

GRIP: Algorithm-Agnostic Machine Unlearning for Mixture-of-Experts via Geometric Router Constraints

Andy Zhu¹, Rongzhe Wei¹, Yupu Gu², Pan Li¹

¹ Georgia Institute of Technology

² Tsinghua University

Abstract

Machine unlearning in Mixture-of-Experts (MoE) large language models presents a critical yet underexplored challenge. Current unlearning methods applied to MoE architectures often exploit dynamic routing as an optimization shortcut: rather than genuinely erasing knowledge from expert parameters, they manipulate routers to redirect queries away from the originally assigned experts. This not only causes severe utility degradation but also leaves hazardous knowledge intact. Consequently, adversaries can bypass the router to recover sensitive information directly from dormant experts. In this study, we propose *Geometric Routing Invariance Preservation (GRIP)*, an algorithm-agnostic framework that resolves these failure modes by enforcing hard geometric constraints on router updates. By projecting router gradient updates into the null space of the retain set’s routing matrix, GRIP suppresses routing manipulation without freezing the router entirely, thereby directing the unlearning pressure into the expert parameters themselves across all relevant experts. GRIP offers two complementary variants: training-time stochastic projection and a post-training closed-form analytical correction. Extensive experiments on two MoE models across hazardous knowledge removal and copyright unlearning benchmarks demonstrate that GRIP restores routing stability from 0.21 to ≥ 0.94 , improves retain accuracy by up to 89%, and reduces white-box adversarial knowledge recovery from 11% to just 3% while in line with dense-architecture unlearning under black-box prompt attack, establishing geometric constraints as a principled solution for genuine unlearning in sparse MoE architectures.¹

1 Introduction

Large language models (LLMs) have achieved remarkable progress across diverse domains (Brown et al., 2020; Touvron et al., 2023; Gemini Team et al., 2023), yet they can unintentionally memorize sensitive information from training data, including personal details, proprietary knowledge, and harmful content (Carlini et al., 2020; Zhang et al., 2023; Nasr et al., 2025; Wei et al., 2024). Machine unlearning (MU) has emerged as a principled solution, providing mechanisms to selectively remove such information from trained models without retraining from scratch (Bourtoule et al., 2021; Cao & Yang, 2015), a capability essential for regulatory compliance with privacy laws such as GDPR’s “right to be forgotten” (European Parliament & Council of the European Union, 2016; Villaronga et al., 2018) and post-deployment safety (Mitchell et al., 2022; Weidinger et al., 2021). Existing efficient unlearning methods (Yao et al., 2023; Kurmanji et al., 2023; Golatkar et al., 2020; Li et al., 2024; Jia et al., 2024; Korbak et al., 2023; Zhang et al., 2024) typically optimize a forget objective (e.g., maximizing loss on target data) while preserving general model capabilities, and have demonstrated strong empirical performance on standard dense LLM architectures.

However, existing unlearning methods are primarily developed and validated on dense architectures, where all parameters engage uniformly with every input. This assumption

¹Code available at <https://github.com/AndyZhu311/GRIP>

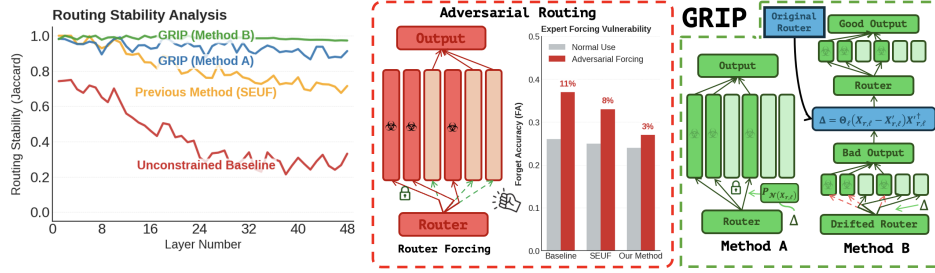


Figure 1: **GRIP prevents expert selection shift and secures MoE models against adversarial knowledge recovery.** (a) **The Problem:** Standard unlearning exploits the router as a shortcut, diverting forget-set queries away from knowledgeable experts rather than erasing their parameters. Routing Stability (RS, Eq (2)) collapses from 1.0 to ≈ 0.2 across layers. (b) **Forcing Vulnerability:** Baselines merely suppress knowledge retrieval via routing; a white-box adversary can bypass this by directly forcing access to dormant experts, recovering sensitive information at a significantly higher rate than GRIP-unlearned models. (c) **Our Solution:** GRIP imposes hard geometric constraints on router updates via Training-Time Null-Space Enforcement (*Method A*) and Post-Training Analytical Realignment (*Method B*), restoring routing stability (≥ 0.94) while achieving genuine knowledge erasure.

breaks down in Mixture-of-Experts (MoE) models (Jiang et al., 2024), which are increasingly deployed in frontier systems due to their computational efficiency. In MoE layers, a learned router dynamically selects a sparse subset of expert networks for each input. Applying standard unlearning methods to these models can cause unstable expert selection for the knowledge that must be retained (see Fig. 1(a)), severely degrading the model’s utility. More critically, shifts in expert selection may leave the knowledge slated for unlearning intact within the original expert parameters. This introduces a potential risk: if adversaries can manipulate the router to access these dormant experts, they can easily recover the knowledge planned to be unlearned. We term this exploit an *Expert Forcing* attack, a vulnerability unique to MoE models as, while suppressing access to knowledge without genuinely erasing it is a known problem in dense unlearning as well (Yang et al., 2026; Xu et al., 2026), dense models lack unentangled pathways to route around for this paradigm.

This vulnerability was first observed by SEUF (Zhuang et al., 2025), which attempts to address the resulting instability by applying soft regularization penalties and restricting unlearning updates to the single most relevant expert. However, empirical observations reveal that soft penalties offer limited protection against routing manipulation (see Fig. 1(a)). Furthermore, updating only a single expert fails to eliminate hazardous knowledge stored across the other initially activated experts, leaving the model highly susceptible to the aforementioned Expert Forcing attacks.

To overcome these challenges, we propose *Geometric Routing Invariance Preservation* (GRIP), a versatile framework for MoE unlearning that integrates with many existing optimization methods. GRIP imposes *hard geometric constraints* on router updates to suppress expert selection shift and ensure unlearning pressure across all activated experts. Specifically, GRIP exerts fine-grained control over the routing mechanism by constraining router gradient updates to the null space defined by the retain set’s relevant experts. This strict constraint substantially mitigates the routing instability responsible for utility collapse (Fig. 1(a)). Concurrently, GRIP enables the optimization of both expert knowledge parameters and unselected routing paths. This broadens the optimization space compared to fixing all router paths, allowing the model to erase the targeted knowledge in a more thorough way. Consequently, GRIP achieves significantly stronger resistance to adversarial expert forcing attacks (Fig. 1(b)). Algorithmically, GRIP offers two complementary realizations (Figure 1(c)): an online approach (Method A) that enforces the constraint during training via stochastic projection, and an offline approach (Method B) that applies a single closed-form analytical correction after unlearning to realign drifted routers.

We validate GRIP on a 14.3B-parameter and a 30B-parameter MoE model (2.7B and 3B active parameters, respectively) across hazardous knowledge removal (Li et al., 2024) and

copyright unlearning (Shi et al., 2025), instantiated with four distinct unlearning objectives. Across all settings, GRIP restores routing stability from 0.21 - 0.45 to at least 0.94, improves retain accuracy by up to 89%, and reduces adversarial knowledge recovery under white-box expert forcing from 11% to 3%, while remaining similarly resistant to black-box prompt attacks as dense-architecture unlearning. Together, these results show that hard geometric routing constraints provide a principled and scalable route to genuine unlearning in MoE models, where knowledge is erased from expert parameters rather than merely hidden by routing manipulation.

2 Related Work

LLM Unlearning Methods. Machine unlearning for LLMs has been studied along several axes. A dominant family of methods operates through objective modification, including gradient ascent on forget data (Jang et al., 2023), gradient difference with retain regularization (Kurmanji et al., 2023), and negative preference optimization (Zhang et al., 2024). A second line of work targets internal representations directly, either by misdirecting activations away from harmful concepts (Li et al., 2024) or by identifying and editing knowledge localized in specific layers or neurons (Meng et al., 2022; Wu et al., 2023). More recent approaches explore parameter-efficient formulations via task arithmetic (Ilharco et al., 2022) and second-order optimization (Jia et al., 2024), and in-context strategies that avoid weight modification entirely (Pawelczyk et al., 2023). While these methods have demonstrated strong empirical performance, they are developed and validated exclusively on dense architectures, where all parameters engage uniformly with every input. This assumption is violated in sparse MoE models, motivating the need for unlearning methods specifically designed for dynamic routing architectures.

Unlearning in Mixture-of-Experts Models. Sparse MoE architectures have become ubiquitous in frontier LLMs, powering systems such as Mixtral (Jiang et al., 2024), DeepSeek-V3 (DeepSeek-AI et al., 2024), and GPT-4 (OpenAI et al., 2023). Unlike dense models, MoE layers route each input through a learned sparse selection of experts, introducing a dynamic computation graph that poses fundamental challenges for post-training modification. To the best of our knowledge, the only prior work that targets MoE unlearning, SEUF (Zhuang et al., 2025), first identified that standard unlearning objectives tend to cause routers to shift expert selections, and proposed to mitigate this through soft regularization on router weights and to restrict parameter updates to the single highest-attributed expert. GRIP departs from it on two conceptual points rather than by tuning its penalty: we treat routing preservation as a feasibility condition satisfied by construction rather than as a soft penalty that discourages drift but admits it, and we treat erasure as inherently *multi-expert*, since the targeted knowledge persists across the other initially-activated experts where it remains recoverable. These instantiate as a hard null-space projection in place of a soft L_2 penalty, and as leaving all expert weights free to update in place of a single-expert restriction. We also propose Expert Forcing, a diagnostic for whether knowledge survives in bypassed experts.

MoE Routing Dynamics and Stability. Recent work has shown that MoE routing patterns encode meaningful semantic structure, with individual experts specializing in specific syntactic roles, domains, and reasoning patterns (Gururangan et al., 2023; Zoph et al., 2022; Fedus et al., 2022), making routing preservation essential for maintaining model capabilities. While training-time stabilization techniques such as load balancing losses (Fedus et al., 2022) and expert choice routing (Zhou et al., 2022) are well-established, they address a different setting than post-training unlearning. To our knowledge, GRIP is the first work to formalize routing preservation as a hard geometric constraint for machine unlearning.

3 Preliminaries

Machine Unlearning. Given a model trained on an original dataset D_{train} , machine unlearning partitions this data into a forget set $D_f \subset D_{train}$ and a retain set $D_r = D_{train} \setminus D_f$. The objective is to produce updated model parameters that behave as if the model were

trained exclusively on D_r . This retain-only oracle is unattainable for any scalable method, since retraining a frontier MoE from scratch is precisely the cost unlearning exists to avoid; Section 6 discusses what this implies for GRIP’s objective.

Mixture-of-Experts Layers. A sparse MoE layer processes an input representation $x_\ell \in \mathbb{R}^d$ at layer ℓ using conditional computation, whereby only a fraction of the layer’s parameters are activated per input. The layer contains E expert networks and a router parameterized by $\Theta_\ell \in \mathbb{R}^{E \times d}$. For an input x_ℓ , the router computes scores $s_\ell = \Theta_\ell x_\ell \in \mathbb{R}^E$ via linear projection and selects a subset of top- k experts, $\mathcal{S}_\ell(x_\ell; \Theta_\ell) \subseteq \{1, \dots, E\}$. The layer output is:

$$y_\ell = \sum_{j \in \mathcal{S}_\ell(x_\ell)} \frac{\exp(s_{\ell,j})}{\sum_{j' \in \mathcal{S}_\ell(x_\ell)} \exp(s_{\ell,j'})} \cdot f_j(x_\ell) \quad (1)$$

where $f_j(x_\ell)$ denotes the output of the j -th selected expert, and the softmax normalization is computed only over the selected top- k experts rather than all E experts in the layer.

