A UNIFIED NTK-BASED FRAMEWORK FOR FORGETTING APPROXIMATION IN FULL FINE-TUNING LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Full fine-tuning enables large language models (LLMs) to acquire new knowledge but also alters previously mastered behavior. Therefore, understanding how forgetting arises during fine-tuning is crucial. We develop a neural tangent kernel (NTK)-based framework to locally approximate output changes associated with forgetting. The derivation separates the change into Gauss–Newton (GGN) quadratic term and linearization remainder terms. We use GGN quadratic to approximate the output change, which consists of three factors: transfer geometry, curvature response, and residual signal. We further show that three major forgetting-mitigation approaches can be viewed as modifications to these factors. Based on this framework, we proposed NTK-Replay, which selects replay strength by balancing approximated drift against replay penalty. Extensive experiments covering three modalities, five datasets, and twelve models validate the effectiveness of our NTK-based approximation framework.

1 INTRODUCTION

Large language models have attracted widespread attention for their powerful knowledge capabilities (Petroni et al., 2019; Roberts et al., 2020; Yang et al., 2026a). Full fine-tuning updates all parameters of LLMs to learn new knowledge and tasks. These updates can also damage previously mastered knowledge, causing catastrophic forgetting. Retaining existing capabilities while learning new ones is therefore a central challenge in full fine-tuning (Kirkpatrick et al., 2017; Saha et al., 2021). Four common approaches to mitigating forgetting include gradient projection (Saha et al., 2021), parameter regularization (Kirkpatrick et al., 2017), experience replay (Buzzega et al., 2020; Aljundi et al., 2019), and restricting trainable parameters (Wang et al., 2023; Zheng et al., 2025).

Modeling the mechanism of forgetting is important for understanding its causes and designing effective interventions. However, accurately modeling forgetting during LLM fine-tuning remains challenging because parameter updates change how examples influence each other. NTK theory offers a natural perspective by describing this influence through gradient interactions; in the infinite-width regime, the associated gradient kernel remains fixed during training (Jacot et al., 2018). Existing NTK-based studies have investigated catastrophic forgetting through gradient interference and kernel-based analyses (Doan et al., 2021; Zheng et al., 2026; Liu et al., 2026; Hidekel & Raviv, 2026). However, applying these analyses to forgetting modeling in LLM fine-tuning still faces several challenges.

Loss-function mismatch. Existing NTK-based mechanistic frameworks for forgetting are developed for squared loss. They do not directly apply to LLM full fine-tuning, which typically uses cross-entropy loss.

Mitigation-strategy mismatch. Existing formulations mainly describe vanilla full fine-tuning, while practical training often incorporates forgetting-mitigation strategies such as projection, regularization, and replay. These approaches alter the local response, which cannot be characterized by existing formulations.

To address these gaps, we develop a local NTK-based approximation of forgetting during LLM full fine-tuning with cross-entropy loss. Our main contributions are:

• **Contribution 1: NTK-based approximation of output drift during LLM full fine-tuning.** Based on NTK theory, we derive a formula to approximate output drift when fully fine-tuning LLMs

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

with cross-entropy loss. The formula decomposes the response into transfer geometry, curvature response, and residual signal. Experiments spanning three modalities, five datasets, and twelve models, exhibit that predicted and observed drifts have a Pearson correlation of 0.796 ± 0.153.

• **Contribution 2: NTK-based unified framework for forgetting mitigation.** Under the NTK approximation, we unify three common forgetting-mitigation strategies, namely projection, regularization, and replay, by viewing them as distinct modifications of the three response factors. Moreover, we characterize the conditions for approximate equivalence among their local response forms.

• **Contribution 3: NTK-based replay strength optimization.** The local NTK approximation enables replay-buffer-weight selection by balancing approximated output drift against a penalty on replay strength. We instantiate this objective in NTK-Replay. In the evaluated configurations, it has the highest mean accuracy and the second-lowest mean forgetting rate on all three datasets.

2 Related Work

Forgetting in LLM full fine-tuning. Prior work studies forgetting through representation interference, parameter importance, and loss-landscape geometry (McCloskey & Cohen, 1989; Kirkpatrick et al., 2017; Li et al., 2024). Existing forgetting-mitigation methods broadly follow four major approaches: projection restricts gradient directions to preserve prior knowledge (Saha et al., 2021; Yang et al., 2026b). Parameter-restriction methods limit adaptation to selected modules or layers (Wang et al., 2023; Zheng et al., 2025). Regularization penalizes changes to important parameters (Kirkpatrick et al., 2017), while replay incorporates gradients from stored examples (Buzzega et al., 2020; Aljundi et al., 2019; Chen et al., 2026). However, these approaches are typically studied in isolation, without a unified framework to characterize their mechanisms.

NTK Theory. NTK theory approximates neural-network training through a local linearization of model outputs (Jacot et al., 2018; Lee et al., 2019), providing a tractable framework for analyzing output changes during training. Using this approximation, prior work measures cross-task gradient-kernel overlap to analyze interference (Doan et al., 2021), relates forgetting and generalization through an NTK formulation (Zheng et al., 2026), studies parameter-efficient continual fine-tuning (Liu et al., 2026), and derives a low-rank characterization of output drift (Hidekel & Raviv, 2026). However, these NTK-based output-drift formulations cannot be directly applied to LLM fine-tuning because they assume squared loss, whereas LLMs are typically fine-tuned with cross-entropy loss.

Random projections for gradient-kernel computation. Full fine-tuning of LLMs involves a large number of trainable parameters, making it costly to store per-example gradients and compute gradient kernels. Johnson–Lindenstrauss projections compress gradients while approximately preserving their geometry, enabling efficient gradient-kernel approximation (Johnson & Lindenstrauss, 1984; Park et al., 2023). Standard choices include Gaussian and Rademacher random projections (Achlioptas, 2003; Park et al., 2023), while CountSketch and TensorSketch provide sparse and structured alternatives (Charikar et al., 2002; Pham & Pagh, 2013). Faster gradient-kernel approximation can therefore facilitate the practical application of NTK theory to LLM fine-tuning.

3 Method

3.1 NTK-based approximation of output drift during LLM full fine-tuning

Hereafter, "fine-tuning" refers to full fine-tuning. Following prior studies of forgetting in fine-tuning (Jin & Ren, 2024; Zhou et al., 2025), we partition each dataset according to the initial model with parameters θ_0. The fine-tuning set $\mathcal{F}$ contains examples answered incorrectly by the initial model and therefore represents the knowledge to be learned. The mastered set $\mathcal{M}$ contains examples answered correctly and represents the knowledge whose preservation we study. The model is fine-tuned on $\mathcal{F}$, while forgetting is measured through its changes on $\mathcal{M}$.

Adapting the output-based perspective of prior NTK analyses (Doan et al., 2021; Hidekel & Raviv, 2026), we represent the model's confidence on each example by the correct-answer log-odds. For an

example x with correct answer token y, let $p_\theta(x)$ denote the token probability under the full output vocabulary and define

$$f_\theta(x) = \log \frac{p_\theta(x)}{1 - p_\theta(x)} = z_y - \log \sum_{v \neq y} \exp z_v. \tag{1}$$

Here z is the next-token logit vector. For a parameter update $u = \theta - \theta_0$, the mastered-set drift is $\Delta f_{\mathcal{M}} = f_{\mathcal{M}}(\theta_0 + u) - f_{\mathcal{M}}(\theta_0)$. Our goal is to study the change of $\Delta f_{\mathcal{M}}$ while fine-tuning on $\mathcal{F}$. We use the following damped fine-tuning objective:

$$L_{\mathcal{F}}(\theta) = -\frac{1}{|\mathcal{F}|}\mathbf{1}^\top \log p_{\mathcal{F}}(\theta) + \frac{\mu}{2|\mathcal{F}|}\|\theta - \theta_0\|_2^2, \tag{2}$$

where $p_{\mathcal{F}}(\theta)$ contains the correct-answer probabilities and $\mu > 0$ is the damping coefficient shared by all response models. Substituting the log-odds definition in Eq. (1) into Eq. (2) gives

$$L_{\mathcal{F}}(\theta) = \frac{1}{|\mathcal{F}|}\mathbf{1}^\top \log(1 + \exp(-f_{\mathcal{F}}(\theta))) + \frac{\mu}{2|\mathcal{F}|}\|\theta - \theta_0\|_2^2. \tag{3}$$

At θ_0, abbreviate $p_{\mathcal{F}}(\theta_0)$ as $p_{\mathcal{F}}$. Differentiating the cross-entropy term in Eq. (3) with respect to its output vector gives

$$\nabla_{f_{\mathcal{F}}} L_{\mathcal{F}} = -\frac{1}{|\mathcal{F}|}(1 - p_{\mathcal{F}}), \qquad \nabla^2_{f_{\mathcal{F}}} L_{\mathcal{F}} = \frac{1}{|\mathcal{F}|} H_{\mathcal{F}}, \tag{4}$$

where $H_{\mathcal{F}} = \mathrm{diag}(p_{\mathcal{F}} \odot (1 - p_{\mathcal{F}}))$. Taking a second-order Taylor expansion of the cross-entropy term around $f_{\mathcal{F}}(\theta_0)$, while retaining the parameter penalty exactly, gives

$$\Delta L_{\mathcal{F}} \approx \frac{1}{|\mathcal{F}|}\left[-(1 - p_{\mathcal{F}})^\top \Delta f_{\mathcal{F}} + \tfrac{1}{2}\Delta f_{\mathcal{F}}^\top H_{\mathcal{F}} \Delta f_{\mathcal{F}}\right] + \frac{\mu}{2|\mathcal{F}|}\|\theta - \theta_0\|_2^2. \tag{5}$$

Here $\Delta L_{\mathcal{F}} = L_{\mathcal{F}}(\theta) - L_{\mathcal{F}}(\theta_0)$ and $\Delta f_{\mathcal{F}} = f_{\mathcal{F}}(\theta) - f_{\mathcal{F}}(\theta_0)$.

