# PaLoRA: Paced Low-Rank Adaptation for Continual Learning

**Yuxuan Li**
School of Computer Science
Peking University
lyxzcx@outlook.com

**Fanhu Zeng**
Institute of Automation
Chinese Academy of Sciences
challengezengfh@gmail.com

**Hao Tang***
School of Computer Science
Peking University
bjdxtanghao@gmail.com

## Abstract

LoRA-based continual learning methods mitigate catastrophic forgetting through various mechanisms, yet nearly all complement these with **small learning rates as a heuristic to restrict gradient scaling magnitude**. Such fixed heuristics lack theoretical guidance on how the strength of this restriction should evolve as tasks accumulate. We reveal that even under directional constraints such as nullspace projection, finite-precision updates inevitably leak into the subspace of accumulated prior knowledge along multiple directions. While small learning rates attenuate such leakage, they cannot prevent the accumulated forgetting from intensifying as the effective rank of historical knowledge grows. We show that the optimal magnitude restriction should adaptively increase with this effective rank to balance **stability and plasticity**, *i.e.*, preservation of previous knowledge and acquisition of new task information. Under an anisotropic leakage model, we derive a pacing law $s^* = \sqrt{R/c}$ that characterizes the optimal **scaling of gradient steps,** *i.e.*, the magnitude restriction itself, where $R$ is the effective rank of past updates. Based on this insight, we propose PaLoRA, which compresses historical knowledge via adaptive SVD truncation, projects gradients onto the nullspace of prior tasks, and applies rank-aware adaptive pacing. Experiments demonstrate consistent improvements over prior methods, with particularly strong performance in long-horizon settings, achieving substantial gains of 4% accuracy on challenging 50-task ImageNet-A and ImageNet-R benchmarks. The code is available at: https://github.com/liyuxuan-github/PaLoRA

## 1 Introduction

Continual learning [37, 35, 48, 60, 12], which refers to the ability to learn sequentially from a stream of tasks without catastrophic forgetting [40, 23, 8], remains a fundamental challenge in machine learning. With the rise of large pre-trained vision transformers (ViTs) [6, 24, 13] as standard backbones for visual recognition [47, 45, 25, 30, 5], parameter-efficient fine-tuning methods [29, 17, 7], particularly Low-Rank Adaptation (LoRA) [17], have become a prominent paradigm for continual adaptation. By learning compact, task-specific low-rank updates while keeping pre-trained weights frozen, LoRA-based continual learning methods [31, 34] achieve an appealing balance between computational efficiency and representational capacity.

Existing LoRA-based continual learning approaches mitigate catastrophic forgetting primarily through directional constraints, *e.g.*, restricting new task updates to the null space of previously learned knowledge [49, 31, 61]. While these methods achieve promising results, they focus on constraining the *direction* of parameter updates while largely overlooking the *magnitude*, *i.e.*, how large the update step is. In practice, nearly all these approaches [49, 31, 61, 34] further adopt small learning

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

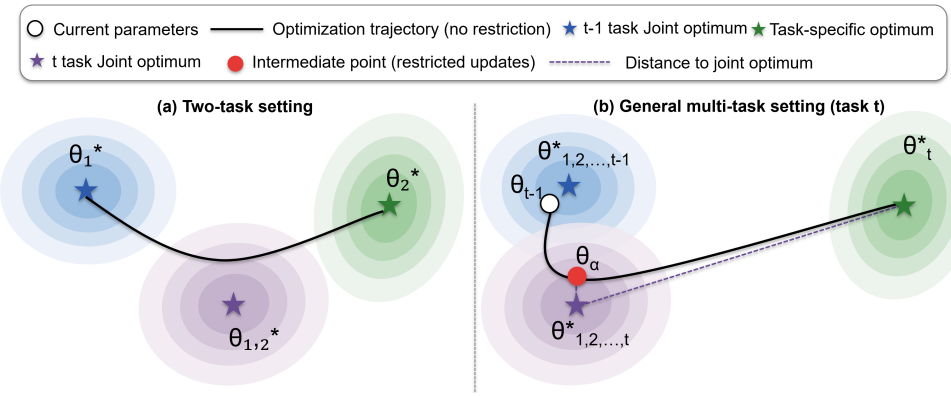



Figure 1: Adaptive parameter update restrictions progressively shift optimization trajectory toward previous task solutions as more tasks are encountered, better balancing stability and plasticity to approach the joint optimum.

rates [58], which equivalently restrict the magnitude of updates, as an additional safeguard to enhance stability. However, **(1) why restricting update magnitude mitigates forgetting** remains theoretically unexplored. More importantly, as tasks accumulate, the volume and complexity of historical knowledge increase, rendering a fixed restriction strength increasingly suboptimal; **(2) therefore, how such restriction should adapt as tasks accumulate** has received no principled investigation.

We illustrate the optimization trajectories in Figure 1 to explain why restricting update magnitude mitigates forgetting. Without constraints, fine-tuning tends to move the model toward the task-specific optimum of the new task, which may deviate from the joint optimum over all tasks. Restricting updates instead limits excessive parameter shifts and keeps optimization closer to the joint solution, achieving a better balance between plasticity for learning new knowledge and stability for preserving previous knowledge. As tasks accumulate in Figure 1b, the current parameters encode increasing historical knowledge, reflected by the effective rank of accumulated updates, while the optimum of a single new task becomes less representative of the overall objective. This progressively shifts the trade-off toward stability, suggesting that the restriction strength should be adaptively increased with the effective rank rather than kept fixed.

To address this, we analyze rank-aware scaling, i.e., magnitude restriction for LoRA updates, in a single-layer linear model under anisotropic leakage and local quadratic old-task sensitivity. We derive a *pacing law* $s^* = \sqrt{R/c}$ for the **gradient-step denominator**, where $R$ is the effective rank of accumulated updates and $c$ sets the per-step forgetting budget relative to the average per-direction leakage floor. Under this model, $s^*$ maximizes first-order new-task progress subject to the budget: weaker restriction violates the budget, while stronger restriction sacrifices progress beyond what the budget requires. For deep ViTs, this result motivates a scaling principle rather than a formal guarantee.

Building upon this foundation, we propose Paced LoRA (PaLoRA), which compresses accumulated updates via adaptive SVD truncation to obtain effective rank $R$, projects new-task gradients onto the historical nullspace to reduce interference, and divides the gradient of the LoRA factor $B_t$ by $\sqrt{R/C}$ to instantiate rank-aware pacing. An empirically motivated feature-space regularizer complements these constraints by promoting current-task discrimination; the pacing-law optimality claim does not extend to this full regularized framework.