Expert Selection Shift and Routing Stability. During unlearning, gradient-based methods update router parameters $\Theta_\ell \rightarrow \Theta_\ell + \Delta\Theta_\ell$, which can induce *expert selection shift*: the set of selected experts changes from $\mathcal{S}_\ell^{\text{pre}}(x) := \mathcal{S}_\ell(x; \Theta_\ell)$ before unlearning to $\mathcal{S}_\ell^{\text{post}}(x) := \mathcal{S}_\ell(x; \Theta_\ell + \Delta\Theta_\ell)$ after, where we use $\mathcal{S}_\ell^{\text{pre}}$ and $\mathcal{S}_\ell^{\text{post}}$ as shorthands hereafter. As introduced in Section 1, unlearning objectives can exploit this by manipulating $\Delta\Theta_\ell$ to redirect queries away from knowledgeable experts rather than erasing knowledge from their parameters. We formally quantify this effect using *routing stability* (RS), the Jaccard similarity between pre- and post-unlearning expert selections over the retain set D_r :

$$\text{RS}_\ell = \frac{1}{|\mathcal{Q}|} \sum_{x \in \mathcal{Q}} \frac{|\mathcal{S}_\ell^{\text{pre}}(x) \cap \mathcal{S}_\ell^{\text{post}}(x)|}{|\mathcal{S}_\ell^{\text{pre}}(x) \cup \mathcal{S}_\ell^{\text{post}}(x)|} \quad (2)$$

where $\mathcal{Q} \subseteq D_r$ is the set of evaluation queries drawn from the retain set. An RS of 1 indicates perfect routing preservation, while 0 indicates complete routing collapse. Note that RS measures fidelity to the *original* model’s routing, not to a retain-only oracle’s (Section 6).

4 Methodology

In standard dense LLMs, unlearning objectives operate over a static computation graph where all parameters engage uniformly with every input. MoE architectures fundamentally alter this landscape: the computation path through expert networks is itself a function of the router parameters $\{\Theta_\ell\}$, creating a dual optimization landscape where the unlearning objective $\mathcal{L}_{\text{MU}}$ can be minimized via two distinct strategies, either by (1) genuinely erasing knowledge from expert parameters W , or (2) manipulating the router to redirect forget-set queries away from knowledgeable experts entirely. Critically, the router is the cheaper descent direction, since a single linear map gates every expert in the layer, and we observe correspondingly larger norms for $\nabla_{\Theta} \mathcal{L}_{\text{MU}}$ than $\nabla_W \mathcal{L}_{\text{MU}}$, creating a structural bias toward the routing shortcut. Without proper constraints, the optimizer will therefore preferentially converge to strategy (2), producing the expert selection shift and superficial forgetting.

GRIP addresses this by imposing a hard constraint directly on router updates. Formally, given any base unlearning objective $\mathcal{L}_{\text{MU}}(D_f, D_r)$, GRIP augments the optimization with a *retain-side routing invariance constraint*: router parameter updates $\Delta\Theta_\ell$ must preserve the expert selections of the retain set D_r , ensuring that $\mathcal{S}_\ell^{\text{post}}(x) = \mathcal{S}_\ell^{\text{pre}}(x)$ for all $x \in D_r$. By making the routing shortcut infeasible on the constrained inputs, this pushes the optimization to achieve $\mathcal{L}_{\text{MU}}$ through knowledge erasure within expert weights instead. Because expert parameter updates remain unconstrained, unlearning can proceed across *all* relevant experts rather than being restricted to a single expert as the existing work does (Zhuang et al., 2025). We instantiate this principle via two variants: *Method A* enforces the constraint online at each training step via stochastic gradient projection (Sec. 4.2), while *Method B* applies a single closed-form analytical correction after unlearning to realign drifted routers (Sec. 4.3). The constraint is placed on the retain side in order both to preserve model

utility and, because router rows are shared across forget and retain inputs, to limit how freely the router can reroute forget queries through that same expert; Appendix G.3 compares this against forget-side and combined alternatives.

4.1 From Routing Invariance to Selection-Preserving Constraints

We need to avoid potentially too restrictive constraints on the router. The retain-side routing invariance constraint requires $\mathcal{S}_\ell^{\text{post}}(x) = \mathcal{S}_\ell^{\text{pre}}(x)$ for all $x \in D_r$. Since the top- k selection function is piecewise constant and non-differentiable with respect to $\Delta\Theta_\ell$, we seek a differentiable sufficient condition. Let $X_{r,\ell} \in \mathbb{R}^{d \times N_r}$ denote the matrix of retain representations at layer ℓ , where each column $x_{r,\ell}^{(i)} \in \mathbb{R}^d$ is the hidden state of the i -th retain input at that layer. A natural sufficient condition is logit invariance $\Delta\Theta_\ell X_{r,\ell} = 0$, requiring that any router update resides in $\text{Null}(X_{r,\ell}^\top)$. However, this global constraint is overly restrictive: routing stability requires only that the top- k selection order is preserved, not that all logits remain static. Enforcing zero logit change for every retain input unnecessarily prohibits benign modifications to non-selected experts, severely limiting the feasible optimization landscape and inducing exactly the parameter rigidity that GRIP is designed to avoid.

Finer-grained Constraints. To resolve this, we decompose the global constraint into an asymmetric, expert-specific formulation. Instead of rigidly fixing the entire routing distribution, we strictly preserve only the top- k selections. The routing paths for non-selected experts can be freely adjusted, provided their scores do not exceed the threshold to become a top- k expert. For expert j at layer ℓ , let $\mathcal{I}_{j,\ell} = \{i : j \in \mathcal{S}_\ell^{\text{pre}}(x_{r,\ell}^{(i)})\}$ denote the retain inputs for which expert j was selected. We impose:

$$(\Delta\Theta_\ell)_{j,:} x_{r,\ell}^{(i)} = 0 \quad \forall i \in \mathcal{I}_{j,\ell}, \quad (\Delta\Theta_\ell)_{j,:} x_{r,\ell}^{(i)} \leq \tau_{i,j,\ell} \quad \forall i \notin \mathcal{I}_{j,\ell} \quad (3)$$

where $s_k = (\Theta_\ell)_k : x_{r,\ell}^{(i)} \in \mathbb{R}$ denotes the routing score of expert k for input $x_{r,\ell}^{(i)}$ under the pre-unlearning router, and $\tau_{i,j,\ell} = \min_{k \in \mathcal{S}_\ell^{\text{pre}}(x_{r,\ell}^{(i)})} [s_k - s_j]$ is the selection margin of expert j on input i , i.e., the minimum score gap between any selected expert and expert j itself. The equality constraint preserves routing decisions for inputs that selected expert j , while the inequality constraint permits score modifications for non-selected inputs, so long as they do not breach the selection margin. Together, these conditions ensure that $\mathcal{S}_\ell^{\text{post}}(x) = \mathcal{S}_\ell^{\text{pre}}(x)$ on every $x \in D_r$ over which they are actually enforced, while substantially relaxing the constraints for genuine knowledge erasure.

However, enforcing these constraints exactly over the full retain set D_r requires projecting onto a global polytope satisfying $\mathcal{O}(|D_r| \times E)$ simultaneous constraints while caching $\mathcal{O}(|D_r| \times d)$ intermediate representations, which is strictly intractable at LLM scale. The preservation we obtain is therefore empirical rather than worst-case, which motivates the batch-level approximations developed in the following sections.

4.2 Training-Time Enforcement via Stochastic Projection

Equation (3) gives the ideal selection-preserving constraints for a router update. Enforcing these constraints exactly over the full retain set at every optimization step is intractable, so a training-time variant applies them at the level of the current retain minibatch. For simplicity, in this subsection we reuse $X_{r,\ell}$ and $\mathcal{I}_{j,\ell}$ to denote the retain representations and selected-example index set for the current minibatch at layer ℓ .

For expert j at layer ℓ , let $u_{j,\ell} \in \mathbb{R}^d$ denote the unconstrained update proposed for the j -th router row by the base unlearning algorithm. GRIP replaces this update by a projected update $\bar{u}_{j,\ell}$ that lies in the minibatch feasible set:

$$\mathcal{K}_{j,\ell} = \left\{ u \in \mathbb{R}^d : \langle u, x_{r,\ell}^{(i)} \rangle = 0 \quad \forall i \in \mathcal{I}_{j,\ell}, \quad \langle u, x_{r,\ell}^{(i)} \rangle \leq \tau_{i,j,\ell} - \varepsilon \quad \forall i \notin \mathcal{I}_{j,\ell} \right\}, \quad (4)$$

where $\varepsilon > 0$ is a safety margin that improves numerical robustness and prevents tie-induced changes in the top- k selected set. We enforce membership in $\mathcal{K}_{j,\ell}$ in two stages.

Stage 1: Equality projection. We first enforce the hard equality constraints for the inputs that originally selected expert j . Let $X_{j,\ell}^{\text{eq}} = [x_{r,\ell}^{(i)}]_{i \in \mathcal{I}_{j,\ell}} \in \mathbb{R}^{d \times |\mathcal{I}_{j,\ell}|}$ collect the corresponding retain representations. The orthogonal projector onto the complement of their span is $P_{j,\ell}^{\text{eq}} = I - X_{j,\ell}^{\text{eq}} (X_{j,\ell}^{\text{eq}\top} X_{j,\ell}^{\text{eq}})^{\dagger} X_{j,\ell}^{\text{eq}\top}$, and the intermediate update is $\tilde{u}_{j,\ell} = P_{j,\ell}^{\text{eq}} u_{j,\ell}$. By construction, $\langle \tilde{u}_{j,\ell}, x_{r,\ell}^{(i)} \rangle = 0$ for all $i \in \mathcal{I}_{j,\ell}$, so the score of expert j is unchanged on the retain inputs that actively use it.

Stage 2: Inequality correction. The update $\tilde{u}_{j,\ell}$ must still satisfy the margin constraints for retain inputs that did not originally select expert j . For each $i \notin \mathcal{I}_{j,\ell}$, define the half-space $\mathcal{H}_{i,j,\ell} = \{u \in \mathbb{R}^d : \langle u, x_{r,\ell}^{(i)} \rangle \leq \tau_{i,j,\ell} - \varepsilon\}$. Exact Euclidean projection of $\tilde{u}_{j,\ell}$ onto the full intersection $\bigcap_{i \notin \mathcal{I}_{j,\ell}} \mathcal{H}_{i,j,\ell}$ remains expensive, so we approximate it with a stochastic alternating-projection procedure. If a sampled input $i \notin \mathcal{I}_{j,\ell}$ violates the margin, i.e., $\langle \tilde{u}_{j,\ell}, x_{r,\ell}^{(i)} \rangle > \tau_{i,j,\ell} - \varepsilon$, the Euclidean projection onto the boundary of $\mathcal{H}_{i,j,\ell}$ is $\tilde{u}_{j,\ell} \leftarrow \tilde{u}_{j,\ell} - \frac{\langle \tilde{u}_{j,\ell}, x_{r,\ell}^{(i)} \rangle - (\tau_{i,j,\ell} - \varepsilon)}{\|x_{r,\ell}^{(i)}\|_2^2} x_{r,\ell}^{(i)}$. We implement this stage using a randomized Kaczmarz-style update for systems of linear *inequalities* (Leventhal & Lewis, 2010; De Loera et al., 2017), an efficient stochastic approximation to the projection onto the polyhedral feasible region. Further discussion of this algorithm is in Appendix C.

After these stages, the resulting $\tilde{u}_{j,\ell}$ is used as the update for the j -th router row. Importantly, GRIP modifies only router updates; expert weights W and other parameters remain exactly those produced by the standard unlearning algorithms such as gradient difference or NPO. Imposing these hard constraints incurs a $1.71 \times$ training overhead. Constraint storage requirements are negligible due to a tiered low-rank caching scheme that holds the active memory footprint below 1% of model size; further analysis of computational and storage costs is contained within Appendix D.

4.3 Post-Training Router Realignment

The training-time variant enforces routing invariance at every step, but faces a limitation in deep multi-layer models: representation drift. As unlearning updates parameters in layers $1, \dots, \ell - 1$, the retain representations $X_{r,\ell}$ shift to $\tilde{X}_{r,\ell}$, rendering precomputed null-space projectors increasingly stale in deeper layers. Rectifying this by continuously recomputing projectors necessitates $\mathcal{O}(Ed^3)$ operations per layer per update step, associating exact online enforcement with high computational cost.

The post-training variant addresses this by deferring enforcement entirely. Instead of projecting router updates at every training step, we first run the chosen unlearning algorithm without router constraints, then apply a single post-hoc correction that restores the router’s behavior on the retain set. Let Θ_ℓ and $X_{r,\ell}$ denote the pretrained router and retain representations at layer ℓ , and let $\tilde{\Theta}_\ell$ and $\tilde{X}_{r,\ell}$ denote their counterparts after unconstrained unlearning. We seek a correction Δ_ℓ such that the original routing scores are restored on the post-unlearning retain representations: $(\tilde{\Theta}_\ell + \Delta_\ell) \tilde{X}_{r,\ell} = \Theta_\ell X_{r,\ell}$. This yields the closed-form least-squares solution:

$$\Delta_\ell = \Theta_\ell X_{r,\ell} \tilde{X}_{r,\ell}^{\dagger} - \tilde{\Theta}_\ell, \quad \tilde{X}_{r,\ell}^{\dagger} = \tilde{X}_{r,\ell}^{\top} (\tilde{X}_{r,\ell} \tilde{X}_{r,\ell}^{\top} + \lambda I)^{-1}, \quad (5)$$

where $\lambda = 10^{-6}$ is a small constant for numerical stability (further details in Appendix D.4).