To connect output changes to parameter updates, define the output Jacobian for a sample set X and its value at the initial model:

$$\Phi_X(\theta) = \frac{\partial f_X(\theta)}{\partial \theta^\top}, \qquad \Phi_X^{(0)} = \Phi_X(\theta_0) \in \mathbb{R}^{|X| \times D}. \tag{6}$$

where D is the number of parameters. Each entry of $\Phi_X(\theta)$ measures how one parameter affects one sample's output. A first-order expansion at θ_0 gives

$$f_X(\theta_0 + u) = f_X(\theta_0) + \Phi_X^{(0)} u + r_X(u), \qquad \Delta f_X = \Phi_X^{(0)} u + r_X(u), \tag{7}$$

where $r_X(u)$ is the output-linearization remainder. In particular,

$$\Delta f_{\mathcal{F}} = \Phi_{\mathcal{F}}^{(0)} u + r_{\mathcal{F}}(u). \tag{8}$$

Let $q_{\mathcal{F}} = 1 - p_{\mathcal{F}}$. Substituting Eq. (8) into Eq. (5), without discarding $r_{\mathcal{F}}(u)$, gives

$$\begin{aligned}\Delta L_{\mathcal{F}} \approx \frac{1}{|\mathcal{F}|}\Big[&- q_{\mathcal{F}}^\top\big(\Phi_{\mathcal{F}}^{(0)} u + r_{\mathcal{F}}(u)\big) \\ &+ \frac{1}{2}\big(\Phi_{\mathcal{F}}^{(0)} u + r_{\mathcal{F}}(u)\big)^\top H_{\mathcal{F}}\big(\Phi_{\mathcal{F}}^{(0)} u + r_{\mathcal{F}}(u)\big) + \frac{\mu}{2}\|u\|_2^2\Big].\end{aligned} \tag{9}$$

Expanding the quadratic product separates the terms that depend only on the initial Jacobian from those containing the output-linearization remainder:

$$\begin{aligned}\Delta L_{\mathcal{F}} \approx \frac{1}{|\mathcal{F}|}\Big[&- q_{\mathcal{F}}^\top \Phi_{\mathcal{F}}^{(0)} u + \frac{1}{2} u^\top \big(\Phi_{\mathcal{F}}^{(0)\top} H_{\mathcal{F}} \Phi_{\mathcal{F}}^{(0)} + \mu I\big) u \\ &- q_{\mathcal{F}}^\top r_{\mathcal{F}}(u) + u^\top \Phi_{\mathcal{F}}^{(0)\top} H_{\mathcal{F}} r_{\mathcal{F}}(u) + \frac{1}{2} r_{\mathcal{F}}(u)^\top H_{\mathcal{F}} r_{\mathcal{F}}(u)\Big].\end{aligned} \tag{10}$$

Define the initial-state GGN quadratic as

$$Q_{\mathrm{GGN}}(u) = \frac{1}{|\mathcal{F}|}\left[-q_{\mathcal{F}}^\top \Phi_{\mathcal{F}}^{(0)} u + \frac{1}{2} u^\top \big(\Phi_{\mathcal{F}}^{(0)\top} H_{\mathcal{F}} \Phi_{\mathcal{F}}^{(0)} + \mu I\big) u\right], \tag{11}$$

and collect every term involving $r_{\mathcal{F}}(u)$ into

$$\rho_r(u) = \frac{1}{|\mathcal{F}|}\Big[-q_{\mathcal{F}}^\top r_{\mathcal{F}}(u) + u^\top \Phi_{\mathcal{F}}^{(0)\top} H_{\mathcal{F}} r_{\mathcal{F}}(u) + \frac{1}{2} r_{\mathcal{F}}(u)^\top H_{\mathcal{F}} r_{\mathcal{F}}(u)\Big]. \tag{12}$$

Therefore, the local objective can be written as

$$\Delta L_{\mathcal{F}} \approx Q_{\mathrm{GGN}}(u) + \rho_r(u). \tag{13}$$

Let

$$A_{\mathcal{F}}^{(0)} = \Phi_{\mathcal{F}}^{(0)\top} H_{\mathcal{F}} \Phi_{\mathcal{F}}^{(0)} + \mu I. \tag{14}$$

Differentiating both terms in Eq. (13) gives the stationary condition

$$0 \approx \nabla_u Q_{\mathrm{GGN}}(u^\star) + \nabla_u \rho_r(u^\star) = \frac{1}{|\mathcal{F}|}\left[A_{\mathcal{F}}^{(0)} u^\star - \Phi_{\mathcal{F}}^{(0)\top} q_{\mathcal{F}}\right] + \nabla_u \rho_r(u^\star). \tag{15}$$

Rearranging Eq. (15) retains the contribution of ρ_r in the parameter update:

$$u^\star = \big(A_{\mathcal{F}}^{(0)}\big)^{-1} \Phi_{\mathcal{F}}^{(0)\top} q_{\mathcal{F}} - |\mathcal{F}| \big(A_{\mathcal{F}}^{(0)}\big)^{-1} \nabla_u \rho_r(u^\star). \tag{16}$$

Define $K_{XY}^{(0)} = \Phi_X^{(0)} \Phi_Y^{(0)\top}$. Each entry of $K_{XY}^{(0)}$ is the inner product between two examples' initial output gradients and measures their local coupling through shared parameters. Applying the push-through identity to the first term of Eq. (16) gives

$$u^\star = \Phi_{\mathcal{F}}^{(0)\top} \big(H_{\mathcal{F}} K_{\mathcal{F}\mathcal{F}}^{(0)} + \mu I\big)^{-1} q_{\mathcal{F}} - |\mathcal{F}| \big(A_{\mathcal{F}}^{(0)}\big)^{-1} \nabla_u \rho_r(u^\star). \tag{17}$$

The mastered-set output change also retains its own output-linearization remainder. Substituting Eq. (17) into $\Delta f_{\mathcal{M}} = \Phi_{\mathcal{M}}^{(0)} u^\star + r_{\mathcal{M}}(u^\star)$ yields

$$\Delta f_{\mathcal{M}} \approx K_{\mathcal{M}\mathcal{F}}^{(0)} \big(H_{\mathcal{F}} K_{\mathcal{F}\mathcal{F}}^{(0)} + \mu I\big)^{-1} q_{\mathcal{F}} + \varepsilon_{\mathrm{NTK}}(u^\star), \tag{18}$$

The correction collects the two remainder contributions:

$$\varepsilon_{\mathrm{NTK}}(u^\star) = r_{\mathcal{M}}(u^\star) - |\mathcal{F}| \Phi_{\mathcal{M}}^{(0)} \big(A_{\mathcal{F}}^{(0)}\big)^{-1} \nabla_u \rho_r(u^\star). \tag{19}$$

To relate this correction to the NTK approximation, define the local tangent-feature drift

$$\eta_{\mathrm{NTK}}(u) = \max_{X \in \{\mathcal{F}, \mathcal{M}\}} \sup_{s \in [0,1]} \left\| \Phi_X(\theta_0 + su) - \Phi_X^{(0)} \right\|_{\mathrm{op}}.$$

The first-order Taylor remainder then satisfies

$$\|r_X(u)\|_2 \le \eta_{\mathrm{NTK}}(u) \|u\|_2, \qquad \|\nabla_u r_X(u)\|_{\mathrm{op}} \le \eta_{\mathrm{NTK}}(u).$$

The bounds above show that this correction is controlled by the local tangent-feature drift together with the norms of the local response factors. NTK theory proposes that the tangent features and resulting kernel remain close to their initial values during training (Jacot et al., 2018; Lee et al., 2019). Accordingly, existing NTK-based studies use the leading initial-state response to approximate training behavior (Doan et al., 2021; Zheng et al., 2026; Liu et al., 2026; Hidekel & Raviv, 2026). Following this convention, we use the leading initial-state term in Eq. (18) to approximate the full output change:

$$\widehat{\Delta f_{\mathcal{M}}} = \underbrace{K_{\mathcal{M}\mathcal{F}}^{(0)}}_{\text{transfer geometry}} \underbrace{(H_{\mathcal{F}} K_{\mathcal{F}\mathcal{F}}^{(0)} + \mu I)^{-1}}_{\text{curvature response}} \underbrace{(1 - p_{\mathcal{F}})}_{\text{residual signal}}. \tag{20}$$

Equation (20) decomposes the log-odds response into transfer geometry, curvature response, and residual signal. Compared with NTK analyses under squared loss, it includes the probability-dependent curvature $H_{\mathcal{F}}$, which arises from the second-order expansion of cross-entropy in log-odds space.

3.2 NTK-BASED UNIFIED PERSPECTIVE ON THREE FORGETTING-MITIGATION METHODS

We next analyze how projection, regularization, and replay modify the local log-odds response in Eq. (20). Here, $R \subset \mathcal{M}$ denotes the mastered examples used for protection or replay. All Jacobians and kernels in this section are evaluated at θ_0, and the superscript (0) is omitted for brevity.

Table 1: Response components modified by each forgetting-mitigation strategy under the NTK-based local approximation. A checkmark indicates a modification relative to vanilla fine-tuning.

Method	Transfer geometry	Curvature response	Residual signal	Direct drift correction
Projection	✓	✓	–	–
Regularization	✓	✓	–	–
Replay	✓	✓	✓	✓

Projection changes transfer geometry and curvature response. An output-preserving projection constrains parameter updates to directions that do not change the protected outputs:

$$\min_u L_{\mathcal{F}}(\theta_0 + u) \quad \text{subject to} \quad \Phi_R u = 0. \tag{21}$$

In Appendix A.1, we show that each training gradient is projected by $P_R = I - \Phi_R^\top K_{RR}^\dagger \Phi_R$, where $\dagger$ denotes the Moore–Penrose pseudoinverse. The resulting local log-odds response is

$$\widehat{\Delta f_{\mathcal{M}}}^{\text{Proj}} = K_{\mathcal{MF}}^{\text{Proj}} (H_{\mathcal{F}} K_{\mathcal{FF}}^{\text{Proj}} + \mu I)^{-1} (1 - p_{\mathcal{F}}), \tag{22}$$

where the cross-set and fine-tuning-set kernels are

$$\begin{aligned} K_{\mathcal{MF}}^{\text{Proj}} &= \Phi_{\mathcal{M}} P_R \Phi_{\mathcal{F}}^\top = K_{\mathcal{MF}} - K_{\mathcal{M}R} K_{RR}^\dagger K_{R\mathcal{F}}, \\ K_{\mathcal{FF}}^{\text{Proj}} &= \Phi_{\mathcal{F}} P_R \Phi_{\mathcal{F}}^\top = K_{\mathcal{FF}} - K_{\mathcal{F}R} K_{RR}^\dagger K_{R\mathcal{F}}. \end{aligned} \tag{23}$$

Regularization changes transfer geometry and curvature response. Let $C \succeq 0$ be a penalty matrix and $\lambda \geq 0$ its strength. In addition to the shared damping μ, regularization introduces the method-specific quadratic penalty

$$L_{\text{reg}}(\theta) = L_{\mathcal{F}}(\theta) + \frac{\lambda}{2|\mathcal{F}|} (\theta - \theta_0)^\top C (\theta - \theta_0). \tag{24}$$

Appendix A.2 derives how this penalty changes the local log-odds response. The penalty contributes $\lambda C u / |\mathcal{F}|$ to the gradient, yielding

$$\widehat{\Delta f_{\mathcal{M}}}^{\text{Reg}} = K_{\mathcal{MF}}^{\text{Reg}} (H_{\mathcal{F}} K_{\mathcal{FF}}^{\text{Reg}} + \mu I)^{-1} (1 - p_{\mathcal{F}}), \tag{25}$$

where

$$\begin{aligned} K_{\mathcal{MF}}^{\text{Reg}} &= \mu \Phi_{\mathcal{M}} (\mu I + \lambda C)^{-1} \Phi_{\mathcal{F}}^\top, \\ K_{\mathcal{FF}}^{\text{Reg}} &= \mu \Phi_{\mathcal{F}} (\mu I + \lambda C)^{-1} \Phi_{\mathcal{F}}^\top. \end{aligned} \tag{26}$$

