO on VLKEB-train and evaluate on VLKEB-multihop. We mainly calculate the accuracy across multi-hops, and also provide *reliability*, *locality* (textual and modal), and *generality* (textual and modal), as well as *portability* (1–4 hops reasoning).

4.2 MoE-GRPO Main Results

To comprehensively evaluate multi-hop reasoning, we evaluate both original (*Orig*) and rephrased (*Reph*) knowledge pairs from VLKEB (see Appendix E). Table4 summarizes the main results. Existing parameter-editing methods (MEND, RLEdit) struggle with multi-hop reasoning, while external-memory methods (SERAC, RECIPE, LEMoE, LiveEdit) suffer from degraded editing quality as the number of edits increases and achieve limited portability. PokeMQA and ReasonKE improve multi-hop portability but sacrifice general editing quality. In contrast, MoE-GRPO achieves a favorable trade-off between editing quality and multi-hop portability: it incurs only a minor degradation on single-hop metrics while substantially improving portability. This trade-off reflects the need for broader parametric updates to support multi-hop reasoning, compared with the more conservative locality constraints of LiveEdit. At 100 edits, MoE-GRPO improves multi-hop portability over LiveEdit by 30.48% on BLIP2-OPT, 7.64% on LLaVA, and 2.17% on MiniGPT4, while maintaining comparable single-hop performance. More results are provided in Appendix G.

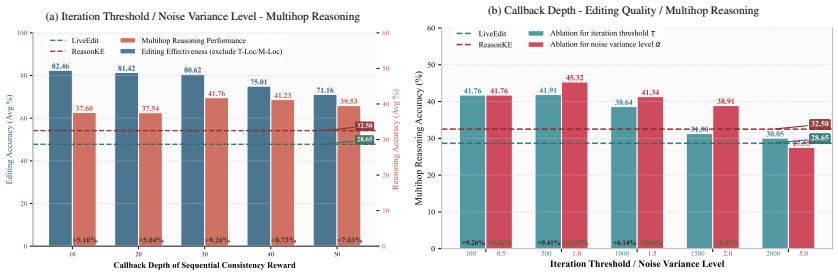


Figure 3: Ablations on callback depth, iteration threshold, and noise variance level.

4.3 Ablation Studies

Ablation Settings	T-Loc	M-Loc	T-Gen	M-Gen	Rel	2-*hop*		3-*hop*		4-*hop*	
						Orig	Reph	Orig	Reph	Orig	Reph
MoE-GRPO	**100.00**	**100.00**	82.38	76.42	83.29	**42.67**	**39.07**	42.22	**38.71**	44.63	**43.16**
- $\mathcal{R}_{\text{epr}}$	100.00	100.00	40.18	37.81	37.88	32.19	32.19	32.91	32.76	36.60	36.01
- $\mathcal{R}_{\text{ecr}}$	100.00	100.00	71.68	64.18	68.84	40.15	38.75	39.53	38.52	40.36	39.71
- $\mathcal{R}_{\text{ncr}}$	100.00	100.00	76.12	64.72	74.17	36.26	35.06	37.34	36.92	38.04	37.59
$\min(\cdot)$	100.00	100.00	63.34	55.37	61.00	34.93	32.58	33.30	29.04	32.27	30.65
$\nabla_{\varpi,\varpi}\mathcal{J}$	100.00	100.00	63.16	57.05	60.36	39.81	38.22	39.31	39.02	40.21	40.44
$\nabla_{\varpi,\phi}\mathcal{J}$	100.00	100.00	49.24	36.08	48.02	27.82	27.56	30.99	30.78	33.03	33.58
$\nabla_{\psi,\phi}\mathcal{J}$	100.00	100.00	52.35	47.14	49.22	41.21	38.50	**46.69**	35.38	**47.30**	40.40
Gaussian	100.00	100.00	**94.75**	**86.27**	**94.40**	29.06	28.94	30.26	30.41	32.78	33.23

Table 3: Ablation studies on reinforcement learning components compared with full MoE-GRPO.

Impact of RL Components. We conduct ablation studies to validate the contribution of each component in MoE-GRPO. As shown in Table 3, the edit performance reward $\mathcal{R}_{epr}$ mirrors the three single-hop evaluation dimensions along which the baseline editor was originally trained (locality, reliability, generality), ensuring that policy updates remain anchored to the editor's operating regime. Removing any of these terms risks collapsing the editor's behavior along the corresponding axis (e.g., dropping $\mathcal{L}_{\text{loc}}$ may cause catastrophic forgetting of unrelated facts; dropping $\mathcal{L}_{\text{gen}}$ may overfit to exact edit prompts) The results on gradient normalization on the two policy components reveal a clear asymmetry. Normalizing ϕ alone consistently degrades generalization, while global normalization enhances multi-hop reasoning at the cost of generality degradation. We attribute this to the distinct sensitivity of the discrete routing mechanism in ϕ to gradient magnitude under aggressive $\max(\cdot)$ updates. This asymmetry reflects the distinct functional roles of the two components: ϕ governs discrete routing and its gradient instability under the $\max(\cdot)$ operator requires regularization, whereas ψ drives knowledge content generation and requires unrestricted gradient flow to maintain editing expressivity.

SVD Perturbation Exploration Mechanisms. The "Gaussian" row in Tab. 3 replaces the SVD-based perturbation with i.i.d. Gaussian noise injected directly into the entries of $(\mathbf{C}, \mathbf{R})$, keeping all other components identical. This variant corresponds to an unstructured exploration baseline: it provides the same amount of stochasticity required by GRPO action sampling but without confining perturbations to the principal subspace. Its multi-hop accuracy drops to 29–33% (vs. 39–44% for the full method), confirming that the principal-subspace confinement, rather than noise injection per se, is what drives the gain. A "no exploration" baseline (i.e., a deterministic policy that emits G identical actions) is not a meaningful comparator under GRPO: the group-relative advantage $\mathcal{A}_t^{(i)}$ vanishes when all G actions coincide, yielding zero gradient throughout training. SVD-based perturbation is therefore not one option among many but the minimal mechanism that simultaneously (i) supplies the stochasticity GRPO requires and (ii) preserves the dominant singular components of the pretrained expert. We provide a theoretical justification of why KL-divergence tuning cannot substitute this structural regularization in Appendix K.

4.4 Discussions

Improvement for Diverse MoE-based MKE Baselines. Our MoE-GRPO framework can be readily extended to other MoE-based knowledge editing approaches. As shown in Table 2, we report the multi-hop reasoning capability improvement that MoE-GRPO achieves on LEMoE (Wang & Li, 2024a) with BLIP2-OPT-2.7B. However, the gains in multi-hop reasoning are smaller than those observed with the original baseline. We attribute this to the reduced scale of the auxiliary editor.

Quantifying Trade-offs. To make the trade-off explicit, we examine the Pareto frontier between average single-hop accuracy (Rel, T-Loc, M-Loc, T-Gen, M-Gen) and multi-hop portability across all baseline methods in Table 4. MoE-GRPO lies on the Pareto frontier, indicating that no baseline

simultaneously achieves higher portability and higher single-hop accuracy at the same edit count. This confirms that the trade-off is not a defect specific to our method but a fundamental characteristic of the lifelong multi-hop editing task. The SVD-based perturbation and asymmetric gradient normalization mitigate but cannot eliminate this trade-off, as confirmed by the gradient-normalization ablation in Table 3.

As shown in Figure 3, we further analyze how the SVD perturbation noise level α, the iteration threshold τ, and the callback depth D influence multi-hop reasoning. To disentangle their effects, we follow a one-factor-at-a-time protocol: when varying α we fix $\tau = 100$, and when varying τ we fix $\alpha = 0.5$. All ablation studies begin with the default configuration $(\alpha, \tau, D) = (0.5, 100, 30)$. We have the following observations. **Obs1**: Increasing the callback depth from 10 to 30 lifts the average multi-hop accuracy from 37.60% to 41.76%, while editing quality (generality and reliability) degrades. **Obs2.** With fixed iteration threshold $\tau = 100$, moderate noise variance ($\alpha \in [0.3, 1.0]$) expands the search range, but once the variance exceeds 1.5, multi-hop performance drops sharply and falls below the baseline method. **Obs3.** With fixed noise variance $\alpha = 0.5$, the iteration threshold governs the duration of the magnitude-aware noise across the two perturbation stages. As we fix the total iterations, an excessively prolonged cold start severely delays the reasoning learning process. For the iteration threshold $\tau \in [100, 2000]$, its influence on multi-hop reasoning capabilities shows a trend similar to that of noise variance level.

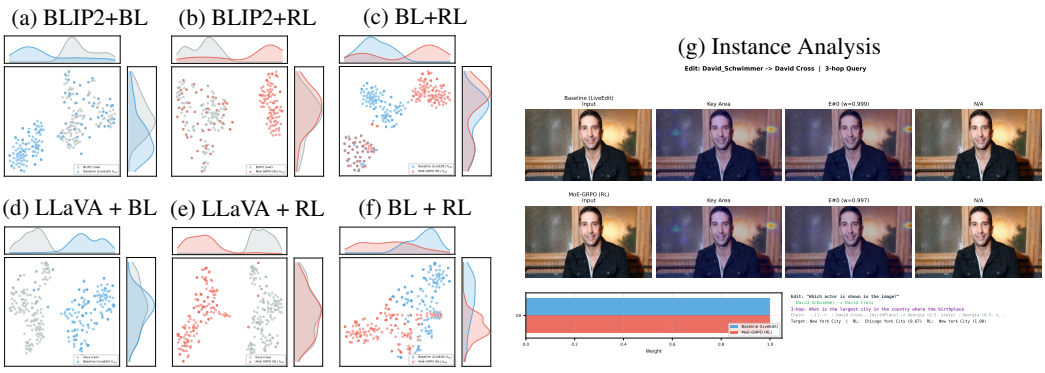


Figure 4: Visualization and instance analysis. BL (baseline) denotes the pretrained distribution of LiveEdit, RL denotes our MoE-GRPO. (a)–(c) are pairwise correspondences based on BLIP2-OPT-2.7B; (d)–(f) are based on LLaVA-1.5-7B. (g) is an instance case visualized with BLIP2-OPT-2.7B.