We evaluate PaLoRA on CIFAR-100, ImageNet-R, and ImageNet-A under different task splits. In the five-seed main comparisons, PaLoRA achieves the highest final and average accuracy across all splits. At 50 tasks, its final accuracy exceeds the strongest baseline in these tables, LoRA$^-$DRS, by 4.09 percentage points on ImageNet-R and 3.99 points on ImageNet-A. From 10 to 50 tasks, PaLoRA's final accuracy decreases by only 2.19 and 0.65 points on these datasets, respectively.

Our contributions are threefold: (1) under the stated leakage and local sensitivity assumptions, we establish how update restriction should grow with historical effective rank; (2) we derive the $\sqrt{R/c}$ pacing law for the budget-constrained model and instantiate this principle in PaLoRA through adaptive rank compression, nullspace projection, and rank-aware gradient scaling; and (3) we demonstrate

strong performance on three benchmarks, including long-horizon (50-task) continual learning, with five-seed main comparisons.

## 2 Related Work and Preliminaries

### 2.1 Related Work

**Parameter-Efficient Fine-Tuning** (PEFT) aims to adapt pre-trained models to downstream tasks with minimal trainable parameters while keeping the pre-trained weights frozen [17, 20]. Recent advances in PEFT have led to several distinct categories. For instance, selective methods [28, 53] fine-tune only a subset of parameters; additive methods [38, 46, 21] insert lightweight adapter modules into intermediate layers; prompt-based methods [29, 55, 33] optimize soft prompt vectors; and reparameterization-based methods [59, 26, 32] sparsify weight matrices to reduce the number of trainable parameters. Among these, Low-Rank Adaptation (LoRA) [17] substantially reduces trainable parameters through low-rank decomposition of weight matrices and incurs no additional inference overhead, making it widely adopted [56, 43, 42] and continuously developed [14, 22].

**Continual Learning** seeks to address catastrophic forgetting when models learn from sequentially arriving data streams [11]. Early work [39, 2, 1, 3] primarily focused on retaining a small subset of historical data for rehearsal to prevent the model from forgetting past knowledge. However, due to storage constraints and privacy concerns, example-free methods have also attracted significant attention. For instance, LwF [4] constrains the model's outputs to remain close to those of the historical model, while Gradient Projection Memory (GPM) [41] reduces interference with prior knowledge via gradient orthogonalization.

With the advancement of PEFT, leveraging pre-trained weights for continual learning has gained increasing popularity. For example, L2P [51] and its subsequent [50, 44, 57, 19] variants employ multiple trainable prompts to preserve historical knowledge, thereby avoiding explicit storage of past data. Meanwhile, by harnessing knowledge from pre-trained weights, these methods better distinguish between different tasks with promising results. Recently, LoRA-based continual learning methods have been extensively studied [9, 10, 31, 52]. For instance, O-LoRA [49] constrains the orthogonality of parameters across different tasks through regularization; BiLoRA [61], based on FourierFT [7], assigns nearly disjoint entries to different tasks to minimize mutual interference. Nevertheless, these methods only consider isolation of learning directions across tasks, and in practice, gradient leakage along these directions still causes forgetting to increase as the number of tasks grows.

### 2.2 Preliminaries

**Problem Definition.** We consider *example-free class-incremental learning* (EFCIL), where a model is trained on a sequence of tasks $\{\mathcal{T}_t\}_{t=1}^{T}$ without access to past data. Each task $\mathcal{T}_t$ introduces a disjoint set of classes $\mathcal{Y}_t$, and we denote the cumulative label space up to step $t$ as $\mathcal{Y}_{1:t} = \bigcup_{\tau=1}^{t} \mathcal{Y}_\tau$.

At training time $t$, the learner has access only to samples from the current distribution $\mathcal{D}_t$ over $\mathcal{X} \times \mathcal{Y}_t$, and no data from previous tasks. At inference time, the model must predict over the entire label space $\mathcal{Y}_{1:T}$ *without access to task identity*.

Let $f(x; W) \in \mathbb{R}^{|\mathcal{Y}_{1:T}|}$ denote the model. The objective at step $t$ is to minimize:

$$\mathcal{L}_t(W) = \mathbb{E}_{(x,y)\sim\mathcal{D}_t}[\ell(f(x; W), y)],$$

while maintaining performance on previously learned classes.

**Low-Rank Adaptation.** We consider low-rank adaptation (LoRA) [17] for parameter-efficient fine-tuning of pre-trained models. Given a frozen pre-trained weight matrix $W_0 \in \mathbb{R}^{d\times d}$, LoRA models task-specific adaptation by adding a low-rank update:

$$W = W_0 + BA,$$

where $A \in \mathbb{R}^{r\times d}$ and $B \in \mathbb{R}^{d\times r}$ with rank $r \ll d$. This parameterization restricts learning to a low-dimensional subspace while keeping the original model weights fixed.

# 3 Methodology

## 3.1 On the Role of Restricted Updates in Continual Learning

Recently, LoRA-based methods [31, 61, 34] often follow the setting of SLCA [58] that adopt a small learning rate during training. Similarly, related work [18] has observed that reducing the number of training epochs yields comparable improvements in continual learning performance. However, there is limited understanding of why restricted updates can effectively mitigate forgetting.

In this section, we provide an intuitive explanation based on the perspective of model update trajectories. Specifically, we argue that restricting the magnitude of parameter movement helps reduce interference with previously learned representations. Furthermore, this restriction should be adaptively strengthened as the number of tasks increases, since accumulated updates progressively increase the risk of cross-task interference.

We first illustrate our key insight using a two-task setting in Figure 1a. Let $\theta_1^*$ and $\theta_2^*$ denote the optima obtained by training on task 1 and task 2 separately, and $\theta_{1,2}^*$ the joint optimum from simultaneous training. After convergence on task 1, unrestricted training on task 2 leads to $\theta_2^*$, which can be far from $\theta_{1,2}^*$. By restricting parameter updates, the model is constrained to an intermediate point along the trajectory between $\theta_1^*$ and $\theta_2^*$, potentially achieving a better balance between preserving previous knowledge and acquiring new knowledge.