Replay changes all three response factors and adds a direct drift correction. Replay adds mastered-example losses with replay weight $\rho \geq 0$:

$$L_{\text{replay}}(\theta) = L_{\mathcal{F}}(\theta) - \frac{\rho}{|\mathcal{F}|} \mathbf{1}^\top \log p_R(\theta). \tag{27}$$

Appendix A.3 derives how replay changes the local log-odds response: it adds curvature from the replay examples and a gradient term that drives their loss downward. The resulting response is

$$\widehat{\Delta f_{\mathcal{M}}}^{\text{Replay}} = a_{\mathcal{M}} + K_{\mathcal{MF}}^{\text{Replay}} (H_{\mathcal{F}} K_{\mathcal{FF}}^{\text{Replay}} + \mu I)^{-1} (1 - p_{\mathcal{F}}^{\text{Replay}}). \tag{28}$$

Here, at θ_0, we abbreviate $p_R(\theta_0)$ as p_R and define $H_R = \text{diag}(p_R \odot (1 - p_R))$ and $F_R = \Phi_R^\top H_R \Phi_R$. The transfer kernel, fine-tuning kernel, and output corrections are

$$\begin{aligned} K_{\mathcal{MF}}^{\text{Replay}} &= \mu \Phi_{\mathcal{M}} (\mu I + \rho F_R)^{-1} \Phi_{\mathcal{F}}^\top, \quad a_{\mathcal{M}} = \rho \Phi_{\mathcal{M}} (\mu I + \rho F_R)^{-1} \Phi_R^\top (1 - p_R), \\ K_{\mathcal{FF}}^{\text{Replay}} &= \mu \Phi_{\mathcal{F}} (\mu I + \rho F_R)^{-1} \Phi_{\mathcal{F}}^\top, \quad a_{\mathcal{F}} = \rho \Phi_{\mathcal{F}} (\mu I + \rho F_R)^{-1} \Phi_R^\top (1 - p_R). \end{aligned} \tag{29}$$

The locally replay-adjusted residual signal is $1 - p_{\mathcal{F}}^{\text{Replay}} := (1 - p_{\mathcal{F}}) - H_{\mathcal{F}} a_{\mathcal{F}}$.

Table 1 summarizes which response components are modified by each strategy.

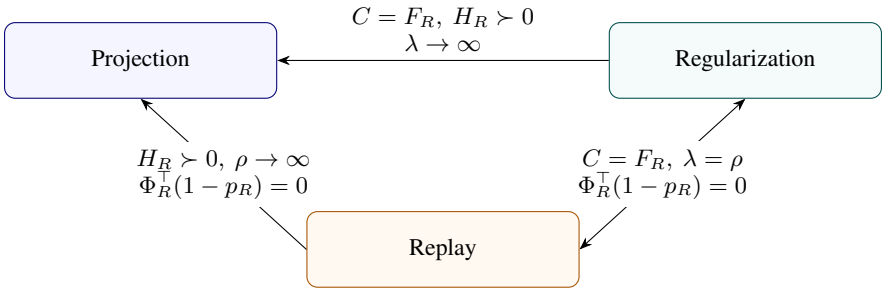

Figure 1: Conditional equivalences between three forgetting-mitigation methods within the local NTK-based approximation. Boxes denote methods, and edge labels specify the conditions. Single arrows denote limiting equivalences; the double arrow denotes equality under the stated conditions.

Conditional equivalences. Within the local approximation, full-curvature regularization converges to the output-preserving projection response as $\lambda \to \infty$. When $C = F_R$ and $H_R \succ 0$, increasing λ suppresses directions that change protected outputs, so the effective kernels satisfy

$$K^{\mathrm{Reg}}_{\mathcal{MF}} \longrightarrow K^{\mathrm{Proj}}_{\mathcal{MF}}, \qquad K^{\mathrm{Reg}}_{\mathcal{FF}} \longrightarrow K^{\mathrm{Proj}}_{\mathcal{FF}} \qquad (\lambda \to \infty), \tag{30}$$

This kernel limit yields the corresponding output-response limit, as proved in Appendix A.2.

Replay and full-curvature regularization have the same local surrogate equivalence when the aggregate replay gradient vanishes. With $C = F_R$ and $\lambda = \rho$, their effective kernels coincide; if $\Phi_R^\top(1 - p_R) = 0$, then $a_{\mathcal{M}} = a_{\mathcal{F}} = 0$ as well. Appendix A.3 proves this equivalence.

The local replay response converges to the output-preserving projection response in the strong-replay limit. When $H_R \succ 0$ and $\Phi_R^\top(1 - p_R) = 0$, replay equals full-curvature regularization at $\lambda = \rho$ within the surrogate; taking $\rho \to \infty$ then gives the projection response. This follows from Appendices A.2 and A.3, and Figure 1 summarizes all three relationships.

3.3 NTK-BASED REPLAY STRENGTH OPTIMIZATION

To demonstrate how the local NTK-based approximation can guide forgetting mitigation, we use its replay response to select the replay weight in NTK-Replay. Specifically, we define the approximated mastered-set drift magnitude as

$$D_{\mathcal{M}}(\rho) = \|\widehat{\Delta f_{\mathcal{M}}}^{\mathrm{Replay}}(\rho)\|_2^2. \tag{31}$$

With $\omega > 0$ controlling the replay-strength penalty, we select

$$\rho^\star = \arg\min_{\rho \geq 0} \frac{D_{\mathcal{M}}(\rho)}{D_{\mathcal{M}}(0)} + \omega\rho^2. \tag{32}$$

The first term favors preservation of mastered outputs, while the second penalizes the replay weight. Dividing by $D_{\mathcal{M}}(0)$ expresses the drift relative to the no-replay baseline and reduces sensitivity to differences in absolute drift scale across model–dataset settings.

4 EXPERIMENTS

4.1 SETUP

Datasets. Our experiments span three modalities—text, image, and speech—across five datasets: MMLU (Hendrycks et al., 2021), MedQA (Jin et al., 2021), and CSQA (Talmor et al., 2019) for text, MMBench (Liu et al., 2024) for image, and MMSU (Wang et al., 2025) for speech.

Models. We evaluate ten text models: *Qwen-1.5B*, *Qwen-3B*, and *Qwen-7B* (Qwen Team, 2024); *Llama-1B* and *Llama-3B* (Meta, 2024), and *Llama-8B* (Grattafiori et al., 2024); *Gemma-2B* and *Gemma-9B* (Gemma Team et al., 2024); *GLM-9B* (GLM Team et al., 2024); and *Mistral-7B* (Jiang et al., 2023). The multimodal cases use *Qwen-VL-3B* (Bai et al., 2025) and *Qwen-Omni-7B* (Xu et al., 2025). Model version numbers are omitted in the remainder of the paper.

Table 2: Pearson correlations between locally approximated and observed log-odds change. No shuffle denotes direct application of the proposed formula, while Shuffle randomly permutes the entries within the specified component. Green and blue shades indicate decreased ($\Delta \leq -0.001$) and increased ($\Delta \geq 0.001$) correlation relative to no shuffle, respectively.

(a) Text tasks

Model	Perturbation	MMLU Pearson ↑	MedQA Pearson ↑	CSQA Pearson ↑	Macro Pearson ↑
Qwen-1.5B	No shuffle	0.747 ±0.013	0.825 ±0.006	0.650 ±0.011	0.741 ±0.088
	Shuffle $K_{\mathcal{MF}}$	0.689 ±0.015	0.791 ±0.008	0.577 ±0.009	0.686 ±0.107
	Shuffle $H_{\mathcal{F}}$	0.723 ±0.013	0.811 ±0.006	0.630 ±0.012	0.721 ±0.091
	Shuffle $K_{\mathcal{FF}}$	0.709 ±0.016	0.790 ±0.006	0.578 ±0.012	0.692 ±0.107
	Shuffle $1 - p_{\mathcal{F}}$	0.745 ±0.013	0.825 ±0.006	0.650 ±0.011	0.740 ±0.088
Qwen-7B	No shuffle	0.840 ±0.011	0.896 ±0.003	0.886 ±0.002	0.874 ±0.030
	Shuffle $K_{\mathcal{MF}}$	0.758 ±0.032	0.876 ±0.007	0.847 ±0.021	0.827 ±0.061
	Shuffle $H_{\mathcal{F}}$	0.822 ±0.009	0.884 ±0.004	0.869 ±0.004	0.858 ±0.032
	Shuffle $K_{\mathcal{FF}}$	0.767 ±0.021	0.879 ±0.005	0.879 ±0.005	0.842 ±0.065
	Shuffle $1 - p_{\mathcal{F}}$	0.840 ±0.011	0.896 ±0.003	0.883 ±0.002	0.873 ±0.029
Llama-8B	No shuffle	0.883 ±0.006	0.954 ±0.003	0.931 ±0.001	0.923 ±0.036
	Shuffle $K_{\mathcal{MF}}$	0.759 ±0.010	0.784 ±0.011	0.860 ±0.008	0.801 ±0.053
	Shuffle $H_{\mathcal{F}}$	0.837 ±0.004	0.918 ±0.006	0.909 ±0.006	0.888 ±0.044
	Shuffle $K_{\mathcal{FF}}$	0.835 ±0.006	0.866 ±0.010	0.903 ±0.000	0.868 ±0.034
	Shuffle $1 - p_{\mathcal{F}}$	0.865 ±0.007	0.937 ±0.005	0.928 ±0.002	0.910 ±0.039

(b) Image and audio tasks

Perturbation	MMBench (*Qwen-VL-3B*) (Image) Pearson ↑	MMSU (*Qwen-Omni-7B*) (Audio) Pearson ↑	Macro Pearson ↑
No shuffle	0.647 ±0.065	0.787 ±0.016	0.717 ±0.099
Shuffle $K_{\mathcal{MF}}$	0.083 ±0.054	0.139 ±0.145	0.111 ±0.040
Shuffle $H_{\mathcal{F}}$	0.513 ±0.082	−0.424 ±0.089	0.045 ±0.663
Shuffle $K_{\mathcal{FF}}$	0.061 ±0.042	0.292 ±0.122	0.176 ±0.163
Shuffle $1 - p_{\mathcal{F}}$	0.641 ±0.066	0.771 ±0.014	0.706 ±0.092

Protocol. For each dataset, we generate three independent random 7:3 train–test splits. Motivated by prior work (Jin & Ren, 2024; Zhou et al., 2025), within each split, we classify the training and test examples separately according to whether the initial model gives the correct answer. This yields the mastered training set $\mathcal{M}_{\text{train}}$ and fine-tuning training set $\mathcal{F}_{\text{train}}$, together with the mastered test set $\mathcal{M}_{\text{test}}$ and fine-tuning test set $\mathcal{F}_{\text{test}}$. In practice, the full mastered set is unavailable during training. Therefore, we train on the initially incorrect examples in $\mathcal{F}_{\text{train}}$ and use $\mathcal{M}_{\text{train}}$ as the reference set for gradient computation when required. $\mathcal{F}_{\text{test}}$ and $\mathcal{M}_{\text{test}}$ remain unseen during training and are only used for evaluation. Results are reported as means ± standard deviations over the three splits. Additional training and implementation details are provided in Appendix B.