4.5 Visualization and Instance Analysis

As discussed, MoE-GRPO is designed for structural space exploration, aiming to learn a distinct editing subspace that enables more focused and consistent corrections to the pretrained representations. To empirically validate this design choice, we visualize the distributional shifts of hidden representations before and after 100 edits via t-SNE (Van der Maaten & Hinton, 2008), providing qualitative evidence for how MoE-GRPO reshapes the latent space differently from the baseline. As shown in Figure 4, the MoE-GRPO-corrected representations form a compact cluster that is substantially shifted from the pretrained distribution, whereas the baseline exhibits a weaker and more dispersed shift. We also report the instance analysis above; more results can be found in Appendix I.

5 Conclusion

We introduce MoE-GRPO, a reinforcement learning framework for lifelong multimodal knowledge editing that bridges the supervision gap between isolated edits and coherent multi-hop reasoning by optimizing mixture-of-experts routing. MoE-GRPO adopts two-stage SVD-based perturbation for structured exploration and inverted upper-biased clipping for gradient compensation. Canonically, asymmetrically normalized gradients in GRPO amplify the advantages towards the critical parameter within the policy model. Extensive experiments demonstrate superior performance in lifelong multi-hop question answering tasks and comparable locality, generality, and reliability. Ablation studies reveal a potential trade-off between editing portability and other metrics.

AI USE STATEMENT

Generative AI tools were used solely for language editing and improving the clarity and presentation of this manuscript. All AI-assisted content was carefully reviewed by the authors. The authors take full responsibility for the research ideas, mathematical formulations, experimental results, and the overall content of the final manuscript.

REFERENCES

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

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pp. 2425–2433, 2015.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

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, Chengyu Wang, Dakan Wang, Taolin Zhang, Wangyue Li, and Xiaofeng He. Lifelong knowledge editing for vision language models with low-rank mixture-of-experts. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 9455–9466, 2025a.

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, 2025b.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.

Siyuan Cheng, Bozhong Tian, Qingbin Liu, Xi Chen, Yongheng Wang, Huajun Chen, and Ningyu Zhang. Can we edit multimodal large language models? *arXiv preprint arXiv:2310.08475*, 2023.

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. *Advances in neural information processing systems*, 36:49250–49267, 2023.

Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. *arXiv preprint arXiv:2104.08164*, 2021.

Qingxiu Dong, Damai Dai, Yifan Song, Jingjing Xu, Zhifang Sui, and Lei Li. Calibrating factual knowledge in pretrained language models. *arXiv preprint arXiv:2210.03329*, 2022.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Xiang Wang, Xiangnan He, and Tat-Seng Chua. Alphaedit: Null-space constrained knowledge editing for language models. arxiv abs/2410.02355 (2024), 2024.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. Causal abstractions of neural networks. *Advances in Neural Information Processing Systems*, 34:9574–9586, 2021.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6904–6913, 2017.

Hengrui Gu, Kaixiong Zhou, Xiaotian Han, Ninghao Liu, Ruobing Wang, and Xin Wang. Pokemqa: Programmable knowledge editing for multi-hop question answering. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8069–8083, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Xiaoqi Han, Ru Li, Ran Yi, Hongye Tan, Zhuomin Liang, Victor Gutierrez Basulto, and Jeff Z Pan. Uncovering and mitigating transient blindness in multimodal model editing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 30934–30942, 2026.

Simao Herdade, Armin Kappeler, Kofi Boakye, and Joao Soares. Image captioning: Transforming objects into words. *Advances in neural information processing systems*, 32, 2019.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

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, 2025a.

Kailin Jiang, Ning Jiang, Yuchen Ren, Yuchen Li, Yifan Gao, Jinhe Bi, Yunpu Ma, Qingqing Liu, Xianhao Wang, Yifan Jia, et al. Mined: Probing and updating with multimodal time-sensitive knowledge for large multimodal models. 2025b.

Nora Kassner, Oyvind Tafjord, Hinrich Schütze, and Peter Clark. Beliefbank: Adding memory to a pre-trained language model for a systematic notion of belief. *arXiv preprint arXiv:2109.14723*, 2021.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *ICML*, 2023.

Qi Li and Xiaowen Chu. Can we continually edit language models? on the knowledge attenuation in sequential model editing. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 5438–5455, 2024.

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, Shezheng Song, Huijun Liu, Bin Ji, Xi Wang, Jun Ma, Jie Yu, Xiaodong Liu, Jing Wang, et al. Swea: Updating factual knowledge in large language models via subject word embedding altering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 24494–24502, 2025a.

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*, 2025b.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *NeurIPS*, 2023.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 26296–26306, 2024.

Tianci Liu, Ruirui Li, Yunzhe Qi, Hui Liu, Xianfeng Tang, Tianqi Zheng, Qingyu Yin, Monica Xiao Cheng, Jun Huan, Haoyu Wang, et al. Unlocking efficient, scalable, and continual knowledge editing with basis-level representation fine-tuning. *arXiv preprint arXiv:2503.00306*, 2025.

Ye Liu, Hui Li, Alberto Garcia-Duran, Mathias Niepert, Daniel Onoro-Rubio, and David S Rosenblum. Mmkg: multi-modal knowledge graphs. In *European Semantic Web Conference*, pp. 459–474. Springer, 2019.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.

Jiasen Lu, Christopher Clark, Sangho Lee, Zichen Zhang, Savya Khosla, Ryan Marten, Derek Hoiem, and Aniruddha Kembhavi. Unified-io 2: Scaling autoregressive multimodal models with vision, language, audio, and action. *arXiv preprint arXiv:2312.17172*, 2023.

Yifan Lu, Yigeng Zhou, Jing Li, Yequan Wang, Xuebo Liu, Daojing He, Fangming Liu, and Min Zhang. Knowledge editing with dynamic knowledge graphs for multi-hop question answering. In *Proceedings of the AAAI conference on artificial intelligence*, volume 39, pp. 24741–24749, 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.

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. PMLR, 2022.

Shiwen Ni, Dingwei Chen, Chengming Li, Xiping Hu, Ruifeng Xu, and Min Yang. Forgetting before learning: Utilizing parametric arithmetic for knowledge updating in large language models. *arXiv preprint arXiv:2311.08011*, 2023.

Haowen Pan, Xiaozhi Wang, Yixin Cao, Zenglin Shi, Xun Yang, Juanzi Li, and Meng Wang. Precise localization of memories: A fine-grained neuron-level knowledge editing technique for llms. *arXiv preprint arXiv:2503.01090*, 2025.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. *arXiv preprint arXiv:2311.03658*, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*, 2023. URL https://arxiv.org/abs/2305.18290.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pp. 1842–1850. PMLR, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. *URL https://arxiv. org/abs/2402.03300*, 2(3):5, 2024.

Jinwei Shi, Wenxuan Huang, Yu Xing, Yunhui Liu, Tao Zheng, Bin Chong, and Tieke He. Echoedit: Consistent multi-hop question answering via ripple control in knowledge editing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 19362–19370, 2026.

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

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

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

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, 2024a.

Renzhi Wang and Piji Li. Memoe: Enhancing model editing with mixture of experts adaptors. *arXiv preprint arXiv:2405.19086*, 2024b.

Yuchen Wu, Liang Ding, Li Shen, and Dacheng Tao. Robust knowledge editing via explicit reasoning chains for distractor-resilient multi-hop qa. *arXiv preprint arXiv:2509.01468*, 2025.

Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D Manning, and Christopher Potts. Reft: Representation finetuning for language models. *Advances in Neural Information Processing Systems*, 37:63908–63962, 2024.

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. *arXiv preprint arXiv:2305.13172*, 2023.

Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, et al. Glm-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026.

Zhange Zhang, Zhicheng Geng, Yuqing Ma, Tianbo Wang, Kai Lv, and Xianglong Liu. Conflict-aware knowledge editing in the wild: Semantic-augmented graph representation for unstructured text. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Zexuan Zhong, Zhengxuan Wu, Christopher D Manning, Christopher Potts, and Danqi Chen. Mquake: Assessing knowledge editing in language models via multi-hop questions. *arXiv preprint arXiv:2305.14795*, 2023.

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.

Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023.

A SEQUENTIAL EDITING AND OTHER EDITING SCENARIOS

Put simply, "sequential" emphasizes updating a certain number of structured knowledge items at one editing step, while "batch" emphasizes updating a batch of knowledge items in one step. For the standard problem of *sequential batch editing*, given a pre-edited model (i.e., the model before editing) G with initial weights (pretrained parameters) θ_0, and n samples of knowledge $\{(x_i, y_i)\}_{i=1}^{n}$ to be integrated into the LLM G_{θ_0}, sequential editing (Li & Chu, 2024) can be defined as follows

$$G_{\theta_t} = \mathcal{E}(G_{\theta_{t-1}}, S_t), \tag{18}$$

where $\mathcal{E}(\cdot, \cdot)$ represents the editing operation function and S_t is the set of edit samples containing the t-th item, where

$$\begin{cases} S_i \cap S_j = \varnothing, & \text{where } i \neq j \\ \bigcup_i S_i = \{x_j, y_j\}_{j=1}^{j=n}, & \text{where } \forall i \end{cases} \tag{19}$$

Here, for each t we have $|S_t| \triangleq n_t \geq 1$ satisfying $N = \sum_t n_t$. Specifically, as illustrated by (Li et al., 2025a; Wang & Li, 2024b;a), model editing scenarios can be divided into "batch editing", "sequential editing", and "sequential batch editing", represented as

Batch Editing

$$\sum_{i=1}^{N} (\tau_i, y_i) \mapsto \sum_{i=1}^{N} (\tau_i, y_i^*) \tag{20}$$

Sequential Editing

$$\sum_{i=1}^{N} [(\tau_i, y_i) \mapsto (\tau_i, y_i^*)] \tag{21}$$

Sequential Batch Editing

$$\bigcup_{S_k} \Big(\sum_{i \in S_k} [(\tau_i, y_i) \mapsto (\tau_i, y_i^*)] \Big) \tag{22}$$

where $(\tau_i, y_i) \mapsto (\tau_i, y_i^*)$ denotes an editing step while $\sum(\tau_i, y_i)$ denotes the updated knowledge series in one step.