This principle extends to the general $t$-task setting in Figure 1b. After learning $t-1$ tasks, the model converges to $\theta_{t-1}$. Unrestricted training on task $t$ leads to the task-specific optimum $\theta_t^*$, deviating from the joint optimum $\theta_{1,2,\ldots,t}^*$. Restricted updates constrain the trajectory, yielding an intermediate point $\theta_\alpha$. We denote by $\theta_{\alpha^*}$ the point on this constrained trajectory closest to the true joint optimum. As the number of tasks increases, $\theta_{t-1}$ becomes increasingly representative of performance on all previous tasks, while $\theta_t^*$ reflects only the current task. Consequently, the optimal trade-off point $\theta_{\alpha^*}$ shifts closer to $\theta_{t-1}$.

This insight suggests that the strength of update restriction should not remain fixed, as is commonly assumed in prior work using small learning rates. Instead, it should be adaptively increased as the number of tasks grows, to better balance stability on previous tasks and plasticity for new tasks.

## 3.2 Theoretical Analysis: Rank-Aware Scaling for LoRA-Based Continual Learning

**Setup.** Consider a single linear layer with frozen pre-trained weights $W_0 \in \mathbb{R}^{d\times d}$. After learning tasks $1,\ldots,t-1$ via LoRA, the accumulated low-rank updates form an effective adaptation $C_{t-1} \in \mathbb{R}^{d\times d}$ with $\mathrm{rank}(C_{t-1}) = R$. The current effective weight is $W^{(t-1)} = W_0 + C_{t-1}$. Task $t$ introduces an additional LoRA increment $\Delta_t$, and the updated weight becomes $W^{(t)} = W^{(t-1)} + \Delta_t$. Each task $t$ is equipped with an input design matrix $X_t \in \mathbb{R}^{d\times n_t}$ with positive-definite covariance $\Sigma_t = X_t X_t^\top \succ 0$. The task-$t$ objective is the Frobenius-norm regression loss

$$\mathcal{L}_t(\Delta_t) = \frac{1}{2}\|(W^{(t-1)} + \Delta_t)X_t - Y_t\|_F^2,$$

optimized via full-batch gradient descent with fixed-step projected gradient descent.

**Nullspace Projection and Leakage.** To prevent forgetting, the new LoRA increment $\Delta_t$ is ideally restricted to the right nullspace of the accumulated adaptation $C_{t-1}$. Let $C_{t-1} = U\Sigma V^\top = \sum_{i=1}^R \sigma_i u_i v_i^\top$ be the compact SVD, and let $V_\perp = [v_{R+1},\ldots,v_d]$ span the orthogonal complement of the row space of $C_{t-1}$. The feasible subspace for $\Delta_t$ is $\mathcal{N}_t \triangleq \{MV_\perp^\top : M \in \mathbb{R}^{d\times(d-R)}\}$, which ensures $\Delta_t x_{\mathrm{old}} = 0$ for all old-task inputs $x_{\mathrm{old}} \in \mathrm{span}(V_{[:R]})$ and preserves old-task forward passes under exact arithmetic (see Section A.1).

Finite-precision arithmetic and imperfect projection motivate a leakage model in the old-task subspace $\mathcal{S} = \mathrm{span}\{u_i v_i^\top\}_{i=1}^R$. Our derivation relies on two core assumptions: (i) anisotropic leakage with $\mathbb{E}[\epsilon_i^2] = \eta^2\gamma_i^2/s^2$, where $\eta$ is the base learning rate, $\gamma_i$ the direction-dependent leakage scale, and $s > 0$ the step-size denominator; and (ii) locally quadratic old-task loss with curvature $\lambda_i > 0$ along each historical singular direction (Assumptions A.1–A.2).

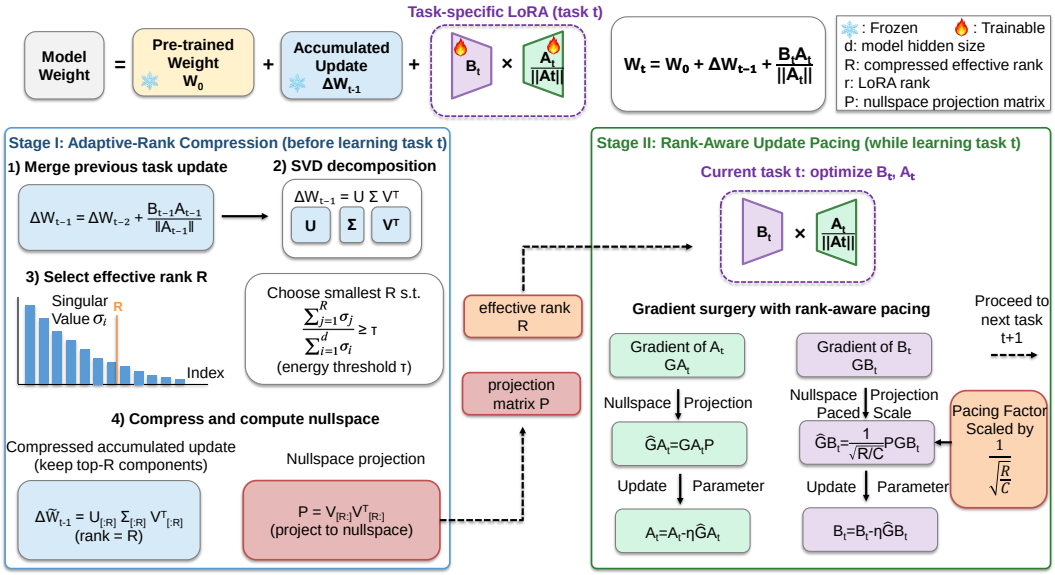



Figure 2: **Overview of PaLoRA.** (A) Model decomposition: pretrained weights and compressed memory from previous tasks are frozen, while a new low-rank update is learned for the current task. (B) Two-stage process: adaptive-rank compression of accumulated knowledge via SVD, followed by rank-aware gradient pacing with nullspace projection and scaled updates.