Baselines. We compare with vanilla SGD, projection methods (GPM (Saha et al., 2021) and OPGD (Zheng et al., 2026)), parameter regularization (EWC (Kirkpatrick et al., 2017)), replay methods (ER (Rolnick et al., 2019), ER + DWA (Liu et al., 2019), DER++ (Buzzega et al., 2020), and Smart Replay (Chen et al., 2026)), and methods that restrict trainable parameters (O-LoRA (Wang et al., 2023) and Bottom-Layer Freezing (BLF) (Zheng et al., 2025)). Several baselines originated in continual learning; we include them because their update rules aim to reduce forgetting. Our experiments focus on a single model-refinement step on fixed data partitions, rather than learning a sequence of tasks. Appendix C summarizes their principles and common comparison protocol.

Metrics. **Pearson correlation** measures the correlation between locally approximated and observed log-odds drifts on initially correct examples. **Accuracy** (Acc) is the percentage of examples

Table 3: Pearson correlations between locally approximated and observed log-odds change for three representative mitigation methods: projection (OGD), regularization (EWC), and replay (ER).

Method	MMLU Pearson ↑	MedQA Pearson ↑	CSQA Pearson ↑	Macro Pearson ↑
OGD	0.949 ±0.004	0.952 ±0.003	0.940 ±0.003	0.947 ±0.006
EWC	0.945 ±0.002	0.957 ±0.003	0.870 ±0.008	0.924 ±0.047
ER	0.949 ±0.003	0.966 ±0.002	0.949 ±0.003	0.955 ±0.010

Table 4: Performance after fine-tuning across different methods. Best means are bold and second-best means are underlined. Under the evaluated configurations, NTK-Replay has the highest Acc on 3/3 datasets and the second-lowest FT on 3/3 datasets.

Method	MMLU		CSQA		MedQA	
	Acc↑	FT↓	Acc↑	FT↓	Acc↑	FT↓
SGD	51.22 ±0.48	33.78 ±1.07	67.16 ±0.83	19.86 ±0.16	34.30 ±0.88	53.14 ±1.14
GPM	52.19 ±0.96	30.73 ±1.37	68.71 ±0.96	16.68 ±0.69	35.80 ±1.05	48.74 ±0.83
OPGD	48.80 ±0.20	38.41 ±0.69	66.70 ±0.64	20.75 ±0.29	33.66 ±1.20	55.82 ±1.99
EWC	53.21 ±0.46	29.31 ±1.36	67.75 ±0.72	18.59 ±0.63	37.09 ±0.97	45.15 ±1.16
O-LoRA	57.61 ±0.49	**1.86** ±0.30	<u>74.16</u> ±0.89	**0.56** ±0.02	43.21 ±1.15	**4.69** ±0.51
BLF	52.15 ±0.68	31.91 ±1.32	67.46 ±0.99	19.20 ±0.81	35.93 ±1.01	48.38 ±0.60
ER	54.84 ±0.99	26.09 ±1.08	69.81 ±1.11	15.57 ±0.99	38.58 ±1.22	41.00 ±1.12
ER + DWA	54.95 ±1.00	25.75 ±0.88	70.04 ±1.09	15.19 ±0.91	38.93 ±1.28	40.00 ±1.16
DER++	56.40 ±0.69	22.16 ±0.86	70.78 ±1.27	13.83 ±1.00	42.21 ±0.94	29.26 ±1.50
Smart Replay	<u>57.69</u> ±0.51	16.74 ±0.98	72.00 ±1.49	11.31 ±1.41	<u>44.31</u> ±0.86	14.65 ±1.89
NTK-Replay	**60.35** ±0.51	<u>8.69</u> ±1.40	**77.28** ±0.84	<u>4.92</u> ±0.36	**46.46** ±1.32	<u>11.24</u> ±0.59

in $\mathcal{M}_{\text{test}} \cup \mathcal{F}_{\text{test}}$ answered correctly after fine-tuning. **Forgetting rate** (FT) is the percentage of examples in $\mathcal{M}_{\text{test}}$ answered incorrectly after fine-tuning.

4.2 The local approximation tracks observed forgetting-related drift

We compare the locally approximated correct-answer log-odds drift with the observed drift on initially mastered examples. Across the resulting 32 model–dataset combinations, the Pearson correlations average 0.796 ± 0.153 (Table 2 and Appendix Table 5). Thus, the NTK-based approximation provides a useful description of log-odds change in the evaluated settings. We also evaluate the root mean squared error (RMSE) of the predicted drift for three representative text models and report the results in Appendix Table 6.

Component analysis. To assess the contribution of each response component to the approximation, we independently shuffle its entries and regard a decrease in Pearson correlation greater than 0.01 as meaningful degradation. Shuffling $K_{\mathcal{MF}}$, $H_{\mathcal{F}}$, and $K_{\mathcal{FF}}$ decreases Pearson correlation in 29/32, 29/32, and 30/32 model–dataset combinations, respectively. The degradation from shuffling $H_{\mathcal{F}}$ supports our inclusion of probability-dependent cross-entropy curvature to improve prediction accuracy. By contrast, shuffling $1 - p_{\mathcal{F}}$ causes meaningful degradation in only 5/32 combinations because this term is close to one for initially incorrect examples (Appendix Figure 2).

Damping sensitivity. To assess the sensitivity of the approximation to damping, we vary the damping-scale coefficient α in $\mu = \alpha\lambda_{\max}(H_{\mathcal{F}}^{1/2} K_{\mathcal{FF}} H_{\mathcal{F}}^{1/2})$ from 0.50 to 1.50. The macro Pearson correlation ranges from 0.723 to 0.752, indicating that the approximation is relatively stable across the evaluated damping range (Appendix Table 7). The experimental method and supporting analyses for the local approximation are detailed in Appendix D.

4.3 THE LOCAL APPROXIMATION UNDER MITIGATION STRATEGIES

To evaluate whether the local approximation applies across different forgetting-mitigation strategies, we examine OGD for projection, EWC for regularization, and ER for replay. Across three text datasets, the Pearson correlations between locally approximated and observed log-odds drifts are 0.947 ± 0.006 for OGD, 0.924 ± 0.047 for EWC, and 0.955 ± 0.010 for ER (Table 3). These correlations indicate that the same local response construction tracks endpoint drift across the three evaluated strategies.

4.4 NTK-REPLAY PERFORMANCE ACROSS LEARNING AND RETENTION METRICS

To evaluate the replay-weight selection derived from the local approximation, we compare NTK-Replay with the baselines on three datasets. Under the evaluated configurations, NTK-Replay achieves the highest mean Acc and the second-lowest mean FT on all three datasets (Table 4).

Efficient gradient-kernel computation. Computing per-example gradient kernels introduces additional time and memory overhead. We therefore propose FG-Sketch, which accelerates JL-style gradient compression by applying factorized TensorSketch directly to the activation and backward-signal factors of linear-layer gradients, avoiding materialization of dense weight-gradient matrices. At the same sketch dimension (8192), FG-Sketch reduces total wall-clock time by 14.7%–24.9% and peak GPU memory by 3.4%–21.9% relative to CountSketch, with 0.76–1.07 percentage points of additional gradient-norm error (Appendix Tables 8, 9, and 10).

Time-to-quality. We further compare wall-clock time of NTK-Replay with Smart Replay (Chen et al., 2026). Including gradient-kernel setup, NTK-Replay with FG-Sketch surpasses Smart Replay's best accuracy over ten epochs using 44.1%–76.7% of Smart Replay's total ten-epoch training time across datasets, compared with 55.0%–97.3% for CountSketch (Appendix Table 11).

Sensitivity to regularization coefficient ω. To determine how the replay-strength penalty affects learning and retention, we vary ω over $\{10^{-5}, 10^{-6}, 10^{-7}\}$ and compare Acc and FT across datasets. Across these settings, NTK-Replay maintains similar accuracy on all three datasets, while larger ω consistently reduces forgetting (Appendix Table 12). Additional methods and supporting results for NTK-Replay are provided in Appendix E.

5 CONCLUSION AND LIMITATIONS

We derive a local NTK-based formula for approximating output drift during cross-entropy LLM full fine-tuning and decompose the response into transfer geometry, curvature response, and residual signal. The same view characterizes how projection, regularization, and replay modify the local response and motivates NTK-Replay for selecting replay strength. Our evaluation covers three modalities, five datasets, and twelve models. The predicted and observed drifts have a mean Pearson correlation of 0.796 ± 0.153, and NTK-Replay achieves the highest mean Acc and the second-lowest mean FT on all three evaluated replay datasets.

Although FG-Sketch accelerates gradient-kernel construction, this stage still introduces additional preparation time (Appendix E.2). Future work could combine distributed computation, such as Distributed Data Parallel, with representative subsampling to approximate the full gradient kernel from a smaller set of examples.

AI USE STATEMENT

In this work, we used generative AI tools primarily to draft portions of the manuscript and improve the clarity and readability of the writing. The tools also assisted with presenting mathematical derivations, interpreting reported results, LaTeX formatting, and figure preparation. They were not used to generate experimental data or results. The complete manuscript and all AI-assisted content were reviewed and revised by all authors.

ETHICS STATEMENT

This work uses publicly available benchmarks and pretrained models to study model behavior during fine-tuning. It does not involve human participants, the collection of private or personally identifiable data, or deployment in real-world decision-making systems. Therefore, we identify no direct ethical concerns arising from the experiments conducted in this study. Dataset and model reuse should follow their respective licenses.

REPRODUCIBILITY STATEMENT

Appendices provide derivations, implementation details, calibration rules, and baseline specifications.

REFERENCES

Dimitris Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *Journal of Computer and System Sciences*, 66(4):671–687, 2003. doi: 10.1016/S0022-0000(03)00025-4.

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems*, volume 32, pp. 11816–11825, 2019.

Shuai Bai, Keqin Chen, Xuejing Liu, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025. URL https://arxiv.org/abs/2502.13923.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://papers.neurips.cc/paper/2020/hash/b704ea2c39778f07c617f6b7ce480e9e-Abstract.html.

Moses Charikar, Kevin C. Chen, and Martin Farach-Colton. Finding frequent items in data streams. In *Automata, Languages and Programming*, volume 2380 of *Lecture Notes in Computer Science*, pp. 693–703. Springer, 2002. doi: 10.1007/3-540-45465-9_59.

Jianting Chen, Dianzhi Yu, and Irwin King. Smart replay: Adaptive scheduling of memory rehearsal for computational resource-aware incremental learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/papers/Chen_Smart_Replay_Adaptive_Scheduling_of_Memory_Rehearsal_for_Computational_Resource-Aware_CVPR_2026_paper.pdf.

Thang Doan, Mehdi Abbana Bennani, Bogdan Mazoure, Guillaume Rabusseau, and Pierre Alquier. A theoretical analysis of catastrophic forgetting through the NTK overlap matrix. In *International Conference on Artificial Intelligence and Statistics*, pp. 1072–1080, 2021. URL https://proceedings.mlr.press/v130/doan21a.html.