B MULTI-HOP REASONING

The multi-hop portability task differs from open-domain multi-hop QA (e.g., HotpotQA, MuSiQue). In open-domain settings, the model retrieves evidence from a large corpus to synthesize an answer. In our setting, the model must perform closed-book reasoning over *internally edited facts*: the edit samples are injected into the model's parameters via the MoE editor, and the multi-hop questions are answered solely from the post-edit model's parametric knowledge, without any retrieval or external evidence. This formulation directly tests whether edited knowledge is integrated deeply enough to support compositional reasoning, which is the core capability that MoE-GRPO aims to improve.

Formally, let a piece of knowledge be represented as a triple $(\boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o}) \in \mathcal{D}$, where $\boldsymbol{s}$ denotes the subject, $\boldsymbol{r}$ the relation, and $\boldsymbol{o}$ the object. A multi-hop reasoning chain over $\mathcal{D}$ can be expressed as a sequence of dependent triples:

$$\mathcal{C} = \big[(\boldsymbol{s}_1, \boldsymbol{r}_1, \boldsymbol{o}_1),\ (\boldsymbol{s}_2, \boldsymbol{r}_2, \boldsymbol{o}_2),\ \ldots,\ (\boldsymbol{s}_n, \boldsymbol{r}_n, \boldsymbol{o}_n)\big], \quad \boldsymbol{s}_{i+1} = f(\boldsymbol{o}_i), \tag{23}$$

where $f(\cdot)$ denotes the semantic grounding function that derives the next subject from the current object (e.g., entity linking). Suppose $(\boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o}) = (\boldsymbol{s}_k, \boldsymbol{r}_k, \boldsymbol{o}_k)$ is the k-th node in $\mathcal{C}$. When this fact is edited to $(\boldsymbol{s}_k, \boldsymbol{r}_k, \boldsymbol{o}'_k)$, consistency requires that all downstream nodes $\{(\boldsymbol{s}_i, \boldsymbol{r}_i, \boldsymbol{o}_i)\}_{i>k}$ be correspondingly updated, since $\boldsymbol{s}_{k+1} = f(\boldsymbol{o}_k)$ must now be re-derived from the modified object $\boldsymbol{o}'_k$:

$$\boldsymbol{s}'_{k+1} = f(\boldsymbol{o}'_k) \ \neq \ \boldsymbol{s}_{k+1} = f(\boldsymbol{o}_k). \tag{24}$$

Thus, a single fact edit at position k induces a cascade of required updates across all $n - k$ downstream nodes—a propagation that point-wise editing methods are by design unable to capture.

C Related Works

C.1 Meta-Learning Method

This paradigm aims to update the pre-edit model weights W in a certain way, depending on the knowledge to be updated.

For meta-learning, a hypernetwork is employed to learn to predict Δ, such that the post-edit model with parameters $W + \Delta$ is able to provide correct answers under a new knowledge background. The paradigm begins with Knowledge Editor (KE) (De Cao et al., 2021), which predicts Δ with a bidirectional LSTM under a latest-knowledge constraint. However, the method encounters a bottleneck when it comes to editing LLMs. Thus, Mitchell *et al.* introduce the hypernetwork MEND (Mitchell et al., 2021), for which the gradients can be decomposed to make the editing process more effective and reduce the training cost. Nevertheless, meta-learning methods can sometimes amplify **catastrophic forgetting**. To overcome this limitation, F-Learning (Ni et al., 2023) is proposed, significantly outperforming full fine-tuning and LoRA fine-tuning, which increases the expandability of the meta-learning paradigm. Towards lifelong learning, Li *et al.* introduce RLEdit (Li et al., 2025b) for continuous editing and construct the reinforcement learning paradigm in model editing.

C.2 Locate-Then-Edit Method

Returning to the motivation of model editing, it aims at utilizing fewer computational resources to filter outdated facts and inject their current counterparts, eliminate hallucinations, cut down offensive output, and so on. Nevertheless, a branch of knowledge inside the large language model is always connected to other counterparts, which is not fully revealed. Thus, this paradigm is proposed to identify where the knowledge is restored and then execute the editing. Based on the representation denoising method, Meng et al. (2022a) derive the closed-form solution $\mathbf{\Delta}^* = (\mathbf{V}_1 - \mathbf{W}\mathbf{K}_1)\mathbf{K}_1^T(\mathbf{K}_0\mathbf{K}_0^T + \mathbf{K}_1\mathbf{K}_1^T)^{-1}$ for the locate-then-edit paradigm in the mathematical expression of

$$\mathbf{\Delta}^* = \arg\min_{\mathbf{\Delta}}(||(\mathbf{W}+\mathbf{\Delta})\mathbf{K}_1 - \mathbf{V}_1||^2 + ||(\mathbf{W}+\mathbf{\Delta})\mathbf{K}_0 - \mathbf{V}_0||^2) \tag{25}$$

where $\mathbf{V}_1 = \mathbf{W}\mathbf{K}_1$ and $\mathbf{W}\mathbf{K}_0 = \mathbf{V}_0$ represent the knowledge to be updated and retained, respectively, based on the explanation of the key-value pair restoration mechanism proposed by Dai *et al.* To edit more memories, Meng et al. (2022b) derive the closed-form solution for multi-layer editing. To modify the l-th layer, the weight can be updated by the formulation $\mathbf{\Delta}^* = \mathbf{R}^l{\mathbf{K}_1^l}^T({\mathbf{K}_0^l}{\mathbf{K}_0^l}^T + {\mathbf{K}_1^l}{\mathbf{K}_1^l}^T)^{-1}$, where $\mathbf{R}^l = \frac{\mathbf{R}^{l^*}}{l^* - l + 1}$. Furthermore, for more precise editing, PMET (Li et al., 2024) reveals that MHSA (multi-head self-attention) layers work as a knowledge extractor, more efficiently localizing knowledge within transformer blocks to update specific components with a trade-off between MHSA and FFN. Due to significant flaws in current localization methods (e.g. causal tracing (Meng et al., 2022a), LoRA (Hu et al., 2022)) and for more fine-grained updating, Pan et al. (2025) propose a neuron-level localization technique, calculating the contribution score between each neuron and token, through which the top-k neurons are updated for each piece of knowledge, which also reduces the computational cost. However, post-edit models probably forget previously updated structured knowledge during the sequential editing process. To address this challenge, Jiang et al. (2025a) devise NSE to mitigate the impairment by collecting "influential neurons". By sorting neuron activations within crucial layers using original weight references and a rewinding mechanism, NSE effectively maintains the coherence of the post-edit model.

C.3 Representation Editing

To validate the existence of a concept within the representation subspace, *distributed interchange intervention* (DII) (Geiger et al., 2021) can effectively identify a subspace to localize or modify a concept via distributed alignment search (DAS). Following the annotations given by Wu et al. (2024), the DII process can be represented as

$$\mathrm{DII}(\mathbf{h}_b, \mathbf{h}_s, \mathbf{R}) = \mathbf{h}_b + \mathbf{R}^T(\mathbf{R}\mathbf{h}_s - \mathbf{R}\mathbf{h}_b) \tag{26}$$

where $\mathbf{h}_b$ and $\mathbf{h}_s$ represent hidden representations at the i-th row and the k-th column when the same model processes the inputs b and s, respectively. Matrix $\mathbf{R} \in \mathbb{R}^{r \times d}$ is a low-rank projection matrix with orthonormal rows. By analyzing and editing the representation within the hidden states, ReFT (Wu et al., 2024) implements drop-in replacements for weight-based PEFTs, inserting additional representation maps in the subspace to intervene in LM behaviors. For more precise editing, Liu *et al.* theoretically analyze the conflict between locality and generality under the ReFT paradigm and derive BaFT (Liu et al., 2025) to provide a new approach for *locality* enhancement in model editing. Building upon the *linear representation hypothesis* (Park et al., 2023), BaFT learns the base vectors of each column space of the counterpart in ReFT, since each basis vector spans a rank-one subspace. With the editing of base vectors in the subspace, BaFT performs more localized knowledge updates for more complex scenarios.

D More Detailed Experimental Settings

D.1 Alternative Formulation: Portability Reward via LLM-synthesized Data

The main paper adopts a three-term edit performance reward $\mathcal{R}_{epr} = -(\mathcal{L}_{\text{loc}} + \lambda_{\text{rel}}\mathcal{L}_{\text{rel}} + \lambda_{\text{gen}}\mathcal{L}_{\text{gen}})$ that does *not* include an explicit portability term. This appendix documents an alternative formulation in which a portability loss $\mathcal{L}_{\text{port}}$ is added to LLM-synthesized multi-hop QA pairs. We report this formulation for completeness; in our experiments, the three-term main-paper variant consistently matches or slightly outperforms the portability-augmented variant, which is why the main results use the former.