The efficacy of this correction relies on the observation that unlearning induces bounded perturbations in representation space, such that the drift $X_{r,\ell} \rightarrow \tilde{X}_{r,\ell}$ preserves the local topology of the retain manifold. This assumption holds almost universally except for slight breakdowns under very aggressive forgetting, at deep layers, or for forget sets entangled with the retain set; the acceptance check of Appendix D.4 flags the layers where it fails,

Table 1: Comprehensive evaluation across WMDP and MUSE benchmarks. ↓/↑ indicates lower/higher is better. Best methods in each category are **bolded**. Best overall method is **highlighted**. Statistics that were controlled are underlined. Router stability values calculate Jaccard similarity in expert selections using Equation (2).

Method	WMDP-Cyber			MUSE				
	FA↓	RA↑	RS↑	C1↓	C2↓	C3→0	C4↑	RS↑
GD	0.26	0.32	0.21	0.05	0.03	-25.6	4.31	0.26
SEUF + GD	0.27	0.44	0.90	0.07	0.09	-21.4	8.54	0.92
Constrained + GD	0.24	0.51	0.94	0.04	0.02	-26.8	11.4	0.96
PTC + GD	0.26	0.55	0.99	0.01	0.00	-30.4	9.42	1.00
KL	0.28	0.35	0.42	15.3	22.9	-33.5	21.3	0.32
SEUF + KL	0.25	0.49	0.89	18.5	30.2	-32.4	37.6	0.91
Constrained + KL	0.27	0.58	0.94	17.2	24.6	-30.1	42.7	0.96
PTC + KL	0.26	0.59	1.00	16.6	22.4	-32.9	41.8	1.00
NPO	0.33	0.48	0.45	16.3	22.1	-39.5	12.6	0.43
SEUF + NPO	0.34	0.53	0.91	14.2	28.7	-35.3	35.1	0.89
Constrained + NPO	0.29	0.62	0.97	15.3	23.8	-42.3	41.6	0.94
PTC + NPO	0.25	0.59	0.99	19.3	27.3	-36.1	41.2	1.00
RMU	0.26	0.35	0.43	0.00	0.01	-41.3	14.2	0.37
SEUF + RMU	0.25	0.58	0.89	0.00	0.03	-40.8	35.1	0.91
Constrained + RMU	0.24	0.65	0.96	0.02	0.00	-42.8	40.5	0.95
PTC + RMU	0.28	0.66	0.99	0.00	0.02	-41.9	39.2	1.00
Dense Baseline	0.28	0.70	N/A	17.0	25.0	-43.5	44.6	N/A

which fall back to Method A. Consequently, the router’s linear decision boundaries do not need to be re-learned from scratch but simply require geometric realignment to the shifted input distribution, which is exactly what Equation (5) achieves. This variant replaces per-step projections by a single closed-form correction per layer, reducing the projection cost from $\mathcal{O}(KLEd^3)$ to $\mathcal{O}(Ld^3)$ and the total overhead from $1.71\times$ to $1.21\times$ training time; Appendix D.4 gives the full cost and memory analysis.

5 Experimental Results

Datasets and Metrics. We adopt Qwen3-30B-A3B (Qwen Team, 2025) and Qwen1.5-MoE-A2.7B-Chat (Bai et al., 2023) as the backbones. We evaluate on the WMDP (Li et al., 2024) and MUSE (Shi et al., 2025) benchmarks using standard protocols as detailed by their original papers, where we approximately controlled the forget metrics (Appendix A.1), i.e., all models were unlearned until they matched the baseline level for forget metrics, to provide the most accurate comparison on retain ability. We report Forget Accuracy (FA) and Retain Accuracy (RA) for the WMDP dataset and metrics C1 through C4 for the MUSE dataset as well as Routing Stability (RS) as defined in Eq. (2) (see Appendix A.1 for more details on training and metrics).

5.1 Main Results

Table 1 presents results across four unlearning objectives (GD, KL, NPO, RMU) on both WMDP-Cyber and MUSE benchmarks, where forget metrics are controlled to approximately matched levels to isolate differences in utility preservation. We organize our analysis around three findings: (i) GRIP consistently restores routing stability to near-perfect levels, (ii) this stability translates directly into superior retain accuracy that approaches dense-model baselines, and (iii) GRIP subsumes SEUF across all settings.

Significant Routing Stability Improvements. GRIP maintains consistent near-perfect routing stability ($RS \geq 0.94$) across all base unlearning methods. While baseline approaches suffer severe routing disruption, with stability dropping as low as 0.21, our post-training

correction (PTC) consistently achieves **perfect or near-perfect stability (0.99-1.00)** and the online constraints also maintain high stability (0.94-0.97).

Superior Knowledge Retention With Unlearning Effectiveness. The above routing stability naturally translates into utility improvements. Our framework demonstrates substantial gains in retain accuracy, improving 23–89% relative to baseline methods. For instance, on WMDP-Cyber with RMU, retain accuracy improves from 0.35 to 0.65–0.66 (+86–89%). Critically, this routing preservation does not compromise unlearning quality, as all GRIP variants reach forget accuracy at or below the baseline, confirming that the utility gains stem from eliminating the routing shortcut rather than from under-unlearning.

Dense-Level Utility Recovery. The bottom row reports dense-architecture baselines from the original papers (Li et al., 2024; Shi et al., 2025) (models in Appendix A.1). Our constrained MoE methods recover utility preservation scores of 40.5–42.7 on C4 for KL, NPO and RMU, closing the gap to the unlearning performance on the dense LLMs (44.6), while unconstrained MoE methods may collapse the utility to as low as 4.3. GRIP also preserves broad capabilities on MoE evaluations, retaining 86–90% of zero-shot MMLU and 88–89% of TruthfulQA against 66% and 69% for unconstrained baselines (Appendix A.3).

Direct Comparison with SEUF. SEUF’s soft regularization (Zhuang et al., 2025) does not overcome the structural bias toward the routing shortcut. At approximately matched forgetting, it underperforms both GRIP variants on every retain-side metric in Table 1: RS of 0.89–0.92 against GRIP’s ≥ 0.94 , and C4 plateauing at 35.1–37.6 against 39.2–42.7 on those same objectives. Soft penalties therefore permit residual router manipulation that diverts retain-set queries to suboptimal experts, which is exactly what a hard constraint forecloses. Appendix G.2 separates the hard/soft and single/all-expert axes.

Performance on 14.3B-parameter MoE. To ensure a fair comparison and to evaluate whether GRIP generalizes across MoE architectures, we additionally evaluate on the Qwen1.5-MoE-A2.7B-Chat model used in the original SEUF study. We restrict this comparison to WMDP-Cyber with the RMU objective, as these represent SEUF’s strongest reported configuration (other benchmarks are omitted because SEUF was not originally evaluated on them). As detailed further in Appendix F, GRIP maintains its advantage even on SEUF’s native architecture: while achieving comparable forget accuracy (0.25–0.27 vs 0.25), we demonstrate superior retain accuracy (0.61–0.63 vs 0.53, a 15–19% improvement) and routing stability (0.94–0.99 vs 0.87, an 8–14% improvement). These results also confirm that GRIP’s geometric constraints do not lose efficacy across different architectures.

5.2 Unlearning-Utility Tradeoffs Across Training

We track the forget-utility tradeoff across the full unlearning process by evaluating intermediate checkpoints for both unconstrained baselines and GRIP variants. Figure 2 plots C4 against C1 metrics at each checkpoint. At every level of forgetting, GRIP-constrained models achieve strictly higher utility than their unconstrained counterparts, confirming that the improvement is not an artifact of early stopping. Unconstrained methods exhibit a steep utility collapse early in training as routing shifts cascade through the network, whereas GRIP maintains a gradual, controlled tradeoff throughout. We next ablate the individual design choices behind these gains.

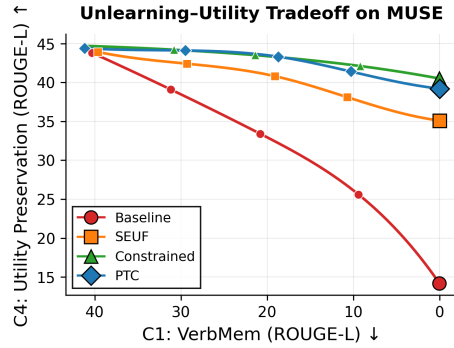


Figure 2: Forget-utility tradeoff on MUSE (RMU). Each curve traces intermediate checkpoints throughout training.

5.3 Ablation Studies and Design Choices

Finally, we conduct ablation analysis on GRIP as well as analysis of null-space thresholds; Appendix G reports further ablations isolating each design choice.

Table 2: Analysis of the null-space threshold ϵ_{null} , distinct from the routing safety margin ϵ of Equation (4).

ϵ_{null}	RS ($\uparrow$)	FA ($\downarrow$)	RA ($\uparrow$)
10^{-4}	0.97	0.31	0.59
10^{-3}	0.96	0.28	0.62
10^{-2}	0.96	0.24	0.65
10^{-1}	0.89	0.25	0.59

Table 3: Ablation analysis of constraint methods. Frozen Router sets $\Delta\Theta_\ell = 0$ and directs all unlearning gradient into the expert MLPs.

Constraint	FA $\downarrow$	RA $\uparrow$	RS $\uparrow$	Time $\downarrow$
None (Baseline)	0.26	0.35	0.43	1.0 $\times$
Frozen Router	0.26	0.32	0.57	1.0 $\times$
Full Null Space	0.38	0.68	0.96	1.67 $\times$
Expert-Specific	0.24	0.65	0.96	1.71 $\times$
Post-Training	0.28	0.66	0.99	1.21 $\times$

Constraint Formulations. As shown in Table 3, while a strict full null-space constraint ($\Delta\Theta_r = 0$) successfully preserves routing (RS = 0.96), it overly restricts parameter plasticity and degrades unlearning effectiveness (FA = 0.38). Our Expert-Specific formulation resolves this trade-off, achieving superior erasure (FA = 0.24) while maintaining high stability (RS = 0.96), though at a 1.71 $\times$ training overhead due to per-step projection. Post-Training Correction (PTC) achieves the highest stability (RS = 0.99) at only 1.21 $\times$ overhead, though its unlearning lags slightly (FA = 0.28) as interim router drift forces expert parameters to optimize against a non-stationary routing distribution. We recommend the Expert-Specific formulation when maximum erasure fidelity is required, and PTC for resource-constrained deployments where the efficiency–efficacy tradeoff favors throughput.

As freezing router *weights* does not freeze routing *decisions*, since routing also depends on the drifting representations reaching each layer (Appendix G.1), GRIP’s gains do not come merely from immobilizing the router. Table 3 shows that freezing router weights leaves forget efficacy unchanged (FA = 0.26) but collapses retain accuracy to 0.32, at baseline level and roughly half of GRIP’s 0.65.

Threshold Sensitivity Analysis. We sweep the null-space threshold of Appendix B across four orders of magnitude ($\epsilon_{\text{null}} \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$) and use $\epsilon_{\text{null}} = 10^{-2}$ throughout. As shown in Table 2, tighter values empty the null space in later layers and degrade unlearning (FA = 0.31), while looser ones cost routing stability (0.89), leaving 10^{-2} on the Pareto frontier between stability and plasticity. Appendix G.4 resolves this sweep by depth band and finds that the optimum does not shift, though sensitivity to departures from it does; layer-adaptive scheduling remains future work.

5.4 Adversarial Evaluation via Expert Forcing

The preceding results establish that GRIP preserves model utility while achieving effective forgetting on standard benchmarks. However, these metrics alone cannot distinguish genuine knowledge erasure from retrieval suppression, where targeted information persists in expert parameters but is rendered inaccessible through altered routing paths. Thus, we design an adversarial evaluation that simulates a white-box attacker with full model access who bypasses the post-unlearning router and forces generation through the original pre-unlearning expert selections. Specifically, for forget-set queries that exhibit routing shifts after unlearning, we override the learned router decisions and force the model to use the expert set selected by the pre-unlearning router (full protocol in Appendix E), a worst-case diagnostic rather than a deployment threat model (Section 5.5).

Table 4: Expert forcing attack on WMDP. Recovery Rate is the fraction of forgotten queries recovered when the router is bypassed. GRIP is Method A.