Gemma Team et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024. URL https://arxiv.org/abs/2408.00118.

GLM Team et al. ChatGLM: A family of large language models from GLM-130B to GLM-4 All Tools. *arXiv preprint arXiv:2406.12793*, 2024. URL https://arxiv.org/abs/2406.12793.

Aaron Grattafiori et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. URL https://arxiv.org/abs/2407.21783.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021.

Ido Nitzan Hidekel and Dan Raviv. Catastrophic forgetting is low-rank: A function-space theory for continual adaptation. *arXiv preprint arXiv:2606.18024*, 2026. URL https://arxiv.org/abs/2606.18024.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, 2018.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, et al. Mistral 7B. *arXiv preprint arXiv:2310.06825*, 2023. URL https://arxiv.org/abs/2310.06825.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421, 2021. doi: 10.3390/app11146421.

Xisen Jin and Xiang Ren. What will my model forget? forecasting forgotten examples in language model refinement. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 22145–22159. PMLR, 2024. URL https://proceedings.mlr.press/v235/jin24d.html.

William B. Johnson and Joram Lindenstrauss. Extensions of lipschitz mappings into a hilbert space. *Contemporary Mathematics*, 26:189–206, 1984.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114.

Jaehoon Lee, Lechao Xiao, Samuel S. Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Hongyu Li, Liang Ding, Meng Fang, and Dacheng Tao. Revisiting catastrophic forgetting in large language model tuning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 4297–4308, 2024. doi: 10.18653/v1/2024.findings-emnlp.249. URL https://aclanthology.org/2024.findings-emnlp.249/.

Jingren Liu, Zhong Ji, Yunlong Yu, Jiale Cao, Yanwei Pang, Jungong Han, and Xuelong Li. Parameter-efficient fine-tuning for continual learning: A neural tangent kernel perspective. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(7):8297–8314, 2026. doi: 10.1109/TPAMI.2026.3670952.

Shikun Liu, Edward Johns, and Andrew J. Davison. End-to-end multi-task learning with attention. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/papers/Liu_End-To-End_Multi-Task_Learning_With_Attention_CVPR_2019_paper.pdf.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. Mmbench: Is your multi-modal model an all-around player? In *European Conference on Computer Vision*, 2024.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation*, 24:109–165, 1989. doi: 10.1016/S0079-7421(08)60536-8.

Meta. Llama 3.2 model card, 2024. URL https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct.

Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. TRAK: Attributing model behavior at scale. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 27074–27113. PMLR, 23–29 Jul 2023.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 2463–2473. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1250. URL https://aclanthology.org/D19-1250/.

Ninh Pham and Rasmus Pagh. Fast and scalable polynomial kernels via explicit feature maps. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 239–247, 2013. doi: 10.1145/2487575.2487591.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arxiv.org/abs/2412.15115.

Adam Roberts, Colin Raffel, and Noam Shazeer. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pp. 5418–5426. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.437. URL https://aclanthology.org/2020.emnlp-main.437/.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://papers.neurips.cc/paper_files/paper/2019/hash/fa7cdfad1a5aaf8370ebeda47a1ff1c3-Abstract.html.

Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=ZK4Cvv5Eci.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 4149–4158, 2019. doi: 10.18653/v1/N19-1421.

Dingdong Wang, Jincenzi Wu, Junan Li, Dongchao Yang, Xueyuan Chen, Tianhua Zhang, and Helen Meng. MMSU: A massive multi-task spoken language understanding and reasoning benchmark. *arXiv preprint arXiv:2506.04779*, 2025. URL https://arxiv.org/abs/2506.04779.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 10658–10671, 2023. doi: 10.18653/v1/2023.findings-emnlp.715.

Jin Xu, Zhifang Guo, Jinzheng He, et al. Qwen2.5-Omni technical report. *arXiv preprint arXiv:2503.20215*, 2025. URL https://arxiv.org/abs/2503.20215.

Mutian Yang, Jiandong Gao, and Ji Wu. Decoupling knowledge and reasoning in LLMs: An exploration using cognitive dual-system theory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 34268–34276, 2026a. doi: 10.1609/aaai.v40i40.40723. URL https://ojs.aaai.org/index.php/AAAI/article/view/40723.

Mutian Yang, Zisen Zhan, Yutong Chen, Haolin Li, Kaiwen Wang, Kaili Zheng, Yuguang Wang, Qi Wang, Jiandong Gao, and Ji Wu. Collaborative parameter learning: Mitigating forgetting via parameter-level gradient analysis. *arXiv preprint arXiv:2601.21577*, 2026b. doi: 10.48550/arXiv.2601.21577. URL https://arxiv.org/abs/2601.21577.

Guodong Zheng, Peng Wang, Shengchao Hu, Quan Zheng, and Li Shen. Understanding the dynamics of forgetting and generalization in continual learning via the neural tangent kernel. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/1f8638a87bddd34e090698f0bfaa9c02-Abstract-Conference.html.

Junhao Zheng, Xidi Cai, Shengjie Qiu, and Qianli Ma. Spurious forgetting in continual learning of language models. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/a774503daed55eb53c634847ae071ec7-Abstract-Conference.html.

Yuxuan Zhou, Xien Liu, Xiao Zhang, Chen Ning, Shijin Wang, Guoping Hu, and Ji Wu. Investigating and mitigating catastrophic forgetting in medical knowledge injection through internal knowledge augmentation learning. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-0308. URL https://papers.neurips.cc/paper_files/paper/2025/hash/0d2c50691b832d8e8901715998e7c22c-Abstract-Conference.html.

A LOCAL-RESPONSE DERIVATIONS FOR THREE FORGETTING-MITIGATION METHODS

We use the notation, average loss, initial output Jacobians, second-order loss expansion, and damping defined in Sections 3.1 and 3.2. Following Section 3.2, the superscript (0) is omitted from initial Jacobians and kernels. The main derivation retains the output-linearization correction through Eq. (19) and uses the leading initial-state response to define the practical predictor in Eq. (20). The method-specific derivations below use the same approximation and give the corresponding initial-state response forms and effective kernels.

A.1 OUTPUT-PRESERVING PROJECTION

An output-preserving projection removes update components that change protected outputs. Under the initial-state approximation of Section 3.1, the retained first-order output change on R is $\Delta f_R \approx \Phi_R u$. Preserving these outputs therefore constrains the loss minimization:

$$\min_u L_{\mathcal{F}}(\theta_0 + u) \quad \text{subject to} \quad \Phi_R u = 0. \tag{33}$$

To construct an allowed update from any proposed update v, subtract its component in the span of the protected output gradients. Each row of Φ_R is one such gradient, so this component can be written as $\Phi_R^\top a$. Requiring the remaining update to preserve the protected outputs gives

$$\Phi_R(v - \Phi_R^\top a) = 0 \quad \Longrightarrow \quad K_{RR} a = \Phi_R v. \tag{34}$$

The minimum-norm coefficient is $a = K_{RR}^\dagger \Phi_R v$, where $\dagger$ denotes the Moore–Penrose pseudoinverse and allows dependent protected gradients. Thus the allowed update is

$$u = v - \Phi_R^\top K_{RR}^\dagger \Phi_R v = P_R v, \qquad P_R = I - \Phi_R^\top K_{RR}^\dagger \Phi_R. \tag{35}$$

This orthogonal projector satisfies $P_R^\top = P_R$, $P_R^2 = P_R$, and $\Phi_R P_R = 0$. Consequently, any feasible update satisfies $u = P_R u$. Projecting each gradient step gives

$$u_{t+1} = u_t - \eta P_R \nabla_u L_{\mathcal{F}}(\theta_0 + u_t), \qquad u_0 = 0, \tag{36}$$

which keeps all updates in the feasible space.

Substituting $u = P_R u$ into the quadratic loss in Eq. (11) gives, on this space,

$$\Delta L_{\mathcal{F}} \approx \frac{1}{|\mathcal{F}|}\left[-(1 - p_{\mathcal{F}})^\top \Phi_{\mathcal{F}} P_R u + \frac{1}{2} u^\top P_R \Phi_{\mathcal{F}}^\top H_{\mathcal{F}} \Phi_{\mathcal{F}} P_R u + \frac{\mu}{2} u^\top u\right]. \tag{37}$$

At a constrained stationary point, the gradient along the allowed directions is zero. Differentiating and using $P_R u = u$ yields

$$0 = P_R \nabla_u \Delta L_{\mathcal{F}} = \frac{1}{|\mathcal{F}|}\left[-P_R \Phi_{\mathcal{F}}^\top (1 - p_{\mathcal{F}}) + P_R \Phi_{\mathcal{F}}^\top H_{\mathcal{F}} \Phi_{\mathcal{F}} P_R u + \mu u\right]. \tag{38}$$

Rearranging and solving gives

$$(P_R \Phi_{\mathcal{F}}^\top H_{\mathcal{F}} \Phi_{\mathcal{F}} P_R + \mu I) u^\star = P_R \Phi_{\mathcal{F}}^\top (1 - p_{\mathcal{F}}), \tag{39}$$

$$u^\star = (P_R \Phi_{\mathcal{F}}^\top H_{\mathcal{F}} \Phi_{\mathcal{F}} P_R + \mu I)^{-1} P_R \Phi_{\mathcal{F}}^\top (1 - p_{\mathcal{F}}). \tag{40}$$

The inverse exists because $\mu > 0$. Multiplying the stationary equation by $I - P_R$ gives $\mu(I - P_R)u^\star = 0$, confirming that this solution satisfies the constraint. Applying the push-through identity, with $P_R^2 = P_R$, rewrites it as

$$u^\star = P_R \Phi_{\mathcal{F}}^\top (H_{\mathcal{F}} \Phi_{\mathcal{F}} P_R \Phi_{\mathcal{F}}^\top + \mu I)^{-1} (1 - p_{\mathcal{F}}). \tag{41}$$

The projected kernel can now be expanded using $K_{XY} = \Phi_X \Phi_Y^\top$:

$$K_{XY}^P = \Phi_X P_R \Phi_Y^\top = K_{XY} - K_{XR} K_{RR}^\dagger K_{RY}. \tag{42}$$

Finally, multiplying the parameter update by $\Phi_{\mathcal{M}}$ gives the mastered-set drift:

$$\begin{aligned} \widehat{\Delta f_{\mathcal{M}}} &= \Phi_{\mathcal{M}} P_R \Phi_{\mathcal{F}}^\top (H_{\mathcal{F}} \Phi_{\mathcal{F}} P_R \Phi_{\mathcal{F}}^\top + \mu I)^{-1} (1 - p_{\mathcal{F}}) \\ &= K_{\mathcal{M}\mathcal{F}}^P (H_{\mathcal{F}} K_{\mathcal{F}\mathcal{F}}^P + \mu I)^{-1} (1 - p_{\mathcal{F}}). \end{aligned} \tag{43}$$

Projection therefore changes the transfer and curvature-response factors through K^P, while retaining the residual signal $(1 - p_{\mathcal{F}})$.