Construction of the Portability Loss. Since the training split of VLKEB provides no reasoning-oriented hints for this reward (see Appendix E), we employ an external LLM to construct the necessary associations. Concretely, given a multimodal fact $(\boldsymbol{I}^e, \boldsymbol{x}^e, \boldsymbol{y}^e)$ whose structured component $\boldsymbol{x}^e = (\boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o})$ is to be edited into $(\boldsymbol{s}', \boldsymbol{r}, \boldsymbol{o})$, we prompt the LLM to generate a chain of m auxiliary properties about the target entity $\boldsymbol{o}$. The response is a sequence of attribute–value pairs $\{(\boldsymbol{r}_1, \boldsymbol{a}_1), (\boldsymbol{r}_2, \boldsymbol{a}_2), \ldots, (\boldsymbol{r}_m, \boldsymbol{a}_m)\}$, where each $\boldsymbol{r}_i$ $(i = 1, 2, \cdots, m)$ is a plausible relation and $\boldsymbol{a}_i$ is the corresponding value. For example, if the original fact is: ("Biden", "holds position", "President of the United States")–to be edited into–("Trump", "holds position", "President of the United States"). The LLM may produce:

$$\begin{aligned}(\boldsymbol{r}_1, \boldsymbol{a}_1) &= (\text{"political party"}, \text{"Republican Party"}) \\ (\boldsymbol{r}_2, \boldsymbol{a}_2) &= (\text{"residence"}, \text{"White House"}) \\ (\boldsymbol{r}_3, \boldsymbol{a}_3) &= (\text{"spouse"}, \text{"Melania Trump"})\end{aligned} \tag{27}$$

These generated pairs are then attached to the new subject $\boldsymbol{s}'$, forming a set of auxiliary facts:

$$(\boldsymbol{s}', \boldsymbol{r}_1, \boldsymbol{a}_1),\ (\boldsymbol{s}', \boldsymbol{r}_2, \boldsymbol{a}_2),\ \ldots,\ (\boldsymbol{s}', \boldsymbol{r}_m, \boldsymbol{a}_m). \tag{28}$$

To obtain multi-hop QA pairs for portability evaluation, we chain the auxiliary attributes into a series of progressive questions. For the i-th step we ask:

```
What is the {r_i} of {s'} given that its {r_1} is {a_1}, ...,
its {r_{i-1}} is {a_{i-1}}?
```

The first question ($i = 1$) simply requests the first property without conditioning; subsequent questions require combining the previously answered attributes to infer the next. In the running example, the resulting questions are:

1. What is the political party of Trump? → expected answer: "Republican Party".
2. Given that Trump's political party is Republican, what is his residence? → expected answer: "White House".
3. Given that Trump's political party is Republican and his residence is the White House, who is his spouse? → expected answer: "Melania Trump".

By answering these progressively, a model demonstrates whether the edit $(\boldsymbol{s}', \boldsymbol{r}, \boldsymbol{o})$ has been successfully *ported* to the auxiliary facts. The portability reward is computed as the accuracy of the model's responses on such QA sequences. All attribute generation is performed with a text-only LLM, where we use GLM-5.1 (Zeng et al., 2026), thereby circumventing hallucinations that could arise from imperfect multimodal alignment. The process can be simply summarized as follows:

```
--- System Prompt ---
You will be given an English question-answer pair. You need to do your
    best to associate various attributes of the answer and generate 1 to
    4 new question-answer pairs. Note that the questions must not contain
     the original noun. Below is an example of input and output that
    generated 3 question-answer pairs:
[INPUT] Q: "Which Illinois city is shown in the picture?" A: "Laredo,
    _Texas"
(Reasoning process: "Laredo,_Texas" is a city in Illinois, is in the
    Central Standard Time (CST/CDT) zone, is the second-largest inland
    port in the U.S., and has the Lincoln-Juarez Bridge port facility)
[OUTPUT]
Q: "What is the time zone of this Illinois city?" A: "Central Standard
    Time (CST/CDT)"
Q: "What is the rank of this Illinois city, which is in the Central
    Standard Time (CST/CDT) zone, as the largest inland port in the U.S
    .?" A: "The second-largest inland port in the U.S"
Q: "What port facilities are found in this Illinois city, which is the
    second-largest inland port in the United States and lies within the
    Central Standard Time (CST/CDT) zone?" A: "The Lincoln-Juarez Bridge"

All Q&A pairs are separated by line breaks. You should only output the Q&
    A pair without the Reasoning process
```

We prepared the dataset based on VLKEB and E-VQA, and we are ready to release it. The construction pipeline is as shown in Figure 5.

E Benchmarks and Metrics for Multimodal Knowledge Editing

E.1 MMEdit

Cheng et al. (2023) first explores the area of editing multimodal knowledge, decomposed into two sub-tasks: Editing VQA (*E-VQA*) and Editing Image Captioning (*E-IC*). *E-VQA* is constructed with curated images from VQAv2 (Goyal et al., 2017) and captions from COCO (Chen et al., 2015), where ChatGLM is utilized for the generation of similar queries. *E-IC* is constructed upon Visual Question Answering (VQA) (Antol et al., 2015) and Image Captioning (Herdade et al., 2019) with manually created prompt templates. The metrics **Reliability**, **Locality** (text locality $\mathcal{M}_{loc}^{Text}$ and image locality $\mathcal{M}_{loc}^{Img}$), and **Generality** (text generality $\mathcal{M}_{gen}^{Text}$ and image generality $\mathcal{M}_{gen}^{Img}$) are proposed for evaluation. As the edit target is denoted as $(i_e, x_e, y_e) \in \mathcal{D}_{\text{edit}}$, **Reliability** is calculated by

$$\mathcal{M}_{rel} = \mathbb{E}_{(i_e,x_e,y_e)\sim\mathcal{D}_{edit}} \left[\mathbf{1}_{f(i_e,x_e;\theta_e(i_e,x_e,y_e))=y_e}\right], \tag{29}$$

where $\theta_e(i_e, x_e, y_e)$ represents the edited weights. Reliability directly assesses the result of the edit. To evaluate how editing influences unrelated knowledge restoration, **Locality** is computed for both

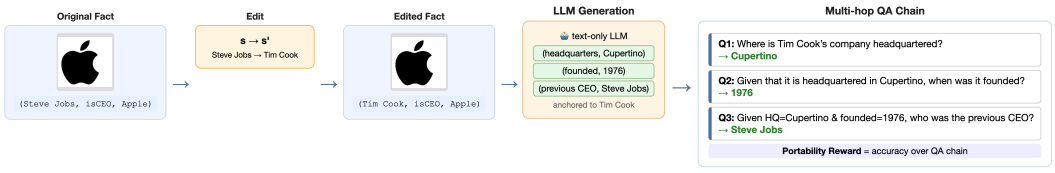


Figure 5: Portability auxiliary data synthesis pipeline.

textual and modal locality, given by

$$\mathcal{M}_{loc}^{Textual} = \mathbb{E}_{(i_e,x_e,y_e)\sim\mathcal{D}_{\text{edit}},\ (i_e,x,y)\sim\mathcal{D}_{\text{loc-t}}}\left[\mathbf{1}_{f(i_e,x;\theta_e(i_e,x_e,y_e))=f(i_e,x;\theta)}\right],$$
$$\mathcal{M}_{loc}^{Modal} = \mathbb{E}_{(i_e,x_e,y_e)\sim\mathcal{D}_{\text{edit}},\ (i_v,x_e,y_e)\sim\mathcal{D}_{\text{loc-v}}}\left[\mathbf{1}_{f(i_v,x_e;\theta_e(i_e,x_e,y_e))=f(i_v,x_e;\theta)}\right], \tag{30}$$

where $(i_e, x, y) \sim \mathcal{D}_{\text{loc-t}}$ denotes the multimodal pair with an unrelated textual prompt, while $(i_v, x_e, y_e) \sim \mathcal{D}_{\text{loc-v}}$ denotes the pair with an unrelated image. We simply denote these two as $\mathcal{M}_{loc}^{T}$ (T-Loc) and $\mathcal{M}_{loc}^{M}$ (M-Loc), respectively. After editing, the post-edit model should recognize rephrased or synonymous information, which is evaluated by **Generality**, on both textual and modal modalities:

$$\mathcal{M}_{\text{gen}}^{Textual} = \mathbb{E}_{(i_e,x_e,y_e)\sim\mathcal{D}_{\text{edit}},\ (x_r)\sim\mathcal{N}(x_e)}\left[\mathbf{1}_{f(i_e,x_r;\theta_e(i_e,x_e,y_e))=f(i_e,x_e;\theta)}\right], \tag{31}$$
$$\mathcal{M}_{\text{gen}}^{Modal} = \mathbb{E}_{(i_e,x_e,y_e)\sim\mathcal{D}_{\text{edit}},\ (i_r)\sim\mathcal{N}(i_e)}\left[\mathbf{1}_{f(i_r,x_e;\theta_e(i_e,x_e,y_e))=f(i_e,x_e;\theta)}\right],$$

where i_r, x_r refer to the rephrased images and textual prompts, $\mathcal{N}(x_e)$ and $\mathcal{N}(i_e)$ denote the corresponding in-domain related samples.

E.2 VLKEB

Compared to MMEdit (Cheng et al., 2023), VLKEB (Huang et al., 2024) provides more manually checked real images with more filtered samples. To evaluate the ability of the post-edit model to infer with newly injected multimodal knowledge across multi-hop reasoning, a harder metric *Portability* is proposed and calculated via

$$\mathcal{M}_{portability} \tag{32}$$
$$=\mathbb{E}_{(i_e,x_e,y_e)\sim\mathcal{D}_{edit},\ (x_p,y_p)\sim\mathcal{P}(i_e,x_e,y_e)}\left[\mathbf{1}_{f(i_e,x_p;\theta_e(i_e,x_e,y_e))=y_p}\right]$$

where x_p, y_p represent the related multi-hop knowledge associated with the multimodal knowledge (i_e, x_e, y_e). The multi-hop reasoning chain (1–4 hops) is denoted as $\mathcal{P}(i_e, x_e, y_e)$. The image and textual prompt pairs in VLKEB are collected from MMKG (Liu et al., 2019) in the form of triplets $(\boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o})$ with the multiplicity of the entity reduced through CLIP-extracted features (Radford et al., 2021).