**The Forgetting–Progress Trade-off.** Under these assumptions, the per-step expected old-task loss increase and first-order new-task loss change are:

$$\mathbb{E}[\Delta\mathcal{L}_{t-1}] = \frac{\eta^2 R\Gamma^2}{2s^2}, \qquad \Gamma^2 \triangleq \frac{1}{R}\sum_{i=1}^{R}\lambda_i\gamma_i^2, \tag{1}$$

$$\Delta\mathcal{L}_t \approx -\frac{\eta G^2}{s}, \qquad G^2 = \|P_{\mathcal{N}_t}(\nabla_{\Delta_t}\mathcal{L}_t)\|_F^2. \tag{2}$$

Here, $\Gamma^2$ averages the leakage–curvature products, $P_{\mathcal{N}_t}(M) = MV_\perp V_\perp^\top$ is the orthogonal projection onto $\mathcal{N}_t$, and $G^2$ is the squared norm of the projected gradient; new-task progress is $-\Delta\mathcal{L}_t$. To see the trade-off, local quadratic sensitivity gives $\mathbb{E}[\Delta\mathcal{L}_{t-1}] \approx \frac{1}{2}\sum_i \lambda_i \mathbb{E}[\epsilon_i^2]$, yielding Eqn. (1), while a first-order Taylor expansion along the projected step gives Eqn. (2). Thus, increasing $s$ suppresses forgetting quadratically but reduces progress only linearly. Full proofs are provided in Section A.4.

**Forgetting Budget.** We maximize new-task progress subject to a per-step expected forgetting cap $\epsilon_0 = c\,\epsilon_{\text{dir}}$, where $\epsilon_{\text{dir}} = \eta^2\Gamma^2/2$ is the average per-direction forgetting from an unscaled step and $c > 0$ is a tolerance coefficient. The optimum is the smallest $s$ satisfying this cap.

**Lemma 3.1** (The $\sqrt{R/c}$ Scaling Law)**.** *Under the stated model, with budget $\epsilon_0 = c\,\eta^2\Gamma^2/2$, the optimal scale simplifies to:*

$$s^* = \sqrt{R/c}\,. \tag{3}$$

**Properties.** Substituting $s^* = \sqrt{R/c}$ into Eqn. (1) shows that the per-step forgetting exactly equals the prescribed budget $c\,\epsilon_{\text{dir}}$, regardless of the specific directions in the rank-$R$ subspace. Any $s < \sqrt{R/c}$ violates the budget; any $s > \sqrt{R/c}$ is suboptimal by Eqn. (2). Hence $s^*$ is the unique Pareto-optimal operating point for this budget-constrained pacing problem under the stated model. Full derivations are in Section A.5.

### 3.3 PaLoRA Framework

Motivated by the scaling law $s^* = \sqrt{R/c}$ under the stated model (Lemma 3.1), we now present the practical PaLoRA framework, illustrated in Figure 2. Our design translates the rank-aware scaling principle into a two-stage algorithm: adaptive-rank compression of accumulated knowledge, followed

by paced gradient updates governed by the effective rank $R$. Following LoRA-style parameterization, we learn a sequence of low-rank factors $\{A_i, B_i\}$ with $\mathrm{rank}$ r to encode task-specific knowledge. During training on task $t$, the pretrained weights $W_0$ and the accumulated update $\Delta W_{t-1}$ are frozen, and only $(A_t, B_t)$ are optimized.

A key distinction of our method is that we normalize $A_t$, enforcing it to capture only directional changes, while $B_t$ controls the magnitude of the update. Since $B_t$ is initialized to zero, scaling its gradient provides a direct mechanism to regulate the learning dynamics of the entire task-specific update.

**Stage I: Adaptive-Rank Compression.** Before learning task $t$, we merge task t-1 updates $\frac{B_{t-1}A_{t-1}}{\|A_{t-1}\|}$ into the accumulated matrix $\Delta W_{t-2}$ and perform SVD decomposition. While assuming a fixed rank growth of $(t-1)r$ is often suboptimal due to redundancy knowledge across tasks and varying task difficulty.

To address this, we define an *effective rank* $R$ by retaining the top singular values whose cumulative energy exceeds a predefined threshold $\tau$. We then reconstruct a compressed representation of $\Delta W_{t-1}$ using these $R$ components and compute its associated nullspace projection matrix $P$. This procedure adaptively removes redundant directions and yields a compact representation of past knowledge.

**Stage II: Rank-Aware Update Pacing.** During training on task $t$, we use the effective rank $R$ and projection matrix $P$ to constrain parameter updates. The gradient of $A_t$ is projected onto the nullspace of $\Delta W_{t-1}$ to reduce interference with previous tasks. For $B_t$, we additionally apply gradient scaling by $\sqrt{R/C}$, where $C$ is a budget constant. This induces a rank-dependent pacing mechanism that adaptively controls the trade-off between stability and plasticity.

**Plasticity Compensation via Regularization.** While nullspace projection reduces interference with previously learned knowledge, it may also restrict the model's ability to learn new tasks. To compensate for this loss of plasticity, we adopt a regularization term inspired by the easy-positive triplet loss [54].

Specifically, for each anchor, we select the closest positive and negative samples to construct a margin-based objective:

$$\mathcal{L}_{\text{reg}} = \frac{1}{N}\sum_{i=1}^{N}\left[d_i^+ - d_i^- + m\right]_+, \tag{4}$$

where $d_i^+$ and $d_i^-$ denote the minimum Euclidean distances to samples of the same and different classes, respectively. In our implementation, the positive set includes the anchor itself, making $d_i^+$ exactly zero. Consequently, the regularizer primarily enforces a margin against the nearest negative sample, encouraging inter-class separation under constrained parameter updates. The training objective $\mathcal{L}_{\text{total}}$ combines the cross-entropy loss $\mathcal{L}_{\text{CE}}$ and the regularization term, balanced by a weighting factor $\lambda$:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CE}} + \lambda\mathcal{L}_{\text{reg}}.$$

**Connection to Theory.** Nullspace projection defines the feasible update subspace, adaptive SVD provides a theory-motivated estimate of the historical effective rank $R$, and pacing instantiates the derived scaling, with $C$ serving as the practical budget coefficient. The triplet loss is an empirically motivated feature-space plasticity compensator; it is not derived from the pacing law, whose Pareto-optimality claim does not cover the full regularized system. For deep ViTs, the analysis motivates an approximate local scaling principle for LoRA updates rather than a formal guarantee for the full nonlinear network.

# 4 Experiments

## 4.1 Experimental Setup

**Datasets and Evaluation Metrics.** We evaluate our method on three benchmark datasets: CIFAR-100 [27], ImageNet-R [15], and ImageNet-A [16]. Following prior work [34], we split each dataset into task sequences with 10, 20, 25, and 50 tasks to evaluate performance under different continual learning scenarios.