A.2 REGULARIZATION

For the regularization method defined in Section 3.2, the modified loss adds the quadratic penalty

$$L_{\mathrm{reg}}(\theta_0 + u) = L_{\mathcal{F}}(\theta_0 + u) + \frac{\lambda}{2|\mathcal{F}|} u^\top C u. \tag{44}$$

Combining the penalty with Eq. (11) gives

$$\Delta L_{\mathrm{reg}} \approx \frac{1}{|\mathcal{F}|}\left[-(1-p_{\mathcal{F}})^\top \Phi_{\mathcal{F}} u + \frac{1}{2} u^\top (\Phi_{\mathcal{F}}^\top H_{\mathcal{F}} \Phi_{\mathcal{F}} + \mu I + \lambda C) u\right]. \tag{45}$$

Differentiating this quadratic with respect to u gives

$$\nabla_u \Delta L_{\mathrm{reg}} \approx \frac{1}{|\mathcal{F}|}\left[-\Phi_{\mathcal{F}}^\top (1-p_{\mathcal{F}}) + (\Phi_{\mathcal{F}}^\top H_{\mathcal{F}} \Phi_{\mathcal{F}} + \mu I + \lambda C) u\right]. \tag{46}$$

Since $\mu > 0$ and $C \succeq 0$, the quadratic has a unique stationary point. Setting its gradient to zero and solving gives

$$(\Phi_{\mathcal{F}}^\top H_{\mathcal{F}} \Phi_{\mathcal{F}} + \mu I + \lambda C) u^\star = \Phi_{\mathcal{F}}^\top (1-p_{\mathcal{F}}), \tag{47}$$

$$u^\star = (\Phi_{\mathcal{F}}^\top H_{\mathcal{F}} \Phi_{\mathcal{F}} + \mu I + \lambda C)^{-1} \Phi_{\mathcal{F}}^\top (1-p_{\mathcal{F}}). \tag{48}$$

Applying the push-through identity rewrites the update as

$$u^\star = (\mu I + \lambda C)^{-1} \Phi_{\mathcal{F}}^\top \big(I + H_{\mathcal{F}} \Phi_{\mathcal{F}} (\mu I + \lambda C)^{-1} \Phi_{\mathcal{F}}^\top\big)^{-1} (1-p_{\mathcal{F}}). \tag{49}$$

To express the result in the same form as Eq. (20), use the effective kernel

$$K_{XY}^{\mathrm{Reg}} = \mu \Phi_X (\mu I + \lambda C)^{-1} \Phi_Y^\top. \tag{50}$$

Multiplying the update by $\Phi_{\mathcal{M}}$ and substituting this definition gives

$$\begin{aligned} \widehat{\Delta f_{\mathcal{M}}} &= \frac{K_{\mathcal{M}\mathcal{F}}^{\mathrm{Reg}}}{\mu}\left(I + \frac{H_{\mathcal{F}} K_{\mathcal{F}\mathcal{F}}^{\mathrm{Reg}}}{\mu}\right)^{-1} (1-p_{\mathcal{F}}) \\ &= K_{\mathcal{M}\mathcal{F}}^{\mathrm{Reg}} (H_{\mathcal{F}} K_{\mathcal{F}\mathcal{F}}^{\mathrm{Reg}} + \mu I)^{-1} (1-p_{\mathcal{F}}). \end{aligned} \tag{51}$$

The last step cancels the factor $1/\mu$ against the factor μ from the inverse. Regularization therefore changes the transfer and curvature-response factors through K^{Reg}, while retaining the residual signal.

To establish the conditional equivalence with projection, specialize to $C = F_R$. For $\lambda > 0$ and $H_R \succ 0$, the Woodbury identity gives

$$\mu(\mu I + \lambda F_R)^{-1} = I - \Phi_R^\top \left(K_{RR} + \tfrac{\mu}{\lambda} H_R^{-1}\right)^{-1} \Phi_R. \tag{52}$$

Multiplying on the left by Φ_X and on the right by $\Phi_Y^\top$ gives

$$K_{XY}^{\mathrm{Reg}} = K_{XY} - K_{XR}\left(K_{RR} + \tfrac{\mu}{\lambda} H_R^{-1}\right)^{-1} K_{RY}. \tag{53}$$

This is the soft counterpart of the projected kernel in Eq. (42). To see the projection limit, consider a direction with curvature eigenvalue $\sigma \geq 0$. The operator $\mu(\mu I + \lambda F_R)^{-1}$ scales that direction by $\mu/(\mu + \lambda\sigma)$. As $\lambda \to \infty$, this factor tends to zero for $\sigma > 0$ and remains one for $\sigma = 0$. Since

$$u^\top F_R u = (\Phi_R u)^\top H_R (\Phi_R u), \tag{54}$$

positive-definite H_R makes the zero-curvature directions exactly those satisfying $\Phi_R u = 0$. The limiting operator is therefore P_R, and $K_{XY}^{\mathrm{Reg}} \to K_{XY}^{P}$. A different penalty curvature need not have the same null space or projection limit.

A.3 REPLAY

Replay adds the losses of mastered examples with replay weight $\rho \geq 0$:

$$\begin{aligned} L_{\text{replay}}(\theta) &= L_{\mathcal{F}}(\theta) - \frac{\rho}{|\mathcal{F}|}\mathbf{1}^\top \log p_R(\theta) \\ &= L_{\mathcal{F}}(\theta) + \frac{\rho}{|\mathcal{F}|}\mathbf{1}^\top \log(1+\exp(-f_R(\theta)))\,. \end{aligned} \tag{55}$$

The shared normalization sets the scale of the replay contribution relative to the fine-tuning loss. As in Section 3.1, the summed replay cross-entropy before weighting has output gradient $-(1-p_R)$ and output curvature H_R, evaluated at θ_0. Its second-order expansion therefore gives

$$\Delta L_{\text{replay}} \approx \Delta L_{\mathcal{F}} + \frac{\rho}{|\mathcal{F}|}\left[-(1-p_R)^\top \Delta f_R + \frac{1}{2}\Delta f_R^\top H_R \Delta f_R\right]. \tag{56}$$

Substituting $\Delta f_R \approx \Phi_R u$ and the quadratic for $\Delta L_{\mathcal{F}}$ yields

$$\begin{aligned} \Delta L_{\text{replay}} \approx \frac{1}{|\mathcal{F}|}\Big[&-(1-p_{\mathcal{F}})^\top \Phi_{\mathcal{F}} u - \rho(1-p_R)^\top \Phi_R u \\ &+ \frac{1}{2}u^\top\big(\Phi_{\mathcal{F}}^\top H_{\mathcal{F}}\Phi_{\mathcal{F}} + \mu I + \rho\Phi_R^\top H_R \Phi_R\big)u\Big]. \end{aligned} \tag{57}$$

Differentiating with respect to u gives

$$\begin{aligned} \nabla_u \Delta L_{\text{replay}} \approx \frac{1}{|\mathcal{F}|}\Big[&-\Phi_{\mathcal{F}}^\top(1-p_{\mathcal{F}}) - \rho\Phi_R^\top(1-p_R) \\ &+ \big(\Phi_{\mathcal{F}}^\top H_{\mathcal{F}}\Phi_{\mathcal{F}} + \mu I + \rho\Phi_R^\top H_R\Phi_R\big)u\Big]. \end{aligned} \tag{58}$$

Replay thus adds both curvature and a term that drives the update toward reducing the replay loss. Denote them by

$$G_\rho = \mu I + \rho\Phi_R^\top H_R \Phi_R, \qquad d_\rho = \rho\Phi_R^\top(1-p_R). \tag{59}$$

Setting the quadratic's gradient to zero and solving gives

$$(\Phi_{\mathcal{F}}^\top H_{\mathcal{F}}\Phi_{\mathcal{F}} + G_\rho)u^\star = \Phi_{\mathcal{F}}^\top(1-p_{\mathcal{F}}) + d_\rho, \tag{60}$$

$$u^\star = (\Phi_{\mathcal{F}}^\top H_{\mathcal{F}}\Phi_{\mathcal{F}} + G_\rho)^{-1}\big[\Phi_{\mathcal{F}}^\top(1-p_{\mathcal{F}}) + d_\rho\big]. \tag{61}$$

To obtain the kernel form, first separate the response to d_ρ. Define $u_0 = G_\rho^{-1}d_\rho$ and write $u^\star = u_0 + v$. Substitution into the stationary equation gives

$$(\Phi_{\mathcal{F}}^\top H_{\mathcal{F}}\Phi_{\mathcal{F}} + G_\rho)v + \Phi_{\mathcal{F}}^\top H_{\mathcal{F}}\Phi_{\mathcal{F}}u_0 + G_\rho u_0 = \Phi_{\mathcal{F}}^\top(1-p_{\mathcal{F}}) + d_\rho. \tag{62}$$

Since $G_\rho u_0 = d_\rho$, these terms cancel. Moving the remaining u_0 term to the right gives

$$(\Phi_{\mathcal{F}}^\top H_{\mathcal{F}}\Phi_{\mathcal{F}} + G_\rho)v = \Phi_{\mathcal{F}}^\top\big[(1-p_{\mathcal{F}}) - H_{\mathcal{F}}\Phi_{\mathcal{F}}u_0\big]. \tag{63}$$

Solving for v and applying the same push-through step as for regularization yields

$$\begin{aligned} v &= (\Phi_{\mathcal{F}}^\top H_{\mathcal{F}}\Phi_{\mathcal{F}} + G_\rho)^{-1}\Phi_{\mathcal{F}}^\top\big[(1-p_{\mathcal{F}}) - H_{\mathcal{F}}\Phi_{\mathcal{F}}u_0\big] \\ &= G_\rho^{-1}\Phi_{\mathcal{F}}^\top(I + H_{\mathcal{F}}\Phi_{\mathcal{F}}G_\rho^{-1}\Phi_{\mathcal{F}}^\top)^{-1}\big[(1-p_{\mathcal{F}}) - H_{\mathcal{F}}\Phi_{\mathcal{F}}u_0\big]. \end{aligned} \tag{64}$$

For any evaluation set X, the output drift is $\Phi_X u^\star = \Phi_X u_0 + \Phi_X v$. Define $\mathcal{K}^G_{XY} = \Phi_X G^{-1}\Phi_Y^\top$. Substituting the expression for v then gives, with $G = G_\rho$,

$$\begin{aligned} \widehat{\Delta f_X} &= \Phi_X u_0 + \Phi_X v \\ &= \Phi_X u_0 + \mathcal{K}^G_{X\mathcal{F}}(I + H_{\mathcal{F}}\mathcal{K}^G_{\mathcal{F}\mathcal{F}})^{-1}\big[(1-p_{\mathcal{F}}) - H_{\mathcal{F}}\Phi_{\mathcal{F}}u_0\big]. \end{aligned} \tag{65}$$

Taking $X = \mathcal{M}$ gives the mastered-set drift. Replay changes the transfer and curvature-response factors through G_ρ, adjusts the residual signal by $-H_{\mathcal{F}}\Phi_{\mathcal{F}}u_0$, and adds the output correction $\Phi_{\mathcal{M}}u_0$.