F Baseline Method for LVLM Knowledge Editing

Currently, knowledge editing methods for large language models can be divided into two categories (Yao et al., 2023), parameter-preserved and parameter-modified, according to whether the parameters are directly edited. Nevertheless, many newly emerging data mining–based methods have appeared that do not fall into this categorization.

SERAC. Mitchell et al. (2022) implemented semi-parametric editing with a retrieval-augmented counterfactual model to address the insufficient accuracy of the model structure, enhancing the expressive ability. Taking notice of the impoverishment of the gradient signal, SERAC leverages an explicit cache for user queries. With a small auxiliary classifier and a counterfactual model trained, the cache directly stores modification examples without any alteration of the original model. When a post-edit case is received, the scope classifier determines whether it falls within the cache's coverage. Some seminal and fundamental works include (Santoro et al., 2016; Lopez-Paz & Ranzato, 2017; Kassner et al., 2021).

MEND. Mitchell et al. (2021) introduce a fast model editing approach via low-rank decomposition of the gradient to make the parameterization of large language model transformations tractable. By training an auxiliary lightweight network, MEND predicts the rank-one vector from the weight gradients to recover the model parameter shifts, making the knowledge editing process more parameter-efficient and scalable.

RECIPE. Chen et al. (2024) propose a retrieval-augmented framework for efficient lifelong LLM editing. Each edit is encoded into a continuous prompt that is prefixed to an embedding, guiding the

LLM to follow the desired update. Leveraging a knowledge sentinel (KS) serving as a trainable intermediary, RECIPE determines whether the retrieval repository contains query-relevant knowledge, avoiding fixed thresholds.

PokeMQA. Gu et al. (2024) decouple questions for knowledge-editing, aware of conflict detection for robust multi-hop question answering. They introduce a programmable scope detector, which first performs efficient pre-detection and then a precise conflict-disambiguation stage to determine whether a sub-question falls within the scope of any edit, sending a factual conflict signal only when necessary. A knowledge prompt generator retrieves external entity information to assist in the correct decomposition of the leading sub-question.

ReasonKE. An end-to-end reasoning-chain-based framework for multi-hop knowledge editing that explicitly filters out irrelevant distractor facts is proposed by Wu et al. (2025). A pretrained LLM is fine-tuned to generate a structured reasoning process including acknowledging the update, determining relevance, identifying pertinent facts, and producing a final answer. The single-pass pipeline eliminates costly iterative retrieval and sets a new state-of-the-art for reliable knowledge updates under distraction.

RLEdit. Li et al. (2025b) first formulate lifelong model editing as a reinforcement learning problem. The hypernetwork is treated as an agent that takes the current LLM state and new knowledge as input, generating parameter updates (actions) to maximize a reward function. The reward comprises an editing loss, a memory-backtracking term, and a regularization term to preserve general capabilities. An offline policy update over the full knowledge sequence enables the hypernetwork to capture long-range dependencies.

LEMoE. Wang & Li (2024a) propose an advanced Mixture-of-Experts adapter designed for lifelong model editing, motivated by analyses of catastrophic forgetting, routing inconsistency, and order sensitivity. As a new batch of edits arrives, a fresh FFN expert is inserted and trained while all previous experts remain frozen, thereby preventing interference with earlier knowledge. A novel KV anchor routing assigns a key vector to each expert and uses instance-level embeddings as values, aligning training and inference routing to boost consistency. A clustering-based editing order planning further groups semantically similar edits into the same batch, leading to superior lifelong and batch editing performance.

LiveEdit. Chen et al. (2025a) bridge the gap between lifelong LLM editing and vision-language models (LVLMs) by training an editing expert generator to produce low-rank experts for each edit batch. All experts are stored in an external repository without modifying the frozen LVLM. Leveraging both soft and hard routing mechanisms, irrelevant experts are filtered while experts with more visual semantics are chosen for the inference stage. LiveEdit achieves strong reliability, generality, and locality on lifelong LVLM editing benchmarks.

G More Experiments for Main Results

We provide the evaluation on other metrics and a different LVLM backbone MiniGPT4-vicuna-7B as shown in Table 5.

H Ablation Study: The Impact of MDP Sub-Sequence Lengths

We provide the detailed experimental results as we vary (δ_L, Δ_L) within $[16, 100] \times [25, 400] \bigcap \mathbf{N}$. Considering the training cost, we randomly select 15 points within the area. For a fair comparison, we train each of the following pairs for 5 hours, with the baseline vision language model BLIP2-OPT (2.7B). We evaluate $\#100$-edit for comparison. We visualize *Effc.* (average of editing efficiency), *Port.* (average of multi-hop reasoning, i.e., portability), and minimum/maximum subsequence lengths (δ_L/Δ_L) in Figure 6. We provide the average of editing efficiency and each hop accuracy to show the impact of the sub-sequence in the Markov decision process. Our analysis can be divided into two parts. First, if $\delta_L = \Delta_L$, MoE-GRPO degrades to normal GRPO, so we should set a large enough batch size ($= \delta_L = \Delta_L$) to ensure the performance. Second, if $\delta_L \neq \Delta_L$, the performance of MoE-GRPO is expected to be stable only in the red-shaded area in Figure 6. The detailed data are shown in Table 6.

Baseline	#Edit	Editors	Accuracy					Portability (Multihop Reasoning)						
			T-Loc	M-Loc	T-Gen	M-Gen	Rel	2-*hop*		3-*hop*		4-*hop*		*Avg*[2-4 hop]
								Orig	Reph	Orig	Reph	Orig	Reph	
BLIP2-OPT(2.7B)	1	FT	47.60	40.35	**99.76**	**99.86**	**100.00**	27.68	27.78	27.82	27.98	30.63	30.59	28.73
		MEND	1.14	0.75	38.34	38.79	38.84	0.96	0.91	0.25	0.26	0.37	0.28	0.51
		SERAC	100.00	100.00	38.33	36.09	36.13	25.96	25.65	27.90	27.68	32.17	31.99	28.56
		RECIPE	100.00	100.00	42.55	39.59	39.59	25.54	25.51	26.61	25.87	31.55	31.12	27.70
		RLEdit	100.00	100.00	38.33	36.09	36.13	25.96	25.65	27.90	27.68	32.17	31.99	28.56
		PokeMQA	100.00	77.07	60.15	65.56	65.78	26.84	26.63	27.94	27.70	32.17	31.99	28.88
		ReasonKE	78.52	49.97	76.14	99.03	98.89	36.19	**35.85**	34.73	**34.82**	35.65	**36.00**	**35.55**
		LEMoE	100.00	100.00	38.33	36.09	36.13	25.95	25.65	27.90	27.67	32.16	31.98	28.55
		LiveEdit	100.00	100.00	96.81	95.23	96.96	26.85	26.95	27.52	28.63	29.95	32.07	28.66
		MoE-GRPO	**100.00**	**100.00**	85.91	79.52	88.09	**36.61**	35.11	**35.64**	33.33	**36.49**	35.10	35.38 (-0.48%)
	100	FT	9.21	10.07	56.06	58.32	58.57	11.13	11.42	11.71	11.72	12.54	12.59	11.85
		MEND	3.26	2.49	1.12	1.13	1.15	0.96	1.01	1.06	1.03	1.20	1.22	1.08
		SERAC	100.00	100.00	38.42	36.13	36.12	25.92	25.61	27.97	27.75	32.20	32.01	28.57
		RECIPE	99.29	95.19	47.85	45.60	45.60	25.98	25.92	27.25	27.14	32.72	32.64	28.61
		RLEdit	100.00	100.00	38.42	36.13	36.15	25.92	25.61	27.97	27.75	32.20	32.01	28.57
		PokeMQA	99.98	78.48	48.29	58.46	58.57	27.07	26.78	28.21	27.86	32.22	32.06	29.03
		ReasonKE	76.89	48.78	54.88	72.53	72.36	31.83	31.57	31.61	31.54	32.07	31.94	31.76
		LEMoE	45.68	48.86	46.04	45.79	45.69	31.54	31.74	31.46	31.08	33.14	33.02	31.99
		LiveEdit	100.00	100.00	**94.99**	**89.36**	**96.85**	27.00	27.25	27.53	28.65	30.19	31.98	28.76
		MoE-GRPO	**100.00**	**100.00**	82.38	76.42	83.29	**42.67**	**39.07**	**42.22**	**38.71**	**44.63**	**43.16**	**41.74** 30.48%
	1000	FT	3.74	8.86	44.51	47.32	48.22	7.69	7.68	9.57	9.47	9.87	9.77	9.01
		MEND	1.49	1.06	0.99	1.05	0.01	1.23	1.22	0.47	0.53	6.24	0.64	1.72
		SERAC	100.00	100.00	38.99	36.05	36.20	27.34	27.22	26.18	27.55	32.87	32.49	28.94
		RECIPE	97.65	90.84	47.85	44.95	44.95	30.04	29.86	31.19	30.97	35.61	35.45	32.19
		RLEdit	100.00	100.00	38.99	36.05	36.20	26.18	25.77	27.97	27.55	32.87	32.49	28.81
		PokeMQA	99.94	60.22	42.51	50.45	50.56	27.72	27.33	27.98	27.71	32.94	32.63	29.39
		ReasonKE	77.53	46.17	47.73	61.58	61.53	31.09	31.50	31.82	31.60	30.04	30.53	31.10
		LEMoE	47.97	54.51	38.71	36.02	35.74	30.75	30.84	32.77	31.45	35.59	35.77	32.60
		LiveEdit	100.00	100.00	**94.88**	**80.15**	**96.60**	27.56	27.15	28.27	29.56	30.44	31.96	29.16
		MoE-GRPO	**100.00**	**100.00**	74.90	72.54	76.32	**40.45**	**39.00**	**40.47**	**38.84**	**42.65**	**41.56**	**40.50** 24.23%
LLaVA-v1.5 (7B)	100	FT	11.08	9.93	64.67	66.86	68.13	5.76	5.86	6.17	6.07	7.27	7.31	6.41
		MEND	2.31	0.30	0.27	0.24	0.24	0.07	0.09	0.24	0.26	0.46	0.49	0.27
		SERAC	100.00	100.00	49.64	47.27	47.08	40.42	40.40	40.41	40.31	42.81	42.59	41.16
		RECIPE	99.58	99.96	51.35	49.37	49.37	40.79	40.34	41.72	41.69	45.06	44.53	42.36
		RLEdit	0.03	0.08	0.06	0.04	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		PokeMQA	94.45	81.43	58.24	72.73	72.90	39.93	39.96	40.13	40.12	42.22	41.99	40.73
		ReasonKE	64.66	62.41	59.17	66.63	66.78	43.28	43.28	39.25	39.54	39.10	39.34	40.63
		LEMoE	59.63	67.92	50.51	48.10	49.11	37.34	37.34	36.83	36.51	39.06	38.68	36.63
		LiveEdit	100.00	100.00	**83.94**	**76.32**	**84.43**	41.02	41.94	39.52	40.96	42.35	42.16	41.32
		MoE-GRPO	**100.00**	**100.00**	81.46	74.47	82.28	**44.27**	**43.82**	**45.06**	**44.77**	**47.94**	**48.76**	**45.77** 7.64%
	1000	FT	6.91	7.51	42.67	44.38	45.00	6.22	6.20	9.14	9.10	9.87	9.80	8.39
		MEND	0.13	0.39	0.37	0.37	0.37	0.08	0.08	0.00	0.00	0.01	0.01	0.03
		SERAC	100.00	100.00	49.09	46.51	46.50	40.61	40.57	40.29	40.18	43.19	43.15	41.33
		RECIPE	99.24	99.70	51.21	48.72	48.72	40.90	40.77	42.25	42.13	45.21	45.04	42.72
		RLEdit	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		PokeMQA	86.50	67.87	51.35	62.84	63.13	40.12	39.79	40.26	40.23	41.17	41.24	40.47
		ReasonKE	64.27	61.62	53.22	57.47	58.08	42.74	42.83	39.59	39.76	38.79	38.66	40.39
		LEMoE	48.68	40.17	42.32	41.02	37.14	30.63	31.02	32.16	29.77	34.21	34.60	32.06
		LiveEdit	100.00	100.00	**74.51**	**67.72**	**81.03**	44.88	43.91	43.31	42.86	45.39	45.57	44.32
		MoE-GRPO	**100.00**	**100.00**	72.60	65.46	78.76	**45.36**	**45.01**	**45.41**	**45.84**	**46.86**	**47.51**	**46.00** 3.79%