Method	Normal FA $\downarrow$	Forced FA $\downarrow$	Recovery Rate $\downarrow$
Baseline	0.26	0.37	11%
SEUF	0.25	0.33	8%
GRIP	0.24	0.27	3%

As shown in Table 4, forced expert access recovers 11% of supposedly forgotten queries in baseline methods and 8% under SEUF’s soft regularization, whereas GRIP limits recovery to just 3%, within noise margins. These results confirm that unconstrained and softly-

constrained methods exploit routing manipulation as a shortcut that hides knowledge rather than erasing it (Yang et al., 2026; Xu et al., 2026), whereas GRIP’s hard geometric constraints force genuine parameter-level erasure that withstands even direct white-box expert access. Further, in order to emulate more realistic attack profiles, we conduct black-box attack analysis on GRIP.

Table 5: Black-box prompt attacks on WMDP-Cyber, following the protocol of Jang et al. (2025). GRIP is Method A.

Method	Normal FA↓	5-shot FA↓	Rephrase-as-Poem FA↓	Recovery Rate↓
Baseline	0.26	0.38	0.37	11.5%
SEUF	0.25	0.33	0.34	8.5%
GRIP	0.24	0.28	0.29	4.5%
Dense Baseline	0.25	0.27	0.28	2.5%

5.5 Black-Box Prompt-Attack Robustness

Following Jang et al. (2025), we run a black-box prompt-attack suite on WMDP-Cyber for the unconstrained baseline, SEUF, and GRIP via lm-evaluation-harness, using two prompt variants and reporting a Recovery Rate analogous to Table 4. As shown in Table 5, GRIP’s forget accuracy under attack stays near chance (0.28–0.29) and just above the dense reference, whereas SEUF degrades further (0.33–0.34) and the unconstrained baseline most of all (0.37–0.38), with the Recovery Rate ordering matching. GRIP’s recovery is in line with dense-architecture baselines, indicating that it introduces marginal additional black-box vulnerability while additionally closing the white-box expert-forcing loophole the baselines leave open.

6 Conclusion and Discussion

We identify a structural failure mode in MoE unlearning: standard objectives exploit the router as an optimization shortcut, redirecting forget-set queries away from knowledgeable experts rather than erasing their parameters. GRIP addresses this by projecting router updates into the null space of the retain set’s routing matrix, decoupling routing stability from parameter plasticity and forcing erasure to occur within the expert weights themselves. Across two MoE models and two benchmarks, GRIP restores routing stability from 0.21 to at least 0.94, improves retain accuracy by up to 89%, and reduces white-box adversarial knowledge recovery from 11% to 3% while matching dense-model robustness under black-box prompt attack, at $1.71\times$ training overhead, or $1.21\times$ for the post-training variant. Crucially, GRIP functions as an algorithm-agnostic adapter: it constrains only the router, leaving the underlying unlearning objective unmodified, which means any future dense-architecture unlearning method can be deployed on MoE models by simply wrapping its optimization with GRIP’s geometric projection.

We acknowledge three limitations. First, caching retain representations incurs $O(LdN_r)$ memory overhead, which may require offloading strategies for extremely deep architectures. Second, GRIP optimizes for faithfulness to the original model’s retain-set routing rather than to a retain-only oracle, whose router there is no reason to expect would reproduce the original expert assignments, since the forget data shaped router learning during pretraining; we adopt preservation because the oracle is unattainable at scale and because the preserved routing proves empirically benign for utility, but this assumes the forget set’s influence on the global router is small, and the two criteria may diverge for large or heavily retain-entangled forget sets. Preserving the base routing geometry also inherits any biases in the original router; selectively relaxing constraints for biased experts remains an important direction for future work. Third, our claims are empirical rather than formal: we establish no formal guarantee of routing preservation, though the preservation we measure is near-complete ($RS \geq 0.94$).

Acknowledgements

A. Zhu, R. Wei and P. Li are partially supported by the NSF under awards PHY-2117997, IIS-2239565, IIS-2428777, and CCF-2402816; the JPMorgan Chase Faculty Award; and the OpenAI Researcher Access Program Credit, the Nvidia Academic Award, and the Google Academic Award.

Ethics Statement

This work develops methods for genuine knowledge erasure in MoE language models, motivated by privacy regulations such as GDPR and the need to remove hazardous or copyrighted content from deployed systems. Our experiments use two publicly available benchmarks (WMDP and MUSE) following their original evaluation protocols; we do not generate or distribute hazardous content. The adversarial expert forcing evaluation in Section 5.4 demonstrates that existing methods leave sensitive knowledge recoverable, which we disclose to motivate stronger erasure guarantees rather than to provide an exploitation toolkit; the attack requires full white-box model access, limiting its practical applicability.

We acknowledge that unlearning methods could in principle be misused to selectively remove safety behaviors from models while preserving harmful capabilities. However, we believe that exposing the superficial nature of current MoE unlearning, where knowledge is hidden by routing manipulation rather than erased, poses the greater risk if left unaddressed, and that publishing principled defenses serves the broader safety community by raising the standard for verifiable knowledge removal.

References

- Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report, 2023. URL <https://arxiv.org/abs/2309.16609>.
- Lucas Bourtole, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. *2021 IEEE Symposium on Security and Privacy (SP)*, pp. 141–159, 2021. URL <https://api.semanticscholar.org/CorpusID:208909851>.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfc4967418bfb8ac142f64a-Paper.pdf.
- Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE Symposium on Security and Privacy*, pp. 463–480, 2015. doi: 10.1109/SP.2015.35.
- Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B. Brown, Dawn Xiaodong Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In *USENIX Security Symposium*, 2020. URL <https://api.semanticscholar.org/CorpusID:229156229>.

Jesús A. De Loera, Jamie Haddock, and Deanna Needell. A sampling kaczmarz–motzkin algorithm for linear feasibility. *SIAM Journal on Scientific Computing*, 39(5):S66–S87, 2017.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhua Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanbiao Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. DeepSeek-V3 Technical Report. *arXiv e-prints*, art. arXiv:2412.19437, December 2024. doi: 10.48550/arXiv.2412.19437.

European Parliament and Council of the European Union. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation), 2016. URL <https://eur-lex.europa.eu/eli/reg/2016/679/oj>.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. *J. Mach. Learn. Res.*, 23(1), January 2022. ISSN 1532-4435.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy Lillicrap, Angeliki Lazaridou, Orhan Firat, James Molloy, Michael Isard, Paul R. Barham, Tom Hennigan, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, Ryan Doherty, Eli Collins, Clemens Meyer, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, Jack Krawczyk, Cosmo Du, Ed Chi, Heng-Tze Cheng, Eric Ni, Purvi Shah, Patrick Kane, Betty Chan, Manaal Faruqui, Aliaksei Severyn, Hanzhao Lin, YaGuang Li, Yong Cheng, Abe Ittycheriah, Mahdis Mahdieh, Mia Chen, Pei Sun, Dustin Tran, Sumit Bagri, Balaji Lakshminarayanan, Jeremiah Liu, Andras Orban, Fabian Gura, Hao Zhou, Xinying Song, Aurelien Boffy, Harish Ganapathy, Steven Zheng, HyunJeong Choe, Ágoston Weisz, Tao Zhu, Yifeng Lu, Siddharth Gopal, Jarrod Kahn, Maciej Kula, Jeff Pitman, Rushin Shah, Emanuel Taropa, Majd Al Meray, Martin Baeuml, Zhifeng Chen, Laurent El Shafey, Yujing Zhang, Olcan Sercinoglu, George Tucker, Enrique Piqueras, Maxim Krikun, Iain Barr, Nikolay Savinov, Ivo Danihelka, Becca Roelofs, Anaïs White,

- Anders Andreassen, Tamara von Glehn, Lakshman Yagati, Mehran Kazemi, Lucas Gonzalez, Misha Khalman, Jakub Sygnowski, Alexandre Frechette, Charlotte Smith, Laura Culp, Lev Proleev, Yi Luan, Xi Chen, James Lottes, Nathan Schucher, Federico Lebron, Alban Rustemi, Natalie Clay, Phil Crone, Tomas Kocisky, Jeffrey Zhao, Bartek Perz, Dian Yu, Heidi Howard, Adam Bloniarz, Jack W. Rae, Han Lu, Laurent Sifre, Marcello Maggioni, Fred Alcober, Dan Garrette, Megan Barnes, Shantanu Thakoor, Jacob Austin, Gabriel Barth-Maron, William Wong, Rishabh Joshi, Rahma Chaabouni, Deeni Fatiha, Arun Ahuja, Gaurav Singh Tomar, Evan Senter, Martin Chadwick, Ilya Kornakov, Nithya Attaluri, Iñaki Iturrate, Ruibo Liu, Yunxuan Li, Sarah Cogan, Jeremy Chen, Chao Jia, Chenjie Gu, Qiao Zhang, Jordan Grimstad, Ale Jakse Hartman, Xavier Garcia, Thanumalayan Sankaranarayanan Pillai, Jacob Devlin, Michael Laskin, Diego de Las Casas, Dasha Valter, Connie Tao, Lorenzo Blanco, Adrià Puigdomènech Badia, David Reitter, Mianna Chen, Jenny Brennan, Clara Rivera, Sergey Brin, Shariq Iqbal, Gabriela Surita, Jane Labanowski, Abhi Rao, Stephanie Winkler, Emilio Parisotto, Yiming Gu, Kate Olszewska, Ravi Addanki, Antoine Miech, Annie Louis, Denis Teplyashin, Geoff Brown, Elliot Catt, Jan Balaguer, Jackie Xiang, Pidong Wang, Zoe Ashwood, Anton Briukhov, Albert Webson, Sanjay Ganapathy, Smit Sanghavi, Ajay Kannan, Ming-Wei Chang, Axel Stjerngren, Josip Djolonga, Yuting Sun, Ankur Bapna, Matthew Aitchison, Pedram Pejman, Henryk Michalewski, Tianhe Yu, Cindy Wang, Juliette Love, Junwhan Ahn, Dawn Bloxwich, Kehang Han, Peter Humphreys, Thibault Sellam, James Bradbury, Varun Godbole, Sina Samangooei, Bogdan Damoc, and Alex Kaskasoli. Gemini: A Family of Highly Capable Multimodal Models. *arXiv e-prints*, art. arXiv:2312.11805, December 2023. doi: 10.48550/arXiv.2312.11805.
- Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal sunshine of the spotless net: Selective forgetting in deep networks. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9301–9309, 2020. URL <https://api.semanticscholar.org/CorpusID:207863297>.
- Suchin Gururangan, Margaret Li, Mike Lewis, Weijia Shi, Tim Althoff, Noah A. Smith, and Luke Zettlemoyer. Scaling expert language models with unsupervised domain discovery. *ArXiv*, abs/2303.14177, 2023. URL <https://api.semanticscholar.org/CorpusID:257756896>.
- Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022.
- Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. Knowledge unlearning for mitigating privacy risks in language models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14389–14408, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.805. URL <https://aclanthology.org/2023.acl-long.805/>.
- Yeonwoo Jang, Shariqah Hossain, Ashwin Sreevatsa, and Diogo Cruz. Prompt attacks reveal superficial knowledge removal in unlearning methods, 2025. URL <https://arxiv.org/abs/2506.10236>.
- Jinghan Jia, Yihua Zhang, Yimeng Zhang, Jiancheng Liu, Bharat Runwal, James Diefenderfer, Bhavya Kailkhura, and Sijia Liu. SOUL: Unlocking the power of second-order optimization for LLM unlearning. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 4276–4292, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.245. URL <https://aclanthology.org/2024.emnlp-main.245/>.
- Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra S Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.
- Tomasz Korbak, Kejian Shi, Angelica Chen, Rasika Bhalerao, Christopher L. Buckley, Jason Phang, Samuel R. Bowman, and Ethan Perez. Pretraining language models with human

- preferences. In *Proceedings of the 40th International Conference on Machine Learning, ICML'23*. JMLR.org, 2023.
- Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. Towards unbounded machine unlearning. In *Proceedings of the 37th International Conference on Neural Information Processing Systems, NIPS '23*, Red Hook, NY, USA, 2023. Curran Associates Inc.
- Dennis Leventhal and Adrian S. Lewis. Randomized methods for linear constraints: Convergence rates and conditioning. *Mathematics of Operations Research*, 35(3):641–654, 2010.
- Nathaniel Li, Alexander Pan, Anjali Gopal, Summer Yue, Daniel Berrios, Alice Gatti, Justin D. Li, Ann-Kathrin Dombrowski, Shashwat Goel, Gabriel Mukobi, Nathan Helm-Burger, Rassan Lababidi, Lennart Justen, Andrew Bo Liu, Michael Chen, Isabelle Barrass, Oliver Zhang, Xiaoyuan Zhu, Rishub Tamirisa, Bhruhu Bharathi, Ariel Herbert-Voss, Cort B Breuer, Andy Zou, Mantas Mazeika, Zifan Wang, Palash Oswal, Weiran Lin, Adam Alfred Hunt, Justin Tienken-Harder, Kevin Y. Shih, Kemper Talley, John Guan, Ian Steneker, David Campbell, Brad Jokubaitis, Steven Basart, Stephen Fitz, Ponnuram Kumaraguru, Kallol Krishna Karmakar, Uday Tupakula, Vijay Varadharajan, Yan Shoshitaishvili, Jimmy Ba, Kevin M. Esvelt, Alexandr Wang, and Dan Hendrycks. The WMDP benchmark: Measuring and reducing malicious use with unlearning. In *Forty-first International Conference on Machine Learning*, 2024. URL <https://openreview.net/forum?id=xlr6AUDuJz>.
- 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, 2022.
- Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. Fast model editing at scale. In *International Conference on Learning Representations*, 2022. URL <https://openreview.net/forum?id=0DcZxeWfOPT>.
- Milad Nasr, Javier Rando, Nicholas Carlini, Jonathan Hayase, Matthew Jagielski, A. Feder Cooper, Daphne Ippolito, Christopher A. Choquette-Choo, Florian Tramèr, and Katherine Lee. Scalable extraction of training data from aligned, production language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=vjel3nWP2a>.
- OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam

- Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O’Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, and Alec Radford. GPT-4 Technical Report. *arXiv e-prints*, art. arXiv:2303.08774, March 2023. doi: 10.48550/arXiv.2303.08774.
- Martin Pawelczyk, Seth Neel, and Himabindu Lakkaraju. In-context unlearning: Language models as few shot unlearners. *arXiv preprint arXiv:2310.07579*, 2023.
- Qwen Team. Qwen3 technical report, 2025. URL <https://arxiv.org/abs/2505.09388>.
- Weijia Shi, Jaechan Lee, Yangsibo Huang, Sathika Malladi, Jieyu Zhao, Ari Holtzman, Daogao Liu, Luke Zettlemoyer, Noah A. Smith, and Chiyuan Zhang. MUSE: Machine unlearning six-way evaluation for language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=TArMA033BU>.
- Thomas Strohmer and Roman Vershynin. A randomized kaczmarz algorithm with exponential convergence. *Journal of Fourier Analysis and Applications*, 15(2):262–278, 2009. doi: 10.1007/s00041-008-9030-4.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023. URL <https://arxiv.org/abs/2307.09288>.
- Eduard Fosch Villaronga, Peter Kieseberg, and Tiffany Li. Humans forget, machines remember: Artificial intelligence and the right to be forgotten. *Computer Law Security Review*, 34(2):304–313, 2018. ISSN 2212-473X. doi: <https://doi.org/10.1016/j.clsr.2017.08.007>. URL <https://www.sciencedirect.com/science/article/pii/S0267364917302091>.
- Rongzhe Wei, Mufei Li, Mohsen Ghassemi, Eleonora Kreačić, Yifan Li, Xiang Yue, Bo Li, Vamsi K Potluru, Pan Li, and Eli Chien. Underestimated privacy risks for minority populations in large language model unlearning. *arXiv preprint arXiv:2412.08559*, 2024.
- Laura Weidinger, John F. J. Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, Zachary Kenton, Sande Minnich Brown, William T. Hawkins, Tom Stepleton, Courtney Biles, Abeba Birhane, Julia Haas, Laura Rimell, Lisa Anne Hendricks, William S. Isaac, Sean Legassick, Geoffrey Irving, and Iason Gabriel. Ethical and social risks of harm from language models. *ArXiv*, abs/2112.04359, 2021. URL <https://api.semanticscholar.org/CorpusID:244954639>.
- Xinwei Wu, Junzhuo Li, Minghui Xu, Weilong Dong, Shuangzhi Wu, Chao Bian, and Deyi Xiong. Depn: Detecting and editing privacy neurons in pretrained language models. In

- Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 2875–2886, 2023.
- Xiaoyu Xu, Xiang Yue, Yang Liu, Qingqing Ye, Huadi Zheng, Peizhao Hu, Minxin Du, and Haibo Hu. Unlearning isn’t deletion: Investigating reversibility of machine unlearning in LLMs. In *The Fourteenth International Conference on Learning Representations*, 2026. URL <https://openreview.net/forum?id=7cEMkTu7Lf>.
- Nakyeong Yang, Dong-Kyum Kim, Jea Kwon, Minsung Kim, Kyomin Jung, and Meeyoung Cha. Erase or hide? suppressing spurious unlearning neurons for robust unlearning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL <https://openreview.net/forum?id=z2zFk9jYpw>.
- Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. Editing large language models: Problems, methods, and opportunities. In Houada Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 10222–10240, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.632. URL <https://aclanthology.org/2023.emnlp-main.632/>.
- Chiyuan Zhang, Daphne Ippolito, Katherine Lee, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. Counterfactual memorization in neural language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=67o9UQgTD0>.
- Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. In *First Conference on Language Modeling*, 2024. URL <https://openreview.net/forum?id=MXLBXjQkmb>.
- Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Y. Zhao, Andrew Dai, Zhifeng Chen, Quoc Le, and James Laudon. Mixture-of-experts with expert choice routing. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS ’22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.
- Haomin Zhuang, Yihua Zhang, Kehan Guo, Jinghan Jia, Gaowen Liu, Sijia Liu, and Xi-angliang Zhang. SEUF: Is unlearning one expert enough for mixture-of-experts LLMs? In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8664–8678, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.424. URL <https://aclanthology.org/2025.acl-long.424/>.
- Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. St-moe: Designing stable and transferable sparse expert models, 2022. URL <https://arxiv.org/abs/2202.08906>.

A Implementation Details

Our experiments were conducted using the PyTorch framework, leveraging Hugging Face’s transformers and accelerate libraries for efficient model handling and distributed training.

Model Architecture We use **Qwen3-30B-A3B** (Qwen Team, 2025), a Mixture-of-Experts member of the Qwen3 family with $L = 48$ layers, hidden dimension $d = 2048$, and $E = 128$ experts per MoE layer under a top-8 routing strategy. All concrete figures in Appendices B–D use these values. The model has approximately 3 billion active parameters during inference.

Hardware and Quantization All unlearning procedures were executed on a single node equipped with 2 NVIDIA H200 141GB GPUs and a single node equipped with 8 NVIDIA L40S 48GB GPUs. To maximize throughput and reduce the memory footprint, we employed FP8 precision for both weights and activations during the unlearning steps. Specifically, we utilized the E4M3 floating-point format, which is well-suited for training stability, and

integrated NVIDIA’s TransformerEngine library to handle the FP8 mixed-precision logic seamlessly.

Hyperparameters The unlearning process for our proposed method and all baselines was configured with the hyperparameters listed in Table 6, chosen via a preliminary sweep on a held-out validation set.

Table 6: Hyperparameters for all unlearning experiments.

Hyperparameter	Value
Optimizer	AdamW
Learning Rate	3×10^{-5}
Batch Size (per GPU)	2
Adam β_1	0.9
Adam β_2	0.999
Adam ϵ	1×10^{-8}
Weight Decay	0.01
Maximum Unlearning Steps	10000

A.1 Evaluation Metric Details

WMDP Metrics. We follow the evaluation protocol of Li et al. (2024). All models are evaluated on the WMDP-Cyber multiple-choice question answering (MCQA) benchmark, which tests knowledge of hazardous cybersecurity content.

- **Forget Accuracy (FA):** Accuracy on MCQA questions derived from the hazardous cybersecurity corpus. A perfectly unlearned model should perform at random-chance level (≈ 0.25 for 4-way MCQA). Lower is better.
- **Retain Accuracy (RA):** Accuracy on the MMLU general knowledge MCQA benchmark, measuring preservation of broad model capabilities unrelated to the forget set. Higher is better (approaching the original MoE backbone’s score of ≈ 0.70).

Controlled Evaluation Protocol. To isolate the effect of routing constraints on utility preservation, we control for forgetting level: all methods are trained until their forget metrics (FA for WMDP; C1 and C2 for MUSE) match the unconstrained baseline as closely as the base objective permits, and we compare retain-side metrics at that level. This prevents methods from appearing superior simply by under-unlearning. We report C3 for fidelity to the full MUSE evaluation of Shi et al. (2025) rather than because our claims rest on it; being two-sided, it is not used as a stopping criterion, and no comparison in this paper is drawn from it. The match is close for GD, KL and RMU. It is not for NPO: the unconstrained baseline and SEUF plateau at FA 0.33–0.34 and cannot be driven lower, so the GRIP rows of that block in Table 1 are compared against the lowest forget accuracy those two methods reach rather than against an exactly matched value. The same caveat applies to the combined-constraint row of Table 9.

MUSE Metrics. We adhere to the evaluation framework of Shi et al. (2025), which assesses unlearning from both the data owner’s and model deployer’s perspectives. We evaluate on the News corpus. The retrained reference used to normalize C3 is a retain-only finetune of the same backbone under identical hyperparameters, and all C3 values are reported relative to this same-architecture reference rather than to the dense retrained checkpoints released with the benchmark.

- **C1: Verbatim Memorization (VerbMem):** The model is prompted with the first l tokens of a sequence from the forget set, and the ROUGE-L F1 score is computed between the model’s continuation and the true continuation. This measures the model’s tendency to reproduce copyrighted text verbatim. Lower is better.
- **C2: Knowledge Memorization (KnowMem):** For each example in the forget set, a question-answer pair (q, a) is constructed. The model generates a response to q , and the ROUGE-L score between the generated response and the ground-truth answer

a is averaged across all pairs. This measures retention of factual knowledge from the forget set. Lower is better.

- **C3: Privacy Leakage (PrivLeak):** Quantifies whether the unlearned model leaks membership information about the forget set. We apply the Min-K% Prob membership inference attack following the MUSE protocol (Shi et al., 2025) and compute the AUC-ROC score for distinguishing forget-set members from holdout non-members, normalized against the retrained model’s AUC. A value near zero indicates the unlearned model is indistinguishable from a model that was never trained on the forget set; large positive or negative values indicate under- or over-unlearning respectively. Closer to zero is better, so C3 is a two-sided diagnostic rather than a quantity to be minimized; this is why Table 1 marks no best value in that column. Every value we obtain is negative, as is the dense reference, indicating that all methods including the dense baseline sit on the over-unlearning side at the forgetting levels compared here.
- **C4: Utility Preservation (KnowMem Retain):** Identical to the C2 metric but evaluated on question-answer pairs derived from the retain set. This measures whether unlearning degrades the model’s knowledge of content that should be preserved. Higher is better (approaching the original model’s retain-set performance). On the GD objective every method, constrained or not, collapses C4 (4.31–11.4 against 44.6 for the dense reference), which is why the C4 comparisons of Section 5 are scoped to KL, NPO and RMU.

Routing Stability (RS). In addition to the above benchmark-specific metrics, we report routing stability as defined in Equation (2) for all experiments. RS is computed over the retain set D_r and measures the Jaccard similarity between pre- and post-unlearning expert selections, averaged across all evaluation queries and all MoE layers. An RS of 1.0 indicates that routing decisions are perfectly preserved; values below 0.5 indicate that a substantial fraction of expert assignments have changed. Under the top-8 routing of our backbone an RS of 0.5 corresponds to roughly three of the eight selected experts differing, and half the assignments change only at an RS of about 0.33.

Dense Baselines. The bottom row of Table 1 reports results from the original WMDP and MUSE papers evaluated on dense architectures: Zephyr-7B-beta for WMDP (Li et al., 2024) and LLaMA-2-7B for MUSE (Shi et al., 2025). These serve as reference points for the utility ceiling achievable by dense-architecture unlearning methods, against which we benchmark GRIP’s recovery of MoE utility.

A.2 Unlearning Baseline Formulations

To ensure fair comparison, we standardized the loss functions for all baseline methods used in Section 5. Let $f_\theta(x)$ denote the model output and $\mathcal{L}_{CE}$ denote the cross-entropy loss.

Each objective pairs a forget-side term with a retain-side term of weight α . The forget-side term is what distinguishes them, except for KL, which instead substitutes a divergence anchor on the retain side.

Gradient Difference (GD). We maximize the loss on the forget set D_f while minimizing loss on the retain set D_r (simulated via gradient accumulation or replay).

$$\mathcal{L}_{GD} = -\mathcal{L}_{CE}(f_\theta(x_f), y_f) + \alpha \mathcal{L}_{CE}(f_\theta(x_r), y_r) \quad (6)$$

KL Minimization (KL). We apply the same forget-side ascent, but replace the retain-side cross-entropy with a KL-divergence anchor to a fixed reference model (the initial pre-trained model θ_{ref}), holding retain-set behavior close to the original model rather than only to its labels.

$$\mathcal{L}_{KL} = -\mathcal{L}_{CE}(f_\theta(x_f), y_f) + \alpha \text{KL}(f_{\theta_{ref}}(x_r) \parallel f_\theta(x_r)) \quad (7)$$

Negative Preference Optimization (NPO). Following Zhang et al. (2024), we treat the forget data as a negative preference sample. The objective drives the forget likelihood below that

of the reference model while remaining bounded, avoiding the divergence of unbounded ascent:

$$\mathcal{L}_{NPO} = -\frac{2}{\beta} \log \sigma \left(-\beta \log \frac{\pi_\theta(y_f|x_f)}{\pi_{ref}(y_f|x_f)} \right) + \alpha \mathcal{L}_{CE}(f_\theta(x_r), y_r) \quad (8)$$

where β controls the sharpness of the penalty; as $\beta \rightarrow 0$ the forget-side term recovers gradient ascent.