When the aggregate mastered-data gradient vanishes, $\Phi_R^\top(1-p_R) = 0$, we have $d_\rho = 0$ and hence $u_0 = 0$. The linear replay term then disappears from the quadratic loss, leaving only $\rho u^\top F_R u/(2|\mathcal{F}|)$. This is exactly the full-curvature regularization objective with $\lambda = \rho$, so their

local parameter updates and output drifts coincide. The condition concerns the aggregate gradient, not each sample's residual. Centered soft-target replay likewise has zero first derivative and can retain nonzero curvature. If also $H_R \succ 0$, the regularization limit derived above implies that replay approaches the output-preserving projection response as $\rho \to \infty$.

For replay, the appendix expression uses $\mathcal{K}^G_{XY} = \Phi_X G_\rho^{-1} \Phi_Y^\top$. To recover the main-text normalization, set $K^{\mathrm{Replay}}_{XY} = \mu \mathcal{K}^G_{XY}$ and $a_X = \Phi_X u_0$. Then

$$\mathcal{K}^G_{X\mathcal{F}}(I + H_\mathcal{F} \mathcal{K}^G_{\mathcal{F}\mathcal{F}})^{-1} = K^{\mathrm{Replay}}_{X\mathcal{F}}(H_\mathcal{F} K^{\mathrm{Replay}}_{\mathcal{F}\mathcal{F}} + \mu I)^{-1}, \tag{66}$$

Together with $1 - p^{\mathrm{Replay}}_\mathcal{F} = (1 - p_\mathcal{F}) - H_\mathcal{F} a_\mathcal{F}$, this gives Eq. (28).

B Training and implementation details

All experiments run for 10 epochs and use SGD without momentum or weight decay. Text-model response measurements use full-parameter FP32 training with a learning rate of 2×10^{-7} and batch size 8. The text-model forgetting-mitigation experiments use FP32 training with a learning rate of 5×10^{-7} and effective batch size 8. Inference is deterministic with temperature 0.

The multimodal runs use BF16 model parameters and gradients, while probability evaluation and gradient sketches are computed in FP32. For MMBench, we fine-tune *Qwen-VL-3B* using a learning rate of 1×10^{-5} and batch size 4. For MMSU, we fine-tune *Qwen-Omni-7B* using a learning rate of 2×10^{-6} and batch size 1.

Experiments are conducted on NVIDIA A800-SXM4-80GB GPUs using Python 3.10, PyTorch 2.10.0 with CUDA 12.8, Transformers 4.57.6, NumPy 2.2.6, and SciPy 1.15.3.

C Baseline principles and intervention protocol

Method principles. Vanilla SGD minimizes cross-entropy on the fine-tuning examples without an explicit forgetting-mitigation mechanism. Projection methods instead restrict the update using information from mastered examples. GPM constructs a protected subspace from replay-example activations, while OPGD constructs it from mastered-example output gradients; both remove update components within the corresponding subspace.

EWC represents parameter importance through a diagonal empirical Fisher matrix and penalizes changes to parameters that are important for the mastered examples. This provides a soft constraint on the update, in contrast to the hard directional constraints imposed by projection methods.

Replay methods retain mastered examples as part of the training signal. ER jointly optimizes fine-tuning and replay examples with fixed weighting. ER + DWA adapts their relative loss weights according to recent loss changes. DER++ additionally matches the current logits on replay examples to stored initial-model logits. Smart Replay dynamically schedules the replay ratio across minibatches to balance learning and retention under a computational budget.

Parameter-restriction methods reduce interference by limiting where adaptation occurs. O-LoRA freezes the base weights and separates old and new knowledge into low-rank adapters while discouraging overlap between their update subspaces. BLF freezes the lower layers and confines fine-tuning to the upper layers.

Common comparison protocol. All methods use the same train–test partitions and identical base settings for the optimizer, learning rate, batch size, and number of epochs; only the method-specific objective or update rule changes. Within each split, we uniformly subsample 10% of the initially correct training set $\mathcal{M}_{\mathrm{train}}$ once to form a fixed reference buffer. Every method that uses reference examples or quantities derived from them, including projection, regularization, replay, and NTK-Replay, uses exactly the same buffer.

D Additional experimental details for the local approximation

D.1 Method

We compress each per-example correct-answer log-odds gradient to 8192 dimensions using FG-Sketch, as described in Appendix E.1, and construct gradient kernels from inner products between the resulting sketches. The damping coefficient is $\mu = 0.75\lambda_{\max}(H_{\mathcal{F}}^{1/2} K_{\mathcal{F}\mathcal{F}} H_{\mathcal{F}}^{1/2})$.

For the component analysis, we independently shuffle one component while holding the remaining components and damping fixed: we permute the columns of $K_{\mathcal{M}\mathcal{F}}$, jointly permute the rows and columns of $K_{\mathcal{F}\mathcal{F}}$, permute the diagonal of $H_{\mathcal{F}}$, or permute the entries of $1 - p_{\mathcal{F}}$ according to fine-tuning-example indices. We use five independent permutations per data split, average the resulting correlations over these permutations, and then report the mean and sample standard deviation across three splits.

D.2 Additional text-model component-shuffle results

To test whether the component contributions generalize beyond the models reported in the main text, we extend the shuffle analysis to seven additional text models. For each model–dataset split, we follow the protocol in Appendix D.1, average each shuffled correlation over five permutations, and report the mean and sample standard deviation across three data splits. The additional models show the same overall pattern: disrupting the aligned kernel and curvature components generally reduces Pearson correlation, whereas shuffling $1 - p_{\mathcal{F}}$ has a smaller effect.

D.3 Initial correct-answer probabilities

To explain the limited effect of shuffling $1 - p_{\mathcal{F}}$, we examine the initial correct-label probabilities of fine-tuning examples. For *Qwen-1.5B* on MMLU, MedQA, and CSQA, we plot the distribution of $p_{\mathcal{F}}$ and mark the threshold 0.1. Most initially incorrect examples have $p_{\mathcal{F}} < 0.1$, so $1 - p_{\mathcal{F}}$ is concentrated near one and carries relatively little sample-wise variation.

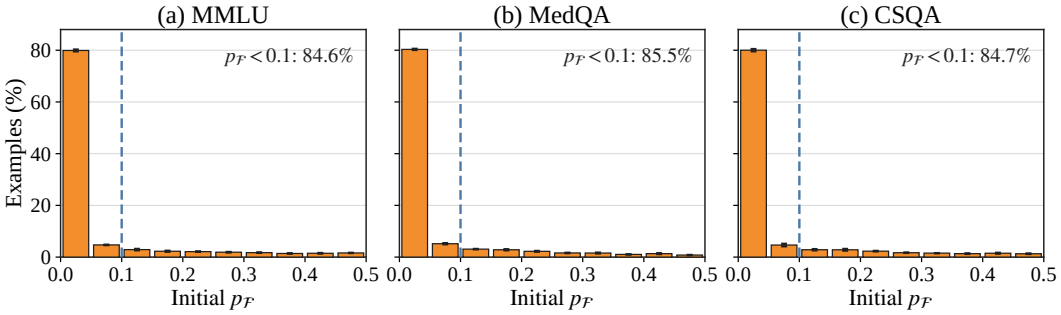

Figure 2: Distribution of the initial correct-label probability $p_{\mathcal{F}}$ for *Qwen-1.5B*. Most values are near zero, making $1 - p_{\mathcal{F}}$ close to one.

D.4 Damping sensitivity for *Qwen-1.5B*

To assess sensitivity to the damping choice, we vary the damping-scale coefficient for *Qwen-1.5B*. We evaluate $\alpha \in \{0.50, 0.75, 1.00, 1.50\}$ in $\mu = \alpha\lambda_{\max}(H_{\mathcal{F}}^{1/2} K_{\mathcal{F}\mathcal{F}} H_{\mathcal{F}}^{1/2})$. Macro Pearson ranges from 0.723 to 0.752, indicating relatively stable performance across the evaluated range.

E Additional experimental details for NTK-Replay

E.1 JL-based gradient-kernel computation

To reduce the cost of constructing per-example gradient kernels, we compress gradients to $k \ll D$ dimensions. CountSketch maps gradient coordinates into k buckets using random hashing and signs

Table 5: Pearson correlations between locally approximated and observed log-odds change. No shuffle denotes direct application of the proposed formula, while Shuffle randomly permutes the entries within the specified component. Green and blue shades indicate decreased ($\Delta \leq -0.001$) and increased ($\Delta \geq 0.001$) correlation relative to no shuffle, respectively.