Table 4: General editing performance and portability (multi-hop reasoning) evaluation. We report results on original (*Orig*) and rephrased (*Reph*) multimodal knowledge pairs. **MoE-GRPO** achieves the best performance on Portability (multi-hop reasoning ability) with strong robustness under rephrasing. The best results are in **bold** and the second is underlined.

I More Hidden Representation Visualization and Analysis

I.1 Approach for Visualization and Analysis

Hidden Representation Analysis via t-SNE. We aim to investigate how the MoE-based expert correction reshapes the LVLM's internal representations during post-edit inference. Following Fang et al. (2024), we conduct the following steps for the MoE-based editor. ***a)*** We randomly select 1000 prompts paired with their corresponding images from the evaluation set and extract their hidden representations $\boldsymbol{h}_{\text{in}}$ at the edit layer of the *unedited* LVLM by temporarily disabling the expert correction hook. ***b)*** Subsequently, we perform $\#100$ edits on the LVLM (baseline LiveEdit & MoE-GRPO) and recompute the hidden representations for the same input pairs. For each input, we collect both $\boldsymbol{h}_{\text{in}}$ and $\boldsymbol{h}_{\text{out}}$ computed via Eq. 3. ***c)*** We apply t-SNE (Van der Maaten & Hinton, 2008) to reduce the collected representations to 2D and visualize them as shown in Figure 4.

Expert Attention Map Visualization. We aim to analyze the routing behavior of the MoE-based editor during lifelong editing, with the following steps. ***a)*** We perform $\#1000$ edits on the LVLM and uniformly sample M checkpoints along the editing trajectory. At each checkpoint, we randomly select a multi-hop portability query (i.e., a visual question requiring compositional reasoning over previously edited facts) along with its associated image. ***b)*** For each sampled instance, we extract the ViT CLS-to-patch attention map from the last Transformer layer via a forward hook, serving as a spatial heatmap indicating the **key visual area** attended by the vision encoder. Simultaneously, we obtain the Q-Former cross-attention weights that map the 32 non-spatial query tokens back to the ViT spatial patch grid. ***c)*** For each expert k in the MoE pool, we compute its per-token residual

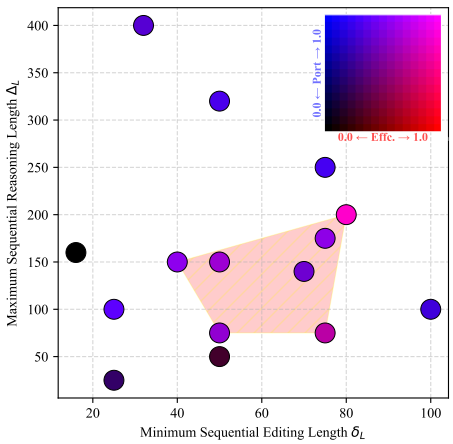


Figure 6: Visualization of the impact of the subsequence lengths in MDP.

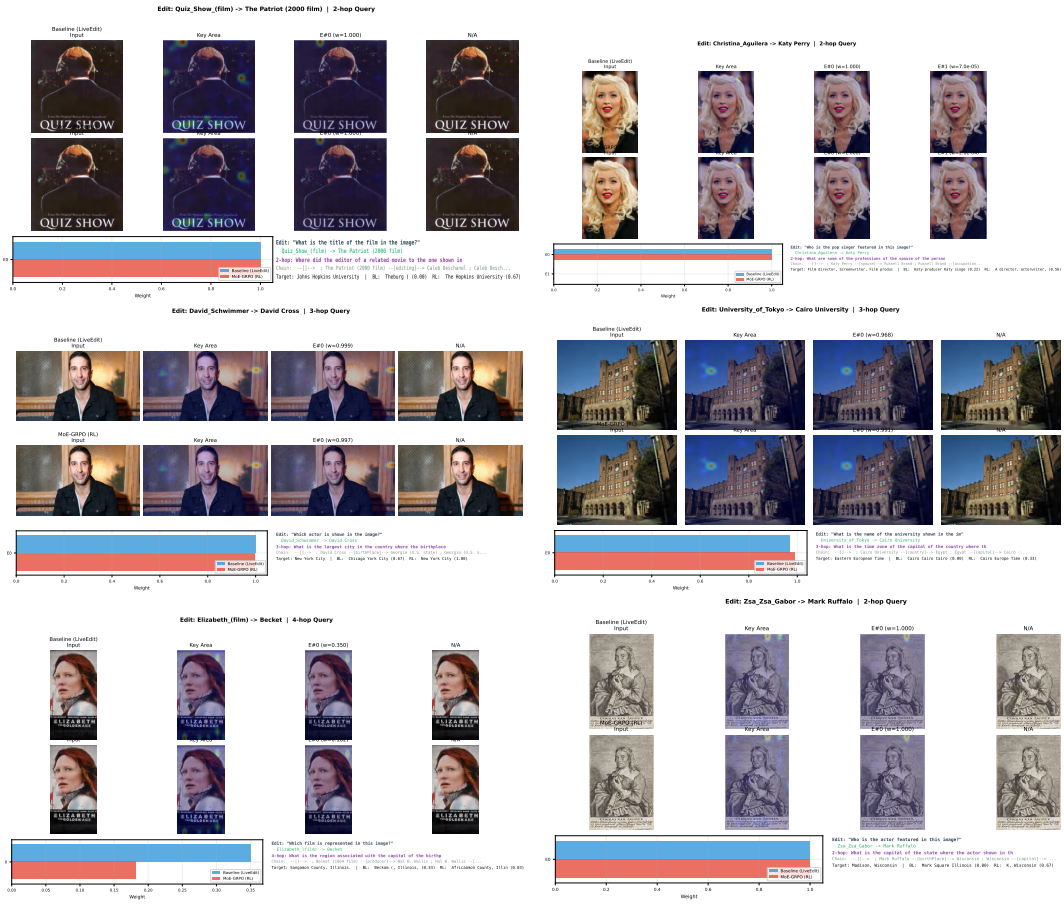


Figure 7: Instance Analysis.

Table 5: General editing efficiency and portability (multi-hop reasoning) evaluation. We report results on original (*Orig*) and rephrased (*Reph*) multimodal knowledge pairs. Settings are the same as in Table 4. The best results are in **bold** and the second is underlined.