Table 1: ImageNet-R accuracy (%, mean ± standard deviation over five seeds). PaLoRA achieves the highest final and average accuracy across all splits, with a 4.09-point final-accuracy gain over LoRA$^-$DRS at 50 tasks.

| Tasks | 10 | | 20 | | 25 | | 50 | |
|---|---|---|---|---|---|---|---|---|
| Method | $ACC_{10}$ | $\overline{ACC_{10}}$ | $ACC_{20}$ | $\overline{ACC_{20}}$ | $ACC_{25}$ | $\overline{ACC_{25}}$ | $ACC_{50}$ | $\overline{ACC_{50}}$ |
| Pretrain | 55.68±0.00 | 59.20±0.00 | 55.68±0.00 | 59.55±0.00 | 55.68±0.00 | 59.78±0.00 | 55.68±0.00 | 59.95±0.00 |
| Sequence | 62.11±2.21 | 73.39±0.97 | 57.30±0.39 | 69.56±0.69 | 54.48±2.16 | 66.98±0.28 | 47.48±0.05 | 62.94±0.60 |
| L2P [51] | 62.28±0.30 | 66.98±0.04 | 61.24±0.16 | 64.73±0.43 | 61.03±0.35 | 64.37±0.37 | 59.44±0.45 | 63.16±0.57 |
| DualPrompt [50] | 58.91±0.23 | 62.66±0.06 | 57.78±0.33 | 61.91±0.14 | 57.53±0.64 | 61.66±0.10 | 56.28±0.97 | 61.72±0.22 |
| CODA-Prompt [44] | 69.81±0.06 | 73.00±0.15 | 66.42±0.13 | 69.55±0.35 | 65.03±0.23 | 68.22±0.18 | 56.19±0.16 | 59.01±0.27 |
| InfLoRA [31] | 74.25±0.74 | 80.35±0.31 | 72.62±0.92 | 78.87±0.03 | 71.42±1.44 | 78.11±0.36 | 66.52±1.13 | 73.09±0.67 |
| BiLoRA [61] | 72.73±0.23 | 76.38±0.11 | 72.38±0.21 | 76.24±0.25 | 72.51±0.37 | 76.12±0.25 | 71.56±0.54 | 75.28±0.15 |
| LoRA$^-$DRS [34] | 75.56±0.23 | 81.42±0.06 | 75.12±0.08 | 80.67±0.45 | 74.68±0.07 | 79.93±0.24 | 72.47±0.56 | 77.19±0.64 |
| **PaLoRA (Ours)** | **78.75±0.11** | **83.08±0.16** | **78.84±0.39** | **83.22±0.67** | **78.21±0.16** | **82.85±0.07** | **76.56±0.23** | **80.68±0.39** |

Table 2: CIFAR-100 accuracy (%, mean ± standard deviation over five seeds). PaLoRA consistently improves both metrics, reaching $88.96 \pm 0.14$ final accuracy at 50 tasks.

| Tasks | 10 | | 20 | | 25 | | 50 | |
|---|---|---|---|---|---|---|---|---|
| Method | $ACC_{10}$ | $\overline{ACC_{10}}$ | $ACC_{20}$ | $\overline{ACC_{20}}$ | $ACC_{25}$ | $\overline{ACC_{25}}$ | $ACC_{50}$ | $\overline{ACC_{50}}$ |
| Pretrain | 83.81±0.00 | 88.82±0.00 | 83.81±0.00 | 89.13±0.00 | 83.81±0.00 | 89.16±0.00 | 83.81±0.00 | 89.33±0.00 |
| Sequence | 87.54±0.94 | 92.12±0.78 | 81.51±2.60 | 89.05±0.48 | 81.93±4.13 | 88.95±1.11 | 77.52±5.94 | 85.01±4.23 |
| L2P [51] | 84.53±0.71 | 87.84±0.39 | 81.00±0.51 | 84.70±0.94 | 78.45±0.37 | 82.08±0.08 | 74.48±1.35 | 78.90±1.78 |
| DualPrompt [50] | 75.37±2.35 | 86.03±0.66 | 77.79±0.04 | 86.49±0.52 | 79.72±0.85 | 86.82±0.12 | 81.11±0.77 | 87.37±0.05 |
| CODA-Prompt [44] | 84.40±0.13 | 88.77±0.38 | 81.69±1.03 | 85.34±0.70 | 79.10±1.12 | 82.19±0.87 | 72.16±0.08 | 82.65±0.49 |
| InfLoRA [31] | 86.51±1.13 | 91.51±0.30 | 82.40±5.13 | 89.17±1.62 | 82.32±2.04 | 88.62±1.48 | 76.04±1.92 | 83.31±2.78 |
| BiLoRA [61] | 87.01±0.61 | 90.69±0.50 | 84.91±0.70 | 88.96±0.85 | 85.32±1.16 | 89.66±1.83 | 79.99±3.82 | 85.51±3.04 |
| LoRA$^-$DRS [34] | 89.55±0.10 | 92.83±0.01 | 88.69±0.01 | 92.57±0.01 | 88.08±0.09 | 92.23±0.01 | 87.02±0.15 | 91.45±0.25 |
| **PaLoRA (Ours)** | **91.02±0.05** | **94.00±0.01** | **90.26±0.07** | **93.64±0.04** | **90.40±0.21** | **93.68±0.02** | **88.96±0.14** | **92.51±0.02** |

Following previous studies [31, 61, 34], we adopt $ACC_T$ and $\overline{ACC_T}$ as evaluation metrics. $ACC_T$ denotes the average accuracy over all tasks after learning the final task $T$, while $\overline{ACC_T}$ represents the average of $ACC_i$ computed after each task $i$ is learned. These metrics jointly reflect the final performance and the stability of the model throughout the continual learning process. Following prior work [34], we further report Backward Transfer (BWT) to quantify forgetting. BWT is computed as the average difference between the accuracy of each task after learning the final task $T$ and its accuracy immediately after it is learned, thereby measuring how much previously acquired knowledge is retained during continual learning.

The main accuracy tables report means and standard deviations over five random seeds (0–4). Other tables and the existing accuracy curves report point estimates. Implementation details and additional analyses are provided in Appendix B.

**Compared Methods.** We compare our method with several LoRA-based continual learning approaches, including InfLoRA [31], BiLoRA [61], and LoRA$^-$DRS [34]. In addition, we include prompt-based methods such as L2P [51], DualPrompt [50], and CODA-Prompt [44]. We additionally compare with SD-LoRA [52] in Section 4.3.

We further introduce two additional baselines: **Pretrain** and **Sequence**. **Pretrain** directly uses a frozen pretrained model without any task adaptation, representing full stability but no plasticity. **Sequence** applies standard LoRA-based fine-tuning across tasks without any mechanism to mitigate forgetting, representing full plasticity but suffering from catastrophic forgetting.