Model	Perturbation	MMLU Pearson ↑	MedQA Pearson ↑	CSQA Pearson ↑	Macro Pearson ↑
Qwen-3B	No shuffle	0.781 ±0.023	0.861 ±0.003	0.767 ±0.017	0.803 ±0.051
	Shuffle $K_{\mathcal{MF}}$	0.713 ±0.038	0.834 ±0.013	0.750 ±0.020	0.766 ±0.062
	Shuffle $H_{\mathcal{F}}$	0.750 ±0.041	0.847 ±0.012	0.769 ±0.023	0.789 ±0.051
	Shuffle $K_{\mathcal{FF}}$	0.754 ±0.037	0.847 ±0.013	0.765 ±0.020	0.789 ±0.051
	Shuffle $1 - p_{\mathcal{F}}$	0.776 ±0.025	0.860 ±0.003	0.767 ±0.019	0.801 ±0.051
Llama-1B	No shuffle	0.981 ±0.006	0.984 ±0.006	0.876 ±0.104	0.947 ±0.062
	Shuffle $K_{\mathcal{MF}}$	0.950 ±0.013	0.987 ±0.001	0.762 ±0.213	0.900 ±0.121
	Shuffle $H_{\mathcal{F}}$	0.936 ±0.024	0.993 ±0.000	0.791 ±0.188	0.907 ±0.104
	Shuffle $K_{\mathcal{FF}}$	0.977 ±0.006	0.975 ±0.023	0.860 ±0.117	0.937 ±0.067
	Shuffle $1 - p_{\mathcal{F}}$	0.981 ±0.006	0.988 ±0.003	0.877 ±0.104	0.949 ±0.062
Llama-3B	No shuffle	0.933 ±0.003	0.974 ±0.002	0.962 ±0.001	0.956 ±0.021
	Shuffle $K_{\mathcal{MF}}$	0.799 ±0.005	0.947 ±0.001	0.880 ±0.007	0.875 ±0.074
	Shuffle $H_{\mathcal{F}}$	0.888 ±0.003	0.971 ±0.002	0.943 ±0.002	0.934 ±0.042
	Shuffle $K_{\mathcal{FF}}$	0.850 ±0.005	0.953 ±0.001	0.930 ±0.003	0.911 ±0.054
	Shuffle $1 - p_{\mathcal{F}}$	0.924 ±0.003	0.973 ±0.002	0.960 ±0.002	0.952 ±0.025
Gemma-2B	No shuffle	0.538 ±0.017	0.494 ±0.056	0.794 ±0.017	0.609 ±0.162
	Shuffle $K_{\mathcal{MF}}$	0.476 ±0.017	0.355 ±0.050	0.789 ±0.017	0.540 ±0.224
	Shuffle $H_{\mathcal{F}}$	0.508 ±0.016	0.445 ±0.061	0.790 ±0.016	0.581 ±0.184
	Shuffle $K_{\mathcal{FF}}$	0.487 ±0.017	0.357 ±0.039	0.793 ±0.016	0.546 ±0.224
	Shuffle $1 - p_{\mathcal{F}}$	0.533 ±0.018	0.493 ±0.064	0.794 ±0.017	0.607 ±0.163
Gemma-9B	No shuffle	0.564 ±0.026	0.669 ±0.027	0.993 ±0.002	0.742 ±0.224
	Shuffle $K_{\mathcal{MF}}$	0.527 ±0.033	0.636 ±0.046	0.996 ±0.000	0.720 ±0.245
	Shuffle $H_{\mathcal{F}}$	0.549 ±0.026	0.650 ±0.042	0.996 ±0.001	0.732 ±0.234
	Shuffle $K_{\mathcal{FF}}$	0.550 ±0.026	0.655 ±0.039	0.992 ±0.002	0.732 ±0.231
	Shuffle $1 - p_{\mathcal{F}}$	0.550 ±0.032	0.659 ±0.031	0.993 ±0.002	0.734 ±0.231
GLM-9B	No shuffle	0.449 ±0.021	0.688 ±0.010	0.673 ±0.021	0.603 ±0.134
	Shuffle $K_{\mathcal{MF}}$	0.383 ±0.022	0.648 ±0.010	0.599 ±0.014	0.543 ±0.141
	Shuffle $H_{\mathcal{F}}$	0.383 ±0.013	0.653 ±0.013	0.615 ±0.018	0.550 ±0.146
	Shuffle $K_{\mathcal{FF}}$	0.451 ±0.021	0.687 ±0.010	0.666 ±0.023	0.601 ±0.131
	Shuffle $1 - p_{\mathcal{F}}$	0.449 ±0.021	0.688 ±0.010	0.673 ±0.021	0.603 ±0.134
Mistral-7B	No shuffle	0.978 ±0.002	0.735 ±0.034	0.742 ±0.013	0.818 ±0.138
	Shuffle $K_{\mathcal{MF}}$	0.970 ±0.004	0.611 ±0.066	0.744 ±0.010	0.775 ±0.181
	Shuffle $H_{\mathcal{F}}$	0.972 ±0.003	0.722 ±0.032	0.741 ±0.014	0.812 ±0.139
	Shuffle $K_{\mathcal{FF}}$	0.975 ±0.002	0.615 ±0.071	0.745 ±0.013	0.778 ±0.182
	Shuffle $1 - p_{\mathcal{F}}$	0.977 ±0.002	0.732 ±0.034	0.742 ±0.012	0.817 ±0.139

Table 6: RMSE of the full response formula for three representative text models.

Model	MMLU RMSE ↓	MedQA RMSE ↓	CSQA RMSE ↓	Macro RMSE ↓
Qwen-1.5B	3.14 ±0.06	2.17 ±0.04	3.28 ±0.03	2.86 ±0.60
Qwen-7B	2.79 ±0.12	2.17 ±0.05	2.01 ±0.04	2.32 ±0.41
Llama-8B	1.40 ±0.02	0.69 ±0.03	1.07 ±0.02	1.05 ±0.36

Table 7: Pearson correlations for *Qwen-1.5B* under four damping coefficients.

α	MMLU	MedQA	CSQA	Macro average
0.50	**0.761** ± 0.013	**0.832** ± 0.005	**0.663** ± 0.012	**0.752**
0.75	0.747 ± 0.013	0.825 ± 0.006	0.650 ± 0.011	0.741
1.00	0.738 ± 0.014	0.821 ± 0.006	0.642 ± 0.011	0.734
1.50	0.727 ± 0.014	0.814 ± 0.006	0.629 ± 0.010	0.723

(Charikar et al., 2002; Park et al., 2023). This reduces kernel storage and computation, but sketch construction still requires traversing every gradient coordinate. By the chain rule, a linear layer's weight gradient decomposes into a sum of outer products of activations and output derivatives. Our FG-Sketch sketches these factors directly and combines them through TensorSketch convolution, avoiding the construction and traversal of dense weight gradients. For a linear-layer weight matrix $W_l \in \mathbb{R}^{d_o \times d_i}$, the gradient factorizes as

$$\nabla_{W_l} f(x) = \sum_{t=1}^{T_l} \delta_{lt}(x) a_{lt}(x)^\top, \tag{67}$$

where a_{lt} and δ_{lt} are the activation and output derivative at token t. FG-Sketch sketches these factors directly and combines them by TensorSketch convolution:

$$\psi(x) = \sum_l B^{-1}\left[\sum_{t=1}^{T_l} B\big(\mathrm{CS}_\delta(\delta_{lt})\big) \odot B\big(\mathrm{CS}_a(a_{lt})\big)\right] \in \mathbb{R}^k, \tag{68}$$

where CS_a and CS_δ are fixed k-dimensional CountSketch maps. Across linear layers, the sketch-construction costs are

$$C_{\mathrm{CS}} = \Theta\left(\sum_l d_i d_o\right), \qquad C_{\mathrm{FG}} = \Theta\left(\sum_l T_l(d_i + d_o + k\log k)\right). \tag{69}$$

Thus, FG-Sketch avoids scanning dense weight-gradient coordinates while retaining the same $O(Nk)$ feature storage as CountSketch. Let Ψ_X contain one sketch per example in X; the approximate kernel is

$$\widehat{K}_{XY} = \Psi_X \Psi_Y^\top, \qquad [\widehat{K}_{XY}]_{ij} = \psi(x_i)^\top \psi(y_j). \tag{70}$$

Evaluation. We compare CountSketch and FG-Sketch at the same sketch dimension of 8192. Wall-clock time and peak GPU memory are measured over each full dataset. To evaluate compression error, we first compute 5,000 uncompressed per-example gradients, then compress the same gradients with both methods and compare their gradient norms with the uncompressed values. FG-Sketch reduces total time by 14.7%–24.9% and peak memory by 3.4%–21.9%, with 0.76–1.07 percentage points of additional relative gradient-norm error.

Table 8: Wall-clock time for CountSketch and FG-Sketch. Compared with CountSketch, FG-Sketch reduces total wall time by 14.7%–24.9%.

Stage	Method	MMLU Wall time (h) ↓	MedQA Wall time (h) ↓	CSQA Wall time (h) ↓
Computation	CountSketch	0.54 ± 0.00	0.68 ± 0.01	0.29 ± 0.01
	FG-Sketch	0.59 ± 0.00	0.71 ± 0.01	0.37 ± 0.01
Sketching	CountSketch	0.47 ± 0.00	0.39 ± 0.00	0.34 ± 0.00
	FG-Sketch	0.17 ± 0.01	0.21 ± 0.03	0.11 ± 0.00
Total	CountSketch	1.01 ± 0.00	1.08 ± 0.01	0.63 ± 0.01
	FG-Sketch	0.76 ± 0.01	0.92 ± 0.02	0.48 ± 0.02

E.2 Wall-clock efficiency

We record gradient-kernel setup time, the elapsed time at which each NTK-Replay variant surpasses the best accuracy attained by Smart Replay over ten epochs, and total ten-epoch time. Both NTK-

Table 9: Peak allocated GPU memory for CountSketch and FG-Sketch.

Stage	Method	MMLU Peak GPU VRAM (GiB) ↓	MedQA Peak GPU VRAM (GiB) ↓	CSQA Peak GPU VRAM (GiB) ↓
Base	CountSketch	14.8	13.7	13.3
	FG-Sketch	14.8	13.7	13.3
Sketch	CountSketch	4.3	4.3	4.3
	FG-Sketch	3.7	2.9	0.5
Total	CountSketch	19.2	18.1	17.7
	FG-Sketch	18.5	16.6	13.8

Table 10: Relative gradient-norm error computed from 5,000 uncompressed gradients and their CountSketch and FG-Sketch approximations. Compared with CountSketch, FG-Sketch introduces 0.76–1.07 percentage points of additional error.

Method	MMLU Relative norm error (%) ↓	MedQA Relative norm error (%) ↓	CSQA Relative norm error (%) ↓
CountSketch	1.20 ± 0.00	1.27 ± 0.06	1.20 ± 0.10
FG-Sketch	2.03 ± 0.25	2.03 ± 0.21	2.27 ± 0.38

Replay variants surpass this accuracy before Smart Replay completes training on all three datasets, while FG-Sketch reduces setup time relative to CountSketch.

Table 11: Wall-clock cost of Smart Replay and NTK-Replay using CountSketch or FG-Sketch. Setup reports the gradient-kernel preparation time. NTK first exceed reports the cumulative time, including setup and training, until NTK-Replay first surpasses the highest accuracy achieved by Smart Replay over ten epochs. Final reports the total time for completing ten training epochs, including setup where applicable.

Dataset	Method	Setup (h)	NTK first exceed (h)	Final (h)
MMLU	Smart Replay	–	–	2.29
	NTK-Replay (CountSketch)	1.01 ± 0.00	1.26	3.54
	NTK-Replay (FG-Sketch)	0.76 ± 0.01	1.01	3.29
MedQA	Smart Replay	–	–	2.85
	NTK-Replay (CountSketch)	1.08 ± 0.01	1.73	4.33
	NTK-Replay (FG-Sketch)	0.92 ± 0.02	1.57	4.17
CSQA	Smart Replay	–	–	0.73
	NTK-Replay (CountSketch)	0.63 ± 0.01	0.71	1.45
	NTK-Replay (FG-Sketch)	0.48 ± 0.02	0.56	1.30

E.3 Sensitivity to the regularization coefficient

To assess the sensitivity of replay-weight selection to its regularization coefficient, we examine the learning–retention trade-off across different values of ω. We evaluate $\omega \in \{10^{-5}, 10^{-6}, 10^{-7}\}$ on MMLU, MedQA, and CSQA and report Acc and FT over three data splits.

Table 12: Accuracy and forgetting rate of NTK-Replay under three regularization coefficients ω. Best means are bold, and the default setting is shaded.

ω	MMLU		CSQA		MedQA	
	Acc↑	FT↓	Acc↑	FT↓	Acc↑	FT↓
10^{-5}	60.10 ± 0.55	**7.28** ± 1.06	77.19 ± 1.24	**4.25** ± 0.98	46.03 ± 0.97	**9.43** ± 0.65
10^{-6}	**60.35** ± 0.51	8.69 ± 1.40	**77.28** ± 0.84	4.92 ± 0.36	**46.46** ± 1.32	11.24 ± 0.59
10^{-7}	60.34 ± 0.23	10.29 ± 0.73	76.13 ± 0.91	8.74 ± 1.56	46.28 ± 1.15	15.60 ± 0.38