Representation Misdirection (RMU). Following Li et al. (2024), we perturb the internal activations $h_\ell(x_f)$ at layer ℓ toward a scaled random target vector v_{rand} , preventing the extraction of semantic knowledge, while anchoring retain activations to those of the frozen reference model h_ℓ^{ref} :

$$\mathcal{L}_{RMU} = \|h_\ell(x_f) - c v_{rand}\|_2^2 + \alpha \|h_\ell(x_r) - h_\ell^{ref}(x_r)\|_2^2 \quad (9)$$

where c is the activation scaling coefficient of Li et al. (2024).

A.3 Performance on General Utility Benchmarks

Beyond the retain-set accuracy reported as RA in Table 1, which follows the few-shot WMDP harness configuration of Li et al. (2024), we separately verify that GRIP preserves broad model capabilities by evaluating *zero-shot* performance on MMLU and TruthfulQA. The two MMLU figures are measured under different protocols, so their retention ratios are not directly comparable. GRIP-unlearned models retain 86–90% of the original model’s MMLU score and 88–89% on TruthfulQA, significantly outperforming unconstrained base-lines, which suffer catastrophic utility collapse and retain only around 66% and 69% for MMLU and TruthfulQA respectively. Together with the retain-set and routing stability gains above, these results establish that GRIP’s constraints preserve model quality at multiple granularities.

B Null Space Projection Construction

We provide the detailed construction of the null space projector $P_{j,\ell}^{eq}$ used in Section 4.2.

B.1 Eigen-decomposition Method

Given the retain representations $X_{j,\ell}^{eq} \in \mathbb{R}^{d \times |\mathcal{I}_{j,\ell}|}$ that selected expert j at layer ℓ , we compute the covariance matrix:

$$C_{j,\ell} = X_{j,\ell}^{eq} X_{j,\ell}^{eq\top} \in \mathbb{R}^{d \times d} \quad (10)$$

We perform eigendecomposition of $C_{j,\ell}$:

$$C_{j,\ell} = U_{j,\ell} \Lambda_{j,\ell} U_{j,\ell}^\top \quad (11)$$

where $U_{j,\ell} = [u_1, u_2, \dots, u_d] \in \mathbb{R}^{d \times d}$ contains orthonormal eigenvectors and $\Lambda_{j,\ell} = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_d)$ contains eigenvalues ordered as $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_d \geq 0$.

The null space of $X_{j,\ell}^{eq\top}$ corresponds to eigenvectors with zero (or near-zero) eigenvalues. In practice, we use a threshold $\epsilon_{\text{null}} = 10^{-2}$ to identify null space eigenvectors:

$$\hat{U}_{j,\ell} = [u_i : \lambda_i < \epsilon_{\text{null}}] \in \mathbb{R}^{d \times m} \quad (12)$$

where $m = |\{i : \lambda_i < \epsilon_{\text{null}}\}|$ is the dimension of the approximate null space. The null space projector is then:

$$P_{j,\ell}^{eq} = \hat{U}_{j,\ell} \hat{U}_{j,\ell}^\top \quad (13)$$

For any router row update $u_{j,\ell} \in \mathbb{R}^d$, the projected update $\tilde{u}_{j,\ell} = P_{j,\ell}^{eq} u_{j,\ell}$ satisfies $\tilde{u}_{j,\ell}^\top X_{j,\ell}^{eq} \approx 0$ within numerical precision.

B.2 Parameterization for Unconstrained Optimization

The null space projector can be used to reparameterize router updates. Instead of optimizing $\Delta\Theta_\ell$ directly with constraints, we can optimize an unconstrained matrix $W_\ell \in \mathbb{R}^{E \times d}$ and set:

$$(\Delta\Theta_\ell)_{j,:} = P_{j,\ell}^{\text{eq}}(W_\ell)_{j,:} \quad (14)$$

This ensures each router row update resides in $\text{Null}(X_{j,\ell}^{\text{eq}\top})$ by construction, eliminating the need for explicit constraint enforcement. It enforces only the Stage 1 equality constraints of Section 4.2, discarding the Stage 2 freedom to move non-selected scores within their margins, and is therefore more restrictive than the formulation we use, though still less restrictive than the global constraint $\Delta\Theta_\ell X_{r,\ell} = 0$ of Section 4.1.

B.3 Computational Complexity

The eigendecomposition of $C_{j,\ell} \in \mathbb{R}^{d \times d}$ requires $\mathcal{O}(d^3)$ operations. Because the retain representations drift as unlearning proceeds (Section 4.3), the projector is in principle a per-step quantity, and re-forming it from scratch at every step would cost $\mathcal{O}(KLEd^3)$. Our implementation instead performs the full decomposition once per expert and layer before unlearning begins and thereafter edits the cached retain-subspace basis incrementally, so the recurring per-step cost is that of the low-rank update and its application rather than of a fresh decomposition. Storing $\hat{U}_{j,\ell} \in \mathbb{R}^{d \times m}$ requires $\mathcal{O}(dm)$ memory, where $m = d - r_{j,\ell}$ and $r_{j,\ell} = \text{rank}(X_{j,\ell}^{\text{eq}}) \leq |\mathcal{I}_{j,\ell}|$ is the rank of the retain subspace that expert j actually serves. Because top- k routing selects each expert for only a small fraction of retain inputs, $r_{j,\ell} \ll d$ and hence m is close to d ; Appendix D exploits this by storing the small retain-subspace basis instead.

Applying the projection to a router row update $u_{j,\ell} \in \mathbb{R}^d$ requires:

$$\tilde{u}_{j,\ell} = \hat{U}_{j,\ell}(\hat{U}_{j,\ell}^\top u_{j,\ell}) \in \mathcal{O}(dm) \quad (15)$$

For our configuration $E = 128$, $d = 2048$, $m \approx 2000$, this is approximately 4.1×10^6 operations per expert per projection step.

C Randomized Kaczmarz for Halfspace Projection

We employ the Randomized Kaczmarz (RK) algorithm to efficiently enforce the inequality constraints described in Section 4.2. Unlike standard cyclic projections, RK samples constraints with probability proportional to their geometric “importance” (squared Euclidean norms), leading to faster convergence rates.

C.1 Problem Formulation

For expert j at layer ℓ , we must find an update $\tilde{u}_{j,\ell} \in \mathbb{R}^d$ that satisfies the set of linear inequalities for the non-selected inputs ($i \notin \mathcal{I}_{j,\ell}$):

$$\langle \tilde{u}_{j,\ell}, x_{r,\ell}^{(i)} \rangle \leq \tau_{i,j,\ell} - \varepsilon \quad (16)$$

where $\tau_{i,j,\ell}$ is the margin threshold defined in Equation (3). Let X_{ineq} be the matrix whose rows are the input vectors $x_{r,\ell}^{(i)}$ for all $i \notin \mathcal{I}_{j,\ell}$, with corresponding thresholds $\tau_{i,j,\ell} - \varepsilon$.

While the classic Kaczmarz method sweeps through rows cyclically, the Randomized Kaczmarz algorithm selects a constraint index i at iteration k with probability p_i :

$$p_i = \frac{\|x_{r,\ell}^{(i)}\|_2^2}{\|X_{\text{ineq}}\|_F^2} \quad (17)$$

where $\|X_{\text{ineq}}\|_F^2 = \sum_{i \notin \mathcal{I}_{j,\ell}} \|x_{r,\ell}^{(i)}\|_2^2$ is the squared Frobenius norm of the constraint matrix.

C.2 Algorithm

Algorithm 1 details the procedure. We precompute the row norms and sampling probabilities. At each step, we sample a constraint; if the constraint is violated, we project the current router row update orthogonally onto the defining hyperplane.

Algorithm 1 GRIP Training-Time Enforcement via Randomized Kaczmarz

Require: Unconstrained router row updates $\{u_{j,\ell}\}$, retain representations $X_{r,\ell}$, expert selections $\{\mathcal{I}_{j,\ell}\}$, thresholds $\{\tau_{i,j,\ell}\}$

Ensure: Constrained updates $\{\tilde{u}_{j,\ell}\}$

- 1: **for** each layer $\ell = 1, \dots, L$ **do**
- 2: **for** each expert $j = 1, \dots, E$ **do**
- 3: {— Phase 1: Equality Constraints (Null Space) —}
- 4: $X_{j,\ell}^{\text{eq}} \leftarrow [x_{r,\ell}^{(i)}]_{i \in \mathcal{I}_{j,\ell}}$
- 5: $P_{j,\ell}^{\text{eq}} \leftarrow I - X_{j,\ell}^{\text{eq}}(X_{j,\ell}^{\text{eq}}^\top X_{j,\ell}^{\text{eq}})^+ X_{j,\ell}^{\text{eq}\top}$ {cached, updated incrementally (App. B)}
- 6: $\tilde{u}_{j,\ell}^{(0)} \leftarrow P_{j,\ell}^{\text{eq}} u_{j,\ell}$
- 7: {— Phase 2: Inequality Constraints (Randomized Kaczmarz) —}
- 8: Let X_{ineq} be the matrix of rows $x_{r,\ell}^{(i)}$ for $i \notin \mathcal{I}_{j,\ell}$
- 9: Compute row norms $w_i = \|x_{r,\ell}^{(i)}\|_2^2$
- 10: Compute probabilities $p_i = w_i / \sum_{i' \notin \mathcal{I}_{j,\ell}} w_{i'}$
- 11: $k \leftarrow 0$
- 12: **while** $k < k_{\max}$ **do**
- 13: Sample index i from distribution p
- 14: $v_{i,j,\ell} \leftarrow (\tilde{u}_{j,\ell}^{(k)})^\top x_{r,\ell}^{(i)} - (\tau_{i,j,\ell} - \epsilon)$ {Compute Residual}
- 15: **if** $v_{i,j,\ell} > 0$ **then**
- 16: $\tilde{u}_{j,\ell}^{(k+1)} \leftarrow \tilde{u}_{j,\ell}^{(k)} - \frac{v_{i,j,\ell}}{w_i} \cdot x_{r,\ell}^{(i)}$
- 17: **else**
- 18: $\tilde{u}_{j,\ell}^{(k+1)} \leftarrow \tilde{u}_{j,\ell}^{(k)}$
- 19: **end if**
- 20: $k \leftarrow k + 1$
- 21: {Optional: Early exit if converged}
- 22: **end while**
- 23: $\tilde{u}_{j,\ell} \leftarrow \tilde{u}_{j,\ell}^{(k)}$
- 24: **end for**
- 25: **end for**
- 26: **return** $\{\tilde{u}_{j,\ell}\}$

C.3 Convergence Analysis

The classical randomized Kaczmarz analysis of [Strohmer & Vershynin \(2009\)](#) concerns *consistent linear equality systems*, whereas $\mathcal{K}_{j,\ell}$ is a mixed system of equalities and inequalities; that result therefore does not by itself cover the projection we perform. [Leventhal & Lewis \(2010\)](#) extend randomized iterated projection precisely to systems of linear equalities and inequalities, bounding the linear rate of convergence in expectation by the Hoffman constant of the system, and the Sampling Kaczmarz–Motzkin family ([De Loera et al., 2017](#)) establishes comparable rates for linear feasibility. It is those results we rely on, and because their analysis admits mixed systems, a single statement covers both stages of our procedure rather than Stage 2 alone.

We state the guarantee as a *per-step* claim. Within a single optimization step the retain representations $X_{r,\ell}$, active-expert sets $\mathcal{I}_{j,\ell}$, and margins $\tau_{i,j,\ell}$ are fixed, so $\mathcal{K}_{j,\ell}$ in Equation (4)

is a fixed intersection of half-spaces and the system solved is a fixed, consistent inequality system, exactly the setting the cited results assume.

Theorem C.1 (Per-step convergence of the Stage 2 projection). *Fix an optimization step, holding $X_{r,\ell}$, $\mathcal{I}_{j,\ell}$, and $\{\tau_{i,j,\ell}\}$ fixed, so that $\mathcal{K}_{j,\ell}$ is a fixed nonempty polyhedron. Then the iterates of Algorithm 1 satisfy*

$$\mathbb{E}[\text{dist}(\tilde{u}_{j,\ell}^{(k)}, \mathcal{K}_{j,\ell})^2] \leq \left(1 - \frac{1}{H^2 \|X_{\text{ineq}}\|_F^2}\right)^k \text{dist}(\tilde{u}_{j,\ell}^{(0)}, \mathcal{K}_{j,\ell})^2 \quad (18)$$

where $\text{dist}(\cdot, \mathcal{K}_{j,\ell})$ is the Euclidean distance to the feasible set, X_{ineq} collects the retain representations defining the constraints, and H is the Hoffman constant of that system, written H rather than L to avoid collision with the layer count.