### 4.2 Main Results

**Overall Performance.** Tables 1, 2, and 3 report five-seed results on ImageNet-R, CIFAR-100, and ImageNet-A. PaLoRA achieves the highest mean final and average accuracy in all settings. On ImageNet-R, its final-accuracy gain over LoRA$^-$DRS ranges from 3.19 to 4.09 percentage points, reaching $76.56 \pm 0.23\%$ at 50 tasks. On CIFAR-100, the corresponding gains range from 1.47 to 2.32 points. On ImageNet-A, PaLoRA improves final accuracy by 2.46–4.05 points, with a 3.99-point

Table 3: ImageNet-A accuracy (%, mean ± standard deviation over five seeds). PaLoRA leads across all splits and exceeds LoRA$^-$DRS by 3.99 points in final accuracy at 50 tasks.

| Tasks | 10 | | 20 | | 25 | | 50 | |
|---|---|---|---|---|---|---|---|---|
| Method | $ACC_{10}$ | $\overline{ACC_{10}}$ | $ACC_{20}$ | $\overline{ACC_{20}}$ | $ACC_{25}$ | $\overline{ACC_{25}}$ | $ACC_{50}$ | $\overline{ACC_{50}}$ |
| Pretrain | 48.85±0.00 | 57.39±0.00 | 48.85±0.00 | 58.47±0.00 | 48.85±0.00 | 58.56±0.00 | 48.85±0.00 | 58.97±0.00 |
| Sequence | 34.13±1.44 | 53.86±2.04 | 23.87±0.23 | 46.07±1.37 | 25.97±3.78 | 43.62±0.24 | 15.61±8.38 | 37.62±1.41 |
| L2P [51] | 36.08±0.65 | 41.66±0.87 | 33.55±0.70 | 38.95±1.18 | 33.64±0.56 | 39.08±0.07 | 32.39±0.65 | 37.55±0.71 |
| DualPrompt [50] | 42.42±3.12 | 53.20±0.83 | 48.09±0.14 | 56.99±0.01 | 48.02±0.33 | 57.19±0.01 | 47.66±0.84 | 57.69±0.47 |
| CODA-Prompt [44] | 52.17±0.23 | 62.64±0.39 | 42.59±0.65 | 49.77±1.08 | 39.30±0.37 | 47.66±0.32 | 47.53±0.28 | 58.05±0.21 |
| InfLoRA [31] | 55.47±1.72 | 68.04±1.64 | 51.61±0.93 | 65.89±0.12 | 50.56±3.45 | 65.67±1.62 | 45.30±3.91 | 60.28±0.39 |
| BiLoRA [61] | 56.45±0.04 | 67.54±0.52 | 56.32±0.69 | 67.25±0.93 | 57.24±1.63 | 69.19±0.75 | 56.12±0.98 | 69.55±1.07 |
| LoRA$^-$DRS [34] | 59.09±1.35 | 70.35±0.93 | 58.50±1.07 | 69.77±0.27 | 58.62±0.33 | 69.52±0.01 | 56.91±0.23 | 68.22±0.28 |
| **PaLoRA (Ours)** | **61.55±0.56** | **72.91±0.15** | **61.66±0.23** | **72.43±0.37** | **62.67±0.18** | **73.62±0.57** | **60.90±0.28** | **72.41±0.16** |

Table 4: Backward transfer (BWT, %; higher is better) on all three datasets under 25- and 50-task splits. PaLoRA yields the least forgetting on CIFAR-100 and ImageNet-A, while LoRA$^-$DRS has slightly better BWT on ImageNet-R. Values are point estimates.

| Method | CIFAR-100 | | ImageNet-R | | ImageNet-A | |
|---|---|---|---|---|---|---|
| | 25 Tasks | 50 Tasks | 25 Tasks | 50 Tasks | 25 Tasks | 50 Tasks |
| Sequence | -16.24 | -14.72 | -36.86 | -42.14 | -31.62 | -53.83 |
| L2P [51] | -8.94 | -14.07 | -9.43 | -10.56 | -6.80 | -11.16 |
| DualPrompt [50] | -8.35 | -6.46 | -8.37 | -9.72 | -6.36 | -8.79 |
| CODA-Prompt [44] | -9.83 | -17.11 | -6.02 | -9.76 | -10.71 | -9.46 |
| InfLoRA [31] | -12.53 | -20.45 | -15.08 | -20.01 | -16.47 | -18.34 |
| BiLoRA [61] | -6.83 | -17.48 | -5.36 | -6.58 | -6.66 | -8.08 |
| LoRA$^-$DRS [34] | -4.64 | -5.26 | **-4.78** | **-4.92** | -6.06 | -9.01 |
| **PaLoRA (Ours)** | **-3.78** | **-4.72** | -5.34 | -5.20 | **-4.87** | **-7.09** |

gain at 50 tasks. The five-seed results support the consistency of these improvements, including in long task sequences.

**Accuracy Evolution.** Figure 3 shows accuracy trajectories on CIFAR-100 (10 tasks), ImageNet-A (50 tasks), and ImageNet-R (50 tasks). PaLoRA maintains strong performance through the task sequence, with clear gains in the two long-horizon settings. Full trajectories across datasets and task splits are provided in Appendix B.2.

**Backward Transfer.** Table 4 consolidates BWT on all three datasets for 25 and 50 tasks. PaLoRA achieves the best BWT on CIFAR-100 and ImageNet-A; at 50 tasks, it improves over LoRA$^-$DRS from $-5.26$ to $-4.72$ and from $-9.01$ to $-7.09$, respectively. On ImageNet-R, its BWT is slightly worse ($-5.34/-5.20$ versus $-4.78/-4.92$ for 25/50 tasks), while its final accuracy is higher (Table 1).