Baseline	#Edit	Editors	Accuracy					Portability (Multihop Reasoning)						
			T-Loc	M-Loc	T-Gen	M-Gen	Rel	2-*hop*		3-*hop*		4-*hop*		*Avg*[2-4 hop]
								Orig	Reph	Orig	Reph	Orig	Reph	
MiniGPT4 -vicuna (7B)	1	FT	58.69	38.64	99.84	99.95	100.00	21.27	21.93	20.30	20.67	21.27	21.84	21.21
		MEND	0.98	0.92	9.29	9.37	9.51	0.45	0.44	0.21	0.24	0.22	0.28	0.31
		SERAC	100.00	100.00	44.38	41.57	41.81	38.81	37.06	41.25	41.05	39.81	39.44	39.57
		RECIPE	100.00	100.00	50.14	48.01	47.97	39.01	38.86	39.83	39.74	40.12	40.10	39.61
		RLEdit	9.11	4.33	2.95	2.99	3.04	0.21	0.21	0.03	0.03	0.00	0.00	0.08
		PokeMQA	100.00	100.0	44.38	41.89	42.13	38.02	38.18	40.51	40.60	42.59	43.04	40.49
		ReasonKE	72.35	66.05	87.05	83.37	82.96	40.81	40.94	40.43	40.85	41.22	42.45	41.12
		LEMoE	100.00	100.00	44.38	41.57	41.81	37.12	37.18	39.54	40.38	41.45	41.06	39.46
		LiveEdit	100.00	100.00	88.04	72.12	86.88	37.66	38.19	39.01	39.96	41.63	41.75	39.69
		MoE-GRPO	100.00	100.00	85.54	83.11	74.21	39.82	40.33	40.61	40.76	46.08	44.58	41.94
	100	FT	21.65	9.68	57.32	57.07	57.99	5.50	5.67	6.28	6.15	6.96	7.10	6.28
		MEND	1.73	0.65	0.00	0.03	0.00	0.00	0.00	0.00	0.02	0.00	0.00	0.00
		SERAC	100.00	100.00	44.45	41.65	41.88	39.22	39.38	41.07	41.20	44.01	43.87	41.46
		RECIPE	99.82	100.00	50.48	48.15	48.02	38.66	38.59	39.96	39.87	40.91	40.79	39.79
		RLEdit	13.08	6.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		PokeMQA	100.00	100.00	44.45	42.03	42.19	38.02	38.18	40.62	40.71	42.61	43.07	40.53
		ReasonKE	71.56	63.31	61.00	68.77	68.52	42.56	42.49	42.65	43.04	44.22	43.24	43.25
		LEMoE	64.80	53.68	44.13	42.09	42.35	29.59	29.43	28.77	28.99	30.63	30.37	29.63
		LiveEdit	100.00	100.00	87.83	72.42	86.87	37.73	38.24	39.12	40.10	41.66	41.91	39.79
		MoE-GRPO	100.00	100.00	84.57	69.15	84.30	40.53	40.81	43.63	42.87	44.83	44.63	42.88
	1000	FT	33.33	33.33	7.47	8.62	36.14	4.78	4.64	7.88	7.74	8.26	8.08	6.90
		MEND	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
		SERAC	100.00	100.00	40.28	43.74	40.63	40.02	39.64	40.81	40.27	41.33	42.76	40.81
		RECIPE	98.37	96.82	50.54	48.60	47.03	38.92	38.33	40.17	40.05	41.30	41.19	39.99
		RLEdit	1.81	1.84	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		PokeMQA	100.00	100.00	43.74	40.69	40.83	38.26	38.52	40.73	40.88	42.71	43.12	40.70
		ReasonKE	72.35	61.46	54.84	59.53	58.98	42.64	42.69	42.03	42.47	41.00	41.32	42.02
		LEMoE	52.72	49.04	46.91	44.23	36.78	24.72	25.83	26.87	26.15	24.43	23.79	25.30
		LiveEdit	100.00	100.00	86.31	71.21	86.57	38.35	38.66	39.52	40.08	41.75	42.53	40.14
		MoE-GRPO	100.00	99.73	83.99	70.62	80.95	42.37	42.83	41.24	42.26	44.35	44.74	42.93

Table 6: Detailed data as varying (δ_L, Δ_L) pairs.

Min Len. δ_L	Max Len. Δ_L	Accuracy Avg	Portability (Multihop Reasoning) 1-*hop* Orig	1-*hop* Reph	2-*hop* Orig	2-*hop* Reph	3-*hop* Orig	3-*hop* Reph	4-*hop* Orig	4-*hop* Reph	Avg[1-4]	Ratio Δ_L/δ_L
16	160	62.71	25.37	27.15	27.91	28.66	30.11	30.91	31.01	32.32	29.18	10.00
25	25	66.24	33.83	33.32	34.73	34.01	35.84	34.85	38.02	37.19	35.22	1.00
25	100	69.11	43.58	41.61	43.43	42.52	46.30	44.10	46.33	44.78	44.08	4.00
32	400	68.62	39.85	38.24	41.34	39.09	43.19	41.16	45.28	44.03	41.52	12.50
40	150	72.45	42.59	40.44	43.38	41.41	44.43	43.47	45.10	43.85	43.08	3.75
50	50	67.24	30.47	30.40	30.51	30.04	32.32	31.55	34.16	33.96	31.68	1.00
50	75	72.66	41.99	39.38	40.91	38.30	42.60	41.12	44.33	42.88	41.44	1.50
50	150	73.45	40.36	39.21	41.04	40.22	42.15	41.27	45.67	42.64	41.57	10.00
50	320	67.80	41.49	39.38	41.98	40.68	45.21	42.82	46.94	45.20	42.96	6.40
70	140	70.24	39.67	39.01	40.35	40.39	42.76	42.11	43.60	43.17	41.38	2.00
75	75	75.48	39.51	37.26	37.91	36.26	40.78	39.17	40.21	39.66	38.85	1.00
75	175	72.46	41.24	40.53	43.02	42.68	41.02	39.94	45.21	43.23	42.11	2.33
75	250	67.43	40.96	39.10	42.89	40.91	45.23	43.54	47.81	46.02	43.31	3.33
80	200	80.20	41.11	39.08	41.54	39.82	41.67	40.81	42.63	41.60	41.03	2.50
100	100	67.03	40.68	39.60	40.78	39.01	43.96	42.29	45.51	44.09	41.99	1.00

at the edit layer as $\mathbf{r}_k = \mathrm{ReLU}(\hat{\mathbf{h}}_{\text{in}}\mathbf{C}_k)\mathbf{R}_k$, where $\hat{\mathbf{h}}_{\text{in}}$ is the normalized input hidden state, and $\mathbf{C}_k, \mathbf{R}_k$ are the expert's contribution and residual matrices, respectively. We then project the per-token magnitude $\|\mathbf{r}_k[t]\|$ onto the spatial grid via the cross-attention weights:

$$\mathbf{s}_k[p] = \sum_t \|\mathbf{r}_k[t]\| \cdot \mathbf{A}_{\text{cross}}[t, p], \tag{33}$$

yielding a per-expert spatial attention map that reveals which visual region each expert focuses on. ***d)*** Finally, we overlay the top-2 routed experts' spatial maps onto the input image at each checkpoint,

comparing the baseline (LiveEdit) with MoE-GRPO. Figure 4 presents the routing weights and spatial attention patterns, illustrating how distinct experts attend to different semantic regions as editing progresses.

More Results. Despite the previously shown visualization and analysis on the main pages, we provide more instance analyses below.

I.2 Instance Cases

We show more instance analyses for multi-hop reasoning in Figure 7. The instance analyses are conducted on BLIP2-OPT (2.7B), and results are all with 100 edits.

J THEORETICAL ANALYSIS OF THE INVERTED UPPER-BIASED CLIPPING

Eq. 8 replaces the standard $\min$-based PPO/GRPO clipping with a $\max$-based inverted upper-biased surrogate. This appendix formalizes the two objectives, delineates the regimes where they differ, and connects the inversion to the structural properties of MoE-based lifelong multimodal knowledge editing.

J.1 SETUP AND NOTATION

Let $r \triangleq r_\theta^{(i,t)} = \pi_\theta(\boldsymbol{a}_t^{(i)} \mid \boldsymbol{s}_t)/\pi_{\theta_{\text{old}}}(\boldsymbol{a}_t^{(i)} \mid \boldsymbol{s}_t)$ denote the importance ratio, $\hat{A} \triangleq \mathcal{A}_t^{(i)}$ the group-relative advantage, and ϵ the clipping radius. We write $\bar{r} \triangleq \text{clip}(r, 1-\epsilon, 1+\epsilon)$. The PPO/GRPO and MoE-GRPO per-token objectives are, respectively,

$$\mathcal{J}_{\min}(r, \hat{A}) = \min\big(r\hat{A},\ \bar{r}\hat{A}\big), \qquad \mathcal{J}_{\max}(r, \hat{A}) = \max\big(r\hat{A},\ \bar{r}\hat{A}\big). \tag{34}$$

J.2 CLOSED-FORM COMPARISON

A case analysis on the sign of $\hat{A}$ and the position of r relative to $[1-\epsilon, 1+\epsilon]$ yields the following.

Standard $\min$-based objective (PPO/GRPO trust region):

$$\mathcal{J}_{\min}(r, \hat{A}) = \begin{cases} r\hat{A}, & r \in [1-\epsilon, 1+\epsilon], \\ (1+\epsilon)\hat{A}, & \hat{A} > 0 \text{ and } r > 1+\epsilon, \\ (1-\epsilon)\hat{A}, & \hat{A} < 0 \text{ and } r < 1-\epsilon, \\ r\hat{A}, & \hat{A} > 0 \text{ and } r < 1-\epsilon, \\ r\hat{A}, & \hat{A} < 0 \text{ and } r > 1+\epsilon. \end{cases} \tag{35}$$

The $\min$ operator symmetrically caps the magnitude of policy updates *in the direction that would otherwise increase the surrogate*: positive-advantage updates are capped at $(1+\epsilon)\hat{A}$ when $r > 1+\epsilon$, and negative-advantage updates are capped at $(1-\epsilon)\hat{A}$ when $r < 1-\epsilon$. In the opposite direction (where the advantage and the ratio disagree), the unclipped $r\hat{A}$ is smaller and is therefore retained, fully penalizing unfavorable moves. This is the canonical PPO trust region.