Remark C.2 (Feasible point, not nearest point). The bound is on the distance to $\mathcal{K}_{j,\ell}$, not to any particular element of it. Unlike the equality case, where randomized Kaczmarz converges to the Euclidean projection of the starting iterate onto the solution set, projection onto an intersection of half-spaces converges to *some* feasible update, which need not be the projection of $\tilde{u}_{j,\ell}^{(0)}$. The procedure therefore returns a router update that satisfies the routing constraints, not the minimum-norm modification of the base algorithm’s update; we do not rely on the latter anywhere in the method.

Remark C.3 (Scope of Theorem C.1). The statement is within-step only. Across optimization steps the feasible region itself moves, since upstream updates shift $X_{r,\ell}$ and hence both the active sets and the margins, so the procedure solves a sequence of related but distinct feasibility problems. We do not claim the per-step rate transfers to this non-stationary setting without further conditions, and we make no formal claim about the outer optimization trajectory; this is part of what motivates Method B, which defers enforcement until the representations have stopped moving.

Within a step, the linear rate $\mathcal{O}(c^k)$, as against the $\mathcal{O}(1/\sqrt{k})$ behavior of cyclic projections, is what allows a small fixed iteration budget ($k_{\max} \approx 100$) to drive the residual violation low even when the number of retain inputs N_r is large.

C.4 Computational Complexity

The computational cost is dominated by the probability precomputation and the iterative updates:

- **Precomputation:** Computing squared norms for all non-selected inputs takes $\mathcal{O}(d \cdot N_{\text{ineq}})$. This is done once per batch.
- **Per Iteration:** Sampling takes $\mathcal{O}(\log N_{\text{ineq}})$ or $\mathcal{O}(1)$ with aliasing methods. The update step involves a single dot product and vector addition: $\mathcal{O}(d)$.
- **Total Complexity:** $\mathcal{O}(dN_{\text{ineq}} + k_{\max}d)$.

Since $k_{\max}$ is a small constant independent of the system size (due to the linear convergence rate), the marginal cost of enforcement scales linearly with the dimension d , which is computationally efficient at the router hidden size of our backbone ($d = 2048$).

C.5 Time Complexity of PTC versus Training-Time Enforcement

Time Complexity Efficiency. The computational advantage of Post-Training Correction (PTC) over Training-Time Enforcement (TTE) stems from the frequency of projection operations:

- **TTE Cost:** Requires projecting updates at every one of the K unlearning steps, for every layer and expert. Re-forming the projectors would scale as $\mathcal{O}(K \cdot L \cdot E \cdot d^3)$ with $K \approx 10,000$; the incremental scheme of Appendix B avoids the repeated decomposition, but a per-step projection cost remains.

- **PTC Cost:** Requires a single analytical correction per layer after training. Total cost scales as $O(1 \cdot L \cdot d^3)$.

Since $K \gg 1$, PTC reduces the projection term by approximately four orders of magnitude. That term is not the dominant cost of an unlearning step in absolute terms, since backpropagation through the model and the optimizer update are larger; it is dominant only within the overhead that constraint enforcement adds. Removing it therefore does not return the total to $1.0\times$, but lowers it from $1.71\times$ to the $1.21\times$ reported in Section 4.3, the residual being representation extraction and the FP32 solve.

D Efficient Implementation and Storage

Enforcing geometric constraints on large-scale Mixture-of-Experts (MoE) models presents a unique systems challenge. While the parameter updates are sparse, the constraint matrices are dense, layer-specific, and expert-specific. Storing a dense $d \times d$ projector for every expert across all layers would require prohibitive memory. For our 30B backbone ($d = 2048$, $E = 128$, $L = 48$), naive storage would require $d^2EL \approx 2.6 \times 10^{10}$ entries, roughly 52 GB in FP16 and over 100 GB in FP32. To resolve this, we implement a *Tiered Hierarchical Caching Strategy*, reducing the active memory footprint to $< 1\%$ of the model size without stalling the training pipeline.

D.1 Implicit Low-Rank Storage

Instead of instantiating and storing the full dense projector $P_{j,\ell}^{\text{eq}} \in \mathbb{R}^{d \times d}$, we exploit the spectral properties of the router representations. Because top- k routing selects any given expert for only a small fraction of retain inputs, the retain subspace that expert j serves is effectively low-rank: $r_{j,\ell} \ll d$, where $r_{j,\ell} = \text{rank}(X_{j,\ell}^{\text{eq}})$. We therefore store the orthonormal basis of that subspace, $U_{j,\ell}^{\text{ret}} \in \mathbb{R}^{d \times r_{j,\ell}}$, rather than the much larger null-space basis, and apply the projector implicitly through its complement form:

$$P_{j,\ell}^{\text{eq}} u_{j,\ell} = u_{j,\ell} - U_{j,\ell}^{\text{ret}} (U_{j,\ell}^{\text{ret}\top} u_{j,\ell}) \quad (19)$$

Three expressions for the same operator appear in this paper and are worth separating. The pseudo-inverse form of Section 4.2 is the exact projector, against which the method is defined. The thresholded eigendecomposition of Appendix B is how that projector is computed, and coincides with it up to the truncation at ϵ_{null} , which is the sole discrepancy between the two and the subject of Table 2. Equation (19) is how the computed projector is stored and applied, and is algebraically identical to the eigendecomposition form since $U_{j,\ell}^{\text{ret}}$ and $\hat{U}_{j,\ell}$ span complementary subspaces.

This reduces the storage complexity per expert from $\mathcal{O}(d^2)$ to $\mathcal{O}(d r_{j,\ell})$. For a typical rank $r_{j,\ell} \approx 50$ and hidden dimension $d = 2048$, this yields a compression ratio of $d/r_{j,\ell} \approx 40\times$, allowing the structural constraints for thousands of experts to reside efficiently in system RAM or on disk.

D.2 Hierarchical Caching System

To manage the flow of these constraints during the unlearning optimization, we developed a custom PyTorch autograd hook (MemoryEfficientConstraintHook) that implements a three-tier Least Recently Used (LRU) cache replacement policy:

- **Tier 1: Hot Cache (VRAM).** Stores the retain-subspace bases $U_{j,\ell}^{\text{ret}}$ for the currently active experts in GPU memory, applied in the implicit form of Equation (19) rather than materialized as dense $d \times d$ projectors. Capacity is strictly limited (e.g., $C_{\text{hot}} = 8$ experts) to prevent out-of-memory (OOM) errors during the backward pass.

- **Tier 2: Warm Cache (Host RAM).** Stores compressed basis vectors $U_{j,\ell}^{\text{ret}}$ for frequently accessed experts in CPU memory. This tier serves as a low-latency buffer, avoiding the I/O bottleneck of disk reads.
- **Tier 3: Cold Storage (NVMe/Disk).** The complete set of expert constraints is sharded into individual files on disk. These are loaded lazily only when a “cache miss” occurs in the upper tiers.

D.3 Stochastic Fallback Mechanism

To ensure numerical stability without incurring unnecessary computation, our implementation employs an adaptive verification step. During the backward pass, the hook first applies the Stage 1 equality projection. It then checks the residual violation against the inequality constraints. Only if the violation exceeds a numerical tolerance ($\eta > 10^{-4}$, distinct from the selection margins $\tau_{i,j,\ell}$) does the system trigger the Randomized Kaczmarz solver. This lazy enforcement ensures that the iterative projection is only invoked for the small subset of gradients that actually threaten the routing stability bounds.

D.4 Post-Training Correction: Complexity and Memory Analysis

We provide a detailed complexity analysis of the Post-Training Correction (PTC) procedure described in Section 4.3, covering both computational cost and memory management.

Time Complexity. The dominant cost per layer is the $O(d^3)$ linear solve of Equation (5). Across all L MoE layers, the total PTC cost is $O(L \cdot d^3)$. In contrast, Training-Time Enforcement (Section 4.2) requires a projection at every one of the K unlearning steps and for every expert, yielding $O(K \cdot L \cdot E \cdot d^3)$ if projectors are re-formed rather than updated incrementally. Since $K \approx 10,000$ in our experiments, PTC reduces the projection term by approximately four orders of magnitude. That term is not the dominant cost of training as a whole, which is set by backpropagation and the optimizer; it is the dominant part of the overhead GRIP adds, which is why the reduction appears as $1.71\times \rightarrow 1.21\times$ rather than as a return to $1.0\times$.

Numerical Precision. All intermediate computations in the correction are performed in FP32 regardless of the model’s native precision (FP8 E4M3 in our experiments). This is necessary because the Gram matrix $\tilde{X}_{r,\ell} \tilde{X}_{r,\ell}^\top$ and linear solve are numerically sensitive: the 3-bit mantissa of E4M3 introduces unacceptable accumulation error in the $O(d^2 N_r)$ outer product. The ridge term $\lambda = 10^{-6}$ of Equation (5) is applied in this precision, sized to regularize the solve without perceptibly shifting the restored scores; the acceptance check below verifies this per layer. The final correction Δ_ℓ is cast back to the model’s native dtype before application. We verify each layer’s correction by computing the cosine similarity between original routing scores $\Theta_\ell X_{r,\ell}$ and corrected scores $(\tilde{\Theta}_\ell + \Delta_\ell) \tilde{X}_{r,\ell}$, requiring similarity > 0.95 for acceptance. As noted in Section 4.3, this check doubles as the detector for PTC’s failure mode: a layer that fails it should fall back to Method A.

Memory Footprint. The figures below assume cached tensors are held in FP16 rather than in the model’s native FP8, and the correction arithmetic in FP32. This is a hardware limitation rather than a design choice: E4M3 is exposed only through the TransformerEngine compute path on our hardware, and the host-RAM and disk tiers used for offloading provide no native FP8 storage format, so cached representations and router copies are upcast on eviction. The cache is therefore larger than the model’s own precision would suggest, and the numbers below are the values that actually apply in our setup.

PTC’s memory usage is dominated by three components:

1. **Retain representations** ($2 \times L \times d \times N_r$): Storing both $X_{r,\ell}$ and $\tilde{X}_{r,\ell}$ across all layers. For our 30B model ($L = 48$, $d = 2048$, and retain-set size $N_r = 2048$) in FP16, this

totals approximately $2 \times 48 \times 2048 \times 2048 \times 2B \approx 0.8\text{GB}$. However, because PTC processes layers sequentially, only one layer’s representations need reside in VRAM at a time; the remainder are offloaded to CPU RAM.

2. **Working memory** ($O(d^2)$): The Gram matrix $\tilde{X}_{r,\ell}\tilde{X}_{r,\ell}^\top \in \mathbb{R}^{d \times d}$ of Equation (5) and the linear solve workspace require $2048^2 \times 4B \approx 17\text{MB}$ in FP32 per layer, allocated and freed within each layer’s computation.
3. **Original router weights** ($L \times E \times d$): A copy of each layer’s pre-unlearning gate weights $\Theta_\ell \in \mathbb{R}^{E \times d}$ is required to compute the correction target. For $E = 128$ experts, this totals $48 \times 128 \times 2048 \times 2B \approx 25\text{MB}$ in FP16.

Caching and Amortization. Because the original model’s representations $X_{r,\ell}$ and gate weights Θ_ℓ are independent of the unlearning objective, we cache them to disk after a single extraction pass. Subsequent PTC runs (e.g., with different unlearning algorithms or hyperparameters) skip the original model entirely, requiring only a forward pass through the unlearned model to obtain $\tilde{X}_{r,\ell}$. Cache integrity is validated via metadata checks on the model identifier, retain-set size, sample fingerprint, and layer indices; mismatches trigger automatic recomputation. Individual layer corrections Δ_ℓ are also cached to disk, enabling incremental recomputation when only a subset of layers requires updating. In practice, this caching reduces the wall-clock time of repeated PTC applications from ~ 45 minutes (with original model loading and extraction) to ~ 8 minutes (unlearned model extraction and correction only) on our hardware configuration ($2 \times \text{H200 } 141\text{GB}$).

E Security Evaluation Protocols

To rigorously quantify the robustness of unlearning, we define the formal evaluation algorithm **Adversarial Expert Forcing**, which tests for latent knowledge retention in expert parameters.