### 4.3 Comparison with SD-LoRA

SD-LoRA [52] decouples the magnitude and direction of LoRA updates and learns task-specific magnitude scalars. PaLoRA instead determines pacing from the accumulated effective rank, without

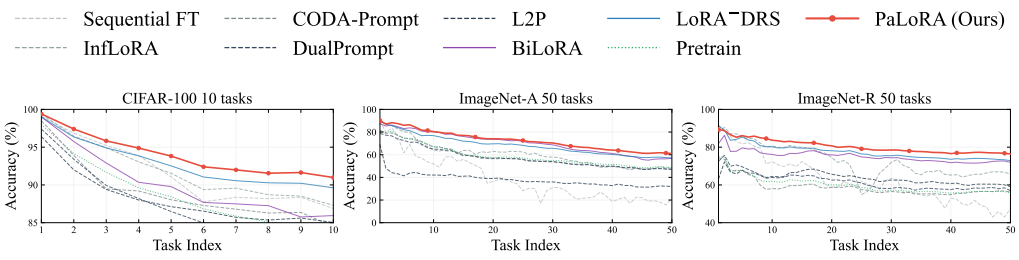



Figure 3: Accuracy trajectories on CIFAR-100 (10 tasks), ImageNet-A (50 tasks), and ImageNet-R (50 tasks), from left to right. PaLoRA maintains strong accuracy, including over long task sequences.

Table 5: Direct comparison with SD-LoRA (accuracy in %; point estimates). PaLoRA achieves larger final-accuracy gains at 50 tasks than at 10 tasks on all three datasets.

| Dataset | Method | 10 Tasks | | 20 Tasks | | 25 Tasks | | 50 Tasks | |
|---|---|---|---|---|---|---|---|---|---|
| | | $ACC_{10}$ | $\overline{ACC_{10}}$ | $ACC_{20}$ | $\overline{ACC_{20}}$ | $ACC_{25}$ | $\overline{ACC_{25}}$ | $ACC_{50}$ | $\overline{ACC_{50}}$ |
| ImageNet-R | SD-LoRA | 78.33 | 82.85 | 74.28 | 79.45 | 75.37 | 80.82 | 69.68 | 76.24 |
| | **PaLoRA** | **78.82** | **82.96** | **78.57** | **82.74** | **78.10** | **82.90** | **76.40** | **80.40** |
| CIFAR-100 | SD-LoRA | 88.05 | 92.47 | 84.20 | 90.48 | 80.32 | 87.54 | 67.70 | 77.60 |
| | **PaLoRA** | **90.98** | **93.99** | **90.31** | **93.67** | **90.25** | **93.69** | **88.86** | **92.53** |
| ImageNet-A | SD-LoRA | 57.21 | 70.68 | 50.63 | 65.06 | 46.94 | 58.19 | 31.99 | 54.78 |
| | **PaLoRA** | **61.95** | **73.02** | **61.82** | **72.70** | **62.54** | **73.21** | **61.09** | **72.52** |

learning a magnitude scalar for each task. Table 5 reports their direct comparison. PaLoRA performs better in every reported setting, with larger final-accuracy gaps at 50 tasks than at 10 tasks. At 50 tasks, the gains are 6.72, 21.16, and 29.10 percentage points on ImageNet-R, CIFAR-100, and ImageNet-A, respectively.

### 4.4 Ablation Study

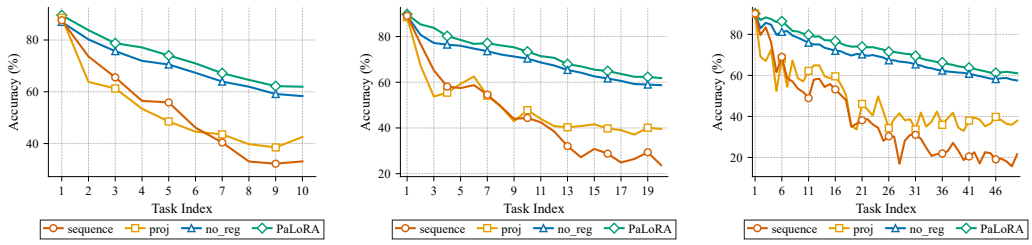
**Module Ablation.** To better understand the contribution of each component, we conduct ablation experiments on ImageNet-A, where **Sequence** performs the worst. We decompose our method into three parts: the regularization term, the nullspace gradient projection, and the core pacing mechanism (which includes SVD-based compression of previous knowledge and gradient scaling based on the resulting effective rank $R$).

We report results under three task settings, as shown in Figure 4. Specifically, **Sequence** includes none of the components, **proj** uses only nullspace gradient projection, **no_reg** combines projection with the pacing mechanism, and PaLoRA includes all three components.

Projection alone mitigates forgetting, while adding rank-aware pacing further improves the accuracy trajectories. The nearly parallel curves with and without regularization suggest an additional plasticity benefit.

Table 6 quantifies the effect of triplet regularization on CIFAR-100. At 50 tasks, PaLoRA without the triplet loss reaches $88.23\%$ final accuracy versus $81.72\%$ for Sequence, and the triplet loss adds a further 0.63 points within PaLoRA. In contrast, adding triplet loss to Sequence reduces final accuracy to $80.00\%$. Together with Figure 4, these results support the pacing framework as the main source of improvement and the triplet loss as an auxiliary plasticity compensator.

**Validation of the Pacing Factor Exponent.** We vary the gradient multiplier as $\kappa/R^{\alpha}$, with fixed $\kappa$ and exponent $\alpha$; its corresponding denominator is $s = R^{\alpha}/\kappa$. Thus, Eqn. (3) predicts $\alpha = 0.5$ (with $\kappa = \sqrt{c}$ for the theoretical normalization). We evaluate $\alpha \in \{0.125, 0.25, 0.5, 1.0, 2.0\}$. The results are shown in Figure 5. We report three metrics: **Accuracy**, which reflects overall performance; **Training Loss** measured at the end of each task, which indicates model plasticity; and **Forgetting**, defined as the average performance drop on all learned tasks after each new task, which measures model stability.

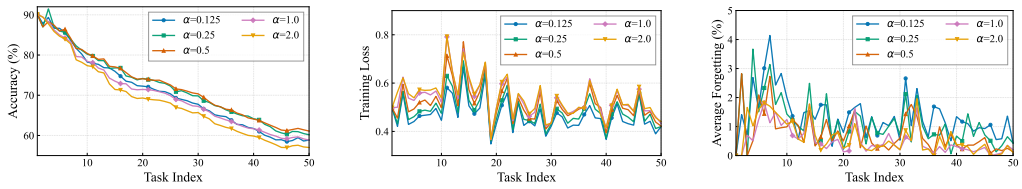



a) 10 Tasks b) 20 Tasks c) 50 Tasks

Figure 4: Component ablation on ImageNet-A with 10, 20, and 50 tasks. Projection and pacing improve the trajectories, while triplet regularization provides an additional accuracy gain. Each setting uses a different class split, so early-task curves are not expected to coincide across panels.