MoE-GRPO $\max$-based objective (inverted upper-biased clipping):

$$\mathcal{J}_{\max}(r, \hat{A}) = \begin{cases} r\hat{A}, & r \in [1-\epsilon, 1+\epsilon], \\ r\hat{A}, & \hat{A} > 0 \text{ and } r > 1+\epsilon, \\ r\hat{A}, & \hat{A} < 0 \text{ and } r < 1-\epsilon, \\ (1-\epsilon)\hat{A}, & \hat{A} > 0 \text{ and } r < 1-\epsilon, \\ (1+\epsilon)\hat{A}, & \hat{A} < 0 \text{ and } r > 1+\epsilon. \end{cases} \tag{36}$$

The $\max$ operator preserves the unbounded $r\hat{A}$ term whenever the advantage and the ratio point in the same direction (positive advantage with $r > 1+\epsilon$, negative advantage with $r < 1-\epsilon$), and only invokes the clipped term when the advantage and the ratio disagree. Importantly, $\mathcal{J}_{\max} \geq \mathcal{J}_{\min}$ pointwise, with equality on $r \in [1-\epsilon, 1+\epsilon]$.

J.3 WHY THE INVERSION IS NECESSARY UNDER MKE

The standard $\min$-based trust region is designed for the standard RLHF/RLAIF setting, where the policy π_θ is initialized from π_{ref} (a supervised-finetuned LLM) and the entire parameter set is updated end-to-end. In that regime, large ratio excursions signal unstable policy drift and should be capped.

The MoE-based lifelong multimodal knowledge editing setting differs along three structurally relevant axes:

1. **Frozen backbone with a narrow gradient interface.** Only the expert parameters $\{\mathbf{C}_{\boldsymbol{e}_k}, \mathbf{R}_{\boldsymbol{e}_k}\}$ and the policy parameters (ψ, ϕ) receive gradients; the LVLM backbone $\mathcal{W}$ is frozen. Gradients reach the policy through a single residual injection at the edit layer l_e, so

the effective update magnitude on π_θ is mechanically attenuated relative to a full-backbone update.

2. **Reference policy already solves the task.** π_{θ_0} is initialized from LiveEdit, which is already a strong editor with high reliability/locality/generality. The RL stage is a *refinement* towards multi-hop reasoning, not a behavior-shaping problem from a weak reference.
3. **Stochasticity is supplied externally.** Action diversity comes from the SVD-based perturbation of expert parameters, not from intrinsic policy stochasticity. Ratio excursions therefore reflect the structured perturbation, not unbounded policy movement.

Under these conditions, the systematic underestimation of policy gradients (point 1) means that the $\min$-based clipping, which is already conservative in the standard setting, becomes overly restrictive: positive-advantage updates that would push the editor toward better multi-hop reasoning are silently capped at $(1+\epsilon)\hat{A}$ even when the underlying perturbation that produced them is well within the principal subspace. The $\max$-based objective restores the gradient magnitude by allowing positive-advantage updates to scale with r.

J.4 Stability Safeguards

The inversion does *not* remove all stability controls. Three mechanisms jointly prevent the unconstrained positive-advantage branch from diverging:

1. **KL regularization to π_{ref}.** The term $-\beta\hat{\mathbb{D}}_{KL}[\pi_\theta\|\pi_{\text{ref}}]$ in Eq. 8 penalizes drift from the LiveEdit-initialized reference, providing a soft trust region at the distributional level.
2. **Asymmetric gradient normalization.** Only the router parameters ϕ are norm-clipped (Eq. 9); the expert generator ψ receives unclipped gradients. This contains the discrete-routing instability that the $\max(\cdot)$ operator could otherwise amplify, as confirmed by the $\|\nabla_{\varnothing,\phi}\mathcal{J}\|_2$ row of Table 3.
3. **SVD-based principal-subspace confinement.** The perturbation that generates r itself is confined to the leading singular directions of $(\mathbf{C},\mathbf{R})$, bounding the maximum excursion in the parameter space.

The $\max$-based surrogate is therefore wrapped by three orthogonal regularizers; removing any single one degrades performance, as shown in the ablation (Table 3, rows that ablate each reward term and each gradient configuration).

J.5 Empirical Behavior of $\max$ vs. $\min$

The $\min(\cdot)$ row of Table 3 isolates the effect of the clipping operator: under identical rewards, SVD-based perturbation, KL coefficient, and gradient normalization, replacing $\max(\cdot)$ with the standard $\min(\cdot)$ drops the average multi-hop portability from 41.74% to 33.31% (a 20.2% relative degradation) at #Edit$=100$ on BLIP2-OPT. The single-hop editing axes collapse even more sharply (T-Gen from 82.38% to 63.34%, M-Gen from 76.42% to 55.37%). This is the expected failure mode predicted by the analysis above: with the $\min$-based trust region, the systematically under-amplified gradients from the frozen-backbone interface are further capped, and the policy cannot move far enough in the direction of multi-hop-improving updates. The $\max$-based inversion is therefore not a heuristic departure from PPO but a setting-specific correction that, jointly with the three regularizers above, recovers a well-behaved optimization dynamics on top of a frozen LVLM.

K Why KL-Regularization Cannot Replace SVD-Based Perturbation

The MoE-GRPO objective in Eq. 8 carries two forms of regularization that both touch the policy's proximity to the LiveEdit-initialized reference: a KL-divergence term with coefficient β acting on the output distribution, and the SVD-based perturbation acting on the weight trajectory of the expert parameters. This appendix clarifies that the two operate at different levels of the policy and are not substitutable: tightening β cannot recover the structural properties that SVD-based perturbation provides.

K.1 Functional vs. Structural Regularization

The KL term acts on the *output distribution* of the policy: $\hat{\mathbb{D}}_{KL}[\pi_\theta(\cdot \mid \boldsymbol{s}_t) \| \pi_{\text{ref}}(\cdot \mid \boldsymbol{s}_t)]$ penalizes divergence between the actions emitted by the current policy and by the LiveEdit-initialized reference, regardless of which weight configuration produced them. Weight configurations θ and θ' that differ substantially in ℓ_2-norm can still induce identical conditional distributions over the discrete expert-routing and continuous expert-parameter actions, because (i) the router outputs are softmax-gated and invariant to a shift in the router logits along the routing direction, and (ii) the expert parameters enter the residual stream through a low-rank projection that is invariant along the null space of $\mathbf{C}_{\boldsymbol{e}_k}$. Consequently, a tight KL constraint is necessary but not sufficient for staying close to the pretrained editor in the parameter space.

SVD-based perturbation, by contrast, regularizes the *trajectory* of the weight updates directly: it confines stochastic action sampling to perturbations of the form $\mathbf{U}\,\mathrm{diag}(\boldsymbol{\sigma}+\boldsymbol{\eta})\,\mathbf{V}^\top$ with $(\mathbf{U}, \mathbf{V})$ fixed to the singular vectors of the pretrained expert. This keeps the exploration inside the principal subspace of the LiveEdit-trained expert, which is the regime where the editor's reliability/locality/generality guarantees remain valid. The KL term cannot enforce this subspace confinement because it is blind to the decomposition of the weight update.

K.2 Restriction vs. Exploration

The two mechanisms also serve qualitatively opposite purposes. The KL term is a *restriction*: it pulls the policy back towards π_{ref}, reducing exploration. SVD-based perturbation is an *exploration* mechanism: it supplies the G stochastic actions that GRPO requires for group-relative advantage estimation. Removing SVD-based perturbation eliminates action stochasticity altogether; tightening β cannot restore it, and in fact pushes the policy further towards the deterministic reference.

Concretely, in the limit of large β, the policy collapses to π_{ref}, the G sampled actions become (near-) identical, and the group-relative advantage $\mathcal{A}_t^{(i)} \to 0$ for all i. The RL gradient vanishes regardless of how well-structured the resulting policy is. SVD-based perturbation is precisely the device that maintains non-trivial $\mathcal{A}_t^{(i)}$ while respecting the structural prior of the pretrained expert.

K.3 Empirical Evidence from the "Gaussian" Ablation

The "Gaussian" ablation row of Table 3 provides the most direct available evidence: even with the KL coefficient β unchanged, replacing SVD-confined perturbation by unstructured i.i.d. Gaussian noise injected directly into the entries of $(\mathbf{C}, \mathbf{R})$ costs roughly 10 percentage points of multi-hop accuracy (portability drops from 41.74% to the 29–33% range at $\#\text{Edit}= 100$ on BLIP2-OPT). The two configurations carry identical stochasticity budgets and identical KL regularization; they differ only in whether the noise respects the principal subspace of the pretrained expert. The gap therefore isolates the contribution of the *structure* of the perturbation—precisely the property that tuning β cannot compensate for, since β acts on the output distribution and is blind to the decomposition of the weight update. Combined with the large-β collapse argument in Section K (Restriction vs. Exploration), this evidence indicates that KL-divergence regularization and SVD-based perturbation address orthogonal failure modes and are not substitutable.

K.4 Summary

KL-divergence regularization and SVD-based perturbation address orthogonal failure modes: the former prevents the output distribution from drifting away from the pretrained editor, while the latter supplies structured action stochasticity that respects the editor's principal subspace. The two are complementary rather than substitutable: the "Gaussian" ablation row of Table 3 confirms that discarding the principal-subspace structure of the perturbation degrades multi-hop reasoning even when the KL term is intact, while the large-β collapse argument in Section K (Restriction vs. Exploration) shows that discarding stochasticity altogether (by tightening KL) collapses the group-relative advantages that GRPO relies on.