E.1 Adversarial Expert Forcing

This protocol simulates a white-box adversary who bypasses the router’s suppression mechanism to directly access erased knowledge; Algorithm 2 states the procedure. Two quantities must be distinguished. The *Adversarial Recovery Rate* (ARR) returned by the algorithm is the raw accuracy of the unlearned model when forced onto the pre-unlearning routing path. The *Recovery Rate* of Table 4 is its increase over the model’s accuracy under its own routing, $\text{ARR} - \text{FA}_{\text{normal}}$, which isolates what forcing recovers rather than what the model answers correctly regardless. The Recovery Rate of Table 5 is the analogous increase, averaged over the two prompt variants of the suite. The query set $\mathcal{D}_{\text{haz}}$ is restricted to forget-set queries whose expert selection actually shifts during unlearning, since forcing is a no-op on the remainder. Both tables report the training-time variant (Method A).

F Direct Comparison with SEUF

To evaluate the efficacy of our framework against the current state-of-the-art in MoE unlearning, we perform a direct head-to-head comparison with SEUF (Zhuang et al., 2025).

Experimental Setup. To ensure a strictly fair comparison, we replicate the specific experimental conditions used in the original SEUF study. We utilize the **Qwen1.5-MoE-A2.7B-Chat** architecture instead of the 30B model used in our main experiments. We evaluate performance on the **WMDP-Cyber** dataset, consistent with the hazardous knowledge removal tasks targeted by both frameworks. We compare both methods when applied to the Representation Misdirection (RMU) unlearning algorithm.

F.1 Results and Analysis

The quantitative results are presented in Table 7. Our analysis highlights three key findings:

Algorithm 2 Adversarial Expert Forcing Evaluation

```

1: Input: Pre-unlearning Model  $\Theta_{pre}$ , Unlearned Model  $\Theta_{post}$ , Hazardous Query Set  $\mathcal{D}_{haz}$ 
2: Input: Normal-routing forget accuracy  $FA_{normal}$  of  $\Theta_{post}$ 
3: Output: ARR, and Recovery Rate  $ARR - FA_{normal}$ 
4:  $N_{correct} \leftarrow 0$ 
5: for each input pair  $(x, y)$  in  $\mathcal{D}_{haz}$  do
6:   {Step 1: Extract original routing path}
7:    $\mathcal{S}_{orig} \leftarrow \text{GetActiveExperts}(\Theta_{pre}, x)$ 
8:   {Step 2: Forward pass with forced routing}
9:   {Override  $\Theta_{post}$  router to strictly use  $\mathcal{S}_{orig}$ }
10:   $\hat{y} \leftarrow \text{Forward}(\Theta_{post}, x, \text{force\_experts} = \mathcal{S}_{orig})$ 
11:  {Step 3: Measure recovery}
12:  if  $\text{argmax}(\hat{y}) = y$  then
13:     $N_{correct} \leftarrow N_{correct} + 1$ 
14:  end if
15: end for
16:  $ARR \leftarrow N_{correct} / |\mathcal{D}_{haz}|$ 
17: return ARR, and the reported Recovery Rate  $ARR - FA_{normal}$ 

```

Table 7: Direct comparison with SEUF on Qwen1.5-MoE-A2.7B-Chat. $\downarrow/\uparrow$ indicates lower/higher is better. Our results are **bolded**.

Method	FA $\downarrow$	RA $\uparrow$	RS $\uparrow$
RMU Baseline	0.28	0.38	0.35
SEUF + RMU	0.25	0.53	0.87
Expert-Specific (GRIP) + RMU	0.25	0.61	0.94
PTC (GRIP) + RMU	0.27	0.63	0.99

Unlearning Efficacy (FA): Both frameworks successfully reduce the model’s accuracy on the forget set to near-random levels. GRIP (Relaxed Constraint) achieves an FA of 0.25, matching SEUF exactly, while PTC achieves 0.27. This confirms that restricting the router’s optimization landscape via null-space projections does not hinder the model’s ability to erase hazardous knowledge.

Superior Utility Preservation (RA): GRIP demonstrates a significant advantage in retaining general model capabilities. While SEUF recovers a Retain Accuracy (RA) of 0.53, GRIP improves this to **0.61** (+15%) with relaxed constraints and **0.63** (+19%) with PTC. SEUF relies on soft regularization penalties (L_2 norms on router weights) to maintain stability. Our results suggest these soft penalties are insufficient to prevent the “optimization shortcut” described in Section 1, leading to degraded utility. In contrast, GRIP’s hard geometric constraints strictly enforce the preservation of retain-set routing, protecting general knowledge more effectively.

Routing Stability (RS): We observe that SEUF improves routing stability over the unconstrained baseline (0.87 vs 0.35) but still permits noticeable drift. GRIP eliminates this drift almost entirely, achieving an RS of **0.94** with training-time constraints and **0.99** with Post-Training Correction. This validates our hypothesis that decoupling routing stability from parameter plasticity is essential for robust MoE unlearning.

G Additional Ablations

This appendix isolates the individual contributions of GRIP’s design choices. Appendix G.1 reports the frozen-router baseline; Appendix G.2 separates the hard-versus-soft constraint axis from the single-versus-all-expert update axis; Appendix G.3 compares constraining routing on the retain set against forget-side and combined alternatives. All runs use WMDP-Cyber with the RMU objective on Qwen3-30B-A3B, at the approximately matched forget

accuracy of Section 5; where a configuration cannot be driven to that level, we report the level it reaches and say so in the text.

G.1 Frozen Router Baseline

The most direct baseline against the routing-shortcut hypothesis is to freeze the router entirely, setting $\Delta\Theta_\ell = 0$ and directing all unlearning gradient into the expert MLPs. If immobilizing the router were sufficient, the expert-specific projections and post-hoc correction of Section 4 would not be justified. As reported in Table 3, forget efficacy is unchanged (FA = 0.26, against 0.26 for the unconstrained baseline), but utility collapses: retain accuracy falls to 0.32, at the level of the unconstrained baseline (0.35) and roughly half of GRIP’s 0.65.

The explanation is that freezing router *weights* does not freeze routing *decisions*. Expert selection depends on both the router parameters Θ_ℓ and the representations $X_{r,\ell}$ arriving at layer ℓ , and as Section 4.3 describes, those representations drift ($X_{r,\ell} \rightarrow \tilde{X}_{r,\ell}$) as upstream experts are updated. A frozen-router model therefore still loses routing stability on the retain set, through upstream drift rather than through router updates, and unlike the post-training correction of Equation (5) it has no mechanism for realigning to the drifted representations.

G.2 Factorial Ablation: Constraint Hardness and Update Breadth

GRIP differs from SEUF (Zhuang et al., 2025) along two axes at once: it replaces a soft L_2 penalty on router weights with a hard null-space projection, and it leaves all activated experts free to update rather than restricting updates to the single highest-attributed expert. The comparison in Table 1 varies both together, so it cannot attribute the gain to either. We therefore run the full 2×2 design, in which the soft/single cell reproduces SEUF and the hard/all cell is GRIP.

Table 8: Factorial ablation on WMDP-Cyber (RMU). Constraint: soft L_2 penalty on router weights versus hard null-space projection. Updates: single highest-attributed expert versus all activated experts. The soft/single configuration corresponds to SEUF and hard/all to GRIP; those two rows reproduce Tables 1 and 4. Rec. is the forced-recovery rate under Expert Forcing (Section 5.4), which is the metric that distinguishes update breadth, since FA is controlled by construction.

Constraint	Updates	FA↓	RA↑	RS↑	Rec.↓
Soft (L_2)	Single expert	0.25	0.58	0.89	8%
Soft (L_2)	All experts	0.26	0.47	0.88	7%
Hard (projection)	Single expert	0.27	0.60	0.96	7%
Hard (projection)	All experts	0.24	0.65	0.96	3%

In Table 8, the two diagonal cells are already reported: soft/single (SEUF) reaches RS 0.89, RA 0.58 and 8% forced recovery, while hard/all (GRIP) reaches RS 0.96, RA 0.65 and 3%. The two off-diagonal cells isolate which axis is responsible. Holding update breadth fixed, moving from a soft penalty to a hard projection raises routing stability from 0.88–0.89 to 0.96 in both rows. Holding constraint hardness fixed, broadening updates from a single expert to all activated experts cuts forced recovery from 7% to 3% under the hard constraint while leaving routing stability unchanged at 0.96. The two axes therefore contribute distinctly: constraint hardness is responsible for routing stability and the retain-side utility that follows from it, whereas update breadth is responsible for erasure depth, which forget accuracy alone cannot reveal because it is matched by construction across all four cells.

The two effects are not independent. Broadening updates under a soft penalty degrades retain accuracy (0.58 to 0.47) and barely improves recovery (8% to 7%), because updating all activated experts amplifies exactly the routing drift a soft penalty fails to prevent. Update breadth pays off only once routing is held fixed by a hard constraint, which is the configuration GRIP occupies.

G.3 Choice of Constraint Surface

Section 4 constrains routing on the retain set. Two alternatives are available: constraining forget-set routing, and constraining both. We compare all three at approximately matched forget accuracy. Note that this axis is distinct from the granularity axis of Table 3, which varies how tightly the retain-side constraint is enforced rather than which inputs it is enforced on.

Table 9: Constraint surface on WMDP-Cyber (RMU). Retain-side is GRIP as described in Section 4 and reproduces the Expert-Specific row of Table 3. Unconstrained-baseline reference values are FA 0.26, RA 0.35, RS 0.43.

Constraint surface	FA↓	RA↑	RS↑	Time↓
Forget-side only	0.26	0.41	0.51	1.31×
Combined forget+retain	0.31	0.64	0.96	2.10×
Retain-side (GRIP)	0.24	0.65	0.96	1.71×

As shown in Table 9, constraining forget-set routing yields RA of 0.41, barely above the 0.35 of the unconstrained baseline and far below the 0.65 of retain-side preservation, confirming that preserving forget-side routing protects little utility. The combined constraint degrades forget accuracy to 0.31 at more than double the training cost, mirroring the over-restriction seen in the “Full Null Space” row of Table 3. Retain-side preservation therefore achieves the best trade-off (FA 0.24, RA 0.65, RS 0.96), consistent with the argument in Section 4 that because router rows are shared across forget and retain inputs, a retain-side constraint already limits forget-query rerouting and an explicit forget-side term adds restriction without adding protection.

G.4 Per-Layer Threshold Selection

The sweep in Table 2 applies a single ϵ_{null} globally. Because representation drift accumulates with depth (Section 4.3), the spectrum of $C_{j,\ell}$ that the threshold cuts is not uniform across the network, so the value that best trades routing stability against plasticity need not be the same at every layer. We resolve the sweep by depth to test this.

Protocol. We partition the $L = 48$ MoE layers into three depth bands, vary ϵ_{null} within one band at a time while holding the remaining bands at the global default 10^{-2} , and report the resulting whole-model metrics. The 10^{-2} row therefore coincides with the global configuration of Table 2 and serves as the reference. All runs use WMDP-Cyber with RMU at matched forget accuracy, and are single-seed, so differences below roughly 0.02 should not be over-interpreted.

Table 10: Per-band threshold sweep on WMDP-Cyber (RMU). Each cell reports RS / FA / RA with ϵ_{null} applied to that band only and all other bands held at 10^{-2} . The 10^{-2} row reproduces the global configuration of Table 2.

ϵ_{null}	Early (1–16)	Mid (17–32)	Late (33–48)
10^{-4}	0.97 / 0.29 / 0.62	0.96 / 0.28 / 0.63	0.97 / 0.27 / 0.62
10^{-3}	0.96 / 0.26 / 0.62	0.96 / 0.25 / 0.64	0.96 / 0.25 / 0.64
10^{-2}	0.96 / 0.24 / 0.65	0.96 / 0.24 / 0.65	0.96 / 0.24 / 0.65
10^{-1}	0.92 / 0.24 / 0.59	0.92 / 0.24 / 0.58	0.93 / 0.24 / 0.59

The optimum does not shift with depth. As shown in Table 10, $\epsilon_{\text{null}} = 10^{-2}$ remains the preferred setting in every band. Tightening buys marginal routing stability at the cost of erasure, while loosening to 10^{-1} leaves forget accuracy at 0.24 but costs both routing stability and roughly six points of retain accuracy. We therefore find no evidence that optimal thresholds diverge with depth at this granularity, which justifies the global setting used throughout the main text.

Layer-adaptive scheduling. Because the per-band optima coincide, a schedule built from them would reduce to the global default, and these results give no reason to prefer layer-adaptive selection at band granularity. The asymmetry above does suggest where a finer-grained schedule might pay: late layers absorb tightening cheaply, so a threshold that decreases with depth is the direction worth exploring, rather than the increasing one that a naive drift argument would suggest. Establishing that requires per-layer or per-expert search rather than the three-way partition used here, and the bands interact through drift so such a search cannot be decomposed band by band. We leave this to future work.