Table 6: Triplet-loss ablation on CIFAR-100 (accuracy in %; point estimates). PaLoRA retains most of its improvement without the triplet loss, which provides a smaller additional gain.

| Method | $ACC_{10}$ | $\overline{ACC_{10}}$ | $ACC_{50}$ | $\overline{ACC_{50}}$ |
|---|---|---|---|---|
| Sequence | 86.88 | 91.57 | 81.72 | 87.99 |
| Sequence + triplet loss | 87.93 | 92.34 | 80.00 | 89.25 |
| PaLoRA w/o triplet loss | 89.59 | 93.18 | 88.23 | 92.40 |
| **PaLoRA** | 90.98 | 93.99 | 88.86 | 92.53 |




a) Accuracy b) Training Loss c) Forgetting

Figure 5: Pacing-exponent ablation on ImageNet-A with 50 tasks. The exponent $\alpha = 0.5$ gives the best accuracy while balancing training loss and forgetting, consistent with the predicted square-root denominator scaling.

The results show that $\alpha = 0.5$ achieves the best performance. When $\alpha > 0.5$, the model suffers from reduced plasticity, leading to performance degradation. In contrast, when $\alpha < 0.5$, the model becomes less stable, resulting in increased forgetting and lower overall performance.

Additional effective-rank comparisons are provided in Appendix B.3; sensitivity, computational overhead, LoRA-variant, and backbone results are reported in Appendix B.

## 5 Limitations and Future Work

Our pacing law is derived in a single-layer linear setting under anisotropic leakage and local quadratic sensitivity assumptions. Its application to deep ViTs provides an approximate scaling principle rather than a formal guarantee; the empirical results do not establish that these assumptions hold in the full nonlinear network. The Pareto-optimality result concerns the budget-constrained pacing mechanism, while the auxiliary triplet loss is an empirically motivated plasticity compensator.

Our evaluation is limited to image classification. Extending PaLoRA to generative models, including LLMs and MLLMs, remains future work. The update-pacing principle may also apply beyond LoRA, but suitable measures of accumulated knowledge require further investigation. Although the sensitivity experiments support the chosen pacing constant and SVD threshold, their behavior in broader continual learning settings remains to be studied.

## 6 Conclusion

We show that LoRA-based continual learning suffers from anisotropic leakage into previously acquired subspaces, rendering fixed gradient scaling heuristics inherently suboptimal. Building on this insight, we derive the rank-aware pacing law $s^* = \sqrt{R/c}$ and theoretically establish its Pareto optimality under the stability–plasticity trade-off. Motivated by this principle, we propose PaLoRA, which combines adaptive SVD compression with rank-dependent gradient pacing. Extensive experiments demonstrate that PaLoRA consistently improves performance across benchmarks, achieving up to 4% gains on the challenging 50-task ImageNet-A and ImageNet-R continual learning settings.

## Acknowledgements

This work is supported by the Fundamental Research Funds for the Central Universities, Peking University.

## References


[1] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *Advances in neural information processing systems*, 32, 2019.

[2] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 233–248, 2018.

[3] Aristotelis Chrysakis and Marie-Francine Moens. Online continual learning from imbalanced data. In *International Conference on Machine Learning*, pages 1952–1961. PMLR, 2020.

[4] Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. Learning without memorizing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5138–5146, 2019.

[5] Peiyan Dong, Lei Lu, Chao Wu, Cheng Lyu, Geng Yuan, Hao Tang, and Yanzhi Wang. Packqvit: Faster sub-8-bit vision transformers via full and packed quantization on the mobile. In *NeurIPS*, 2023.

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[7] Ziqi Gao, Qichao Wang, Aochuan Chen, Zijing Liu, Bingzhe Wu, Liang Chen, and Jia Li. Parameter-efficient fine-tuning with discrete fourier transform. *arXiv preprint arXiv:2405.03003*, 2024.

[8] Haiyang Guo, Yichen Shi, Fei Zhu, Wenzhuo Liu, Hongbo Zhao, Fanhu Zeng, Shijie Ma, Da-Han Wang, and Xu-Yao Zhang. Cl-vista: Benchmarking continual learning in video large language models. *arXiv preprint arXiv:2604.00677*, 2026.

[9] Haiyang Guo, Fanhu Zeng, Ziwei Xiang, Fei Zhu, Da-Han Wang, Xu-Yao Zhang, and Cheng-Lin Liu. Hide-llava: Hierarchical decoupling for continual instruction tuning of multimodal large language model. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13572–13586, 2025.

[10] Haiyang Guo, Fanhu Zeng, Fei Zhu, Wenzhuo Liu, Da-Han Wang, Jian Xu, Xu-Yao Zhang, and Cheng-Lin Liu. Federated continual instruction tuning. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1325–1335. IEEE, 2025.

[11] Haiyang Guo, Fanhu Zeng, Fei Zhu, Jiayi Wang, Xukai Wang, Jingang Zhou, Hongbo Zhao, Wenzhuo Liu, Shijie Ma, Da-Han Wang, et al. Continual learning for generative ai: From llms to mllms and beyond. *arXiv preprint arXiv:2506.13045*, 2025.

[12] Haiyang Guo, Fei Zhu, Hongbo Zhao, Fanhu Zeng, Wenzhuo Liu, Shijie Ma, Da-Han Wang, and Xu-Yao Zhang. Mcitlib: Multimodal continual instruction tuning library and benchmark. *arXiv preprint arXiv:2508.07307*, 2025.

[13] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. A survey on vision transformer. *IEEE transactions on pattern analysis and machine intelligence*, 45(1):87–110, 2022.

[14] Soufiane Hayou, Nikhil Ghosh, and Bin Yu. Lora+: Efficient low rank adaptation of large models. *arXiv preprint arXiv:2402.12354*, 2024.

[15] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8340–8349, 2021.

[16] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15262–15271, 2021.

[17] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[18] Linlan Huang, Xusheng Cao, Haori Lu, Yifan Meng, Fei Yang, and Xialei Liu. Mind the gap: Preserving and compensating for the modality gap in clip-based continual learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3777–3786, 2025.

[19] Yukang Huo and Hao Tang. When continue learning meets multimodal large language model: A survey. *CoRR*, abs/2503.01887, 2025.

[20] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European conference on computer vision*, pages 709–727. Springer, 2022.

[21] Shibo Jie, Zhi-Hong Deng, Shixuan Chen, and Zhijuan Jin. Convolutional bypasses are better vision transformer adapters. In *ECAI 2024: 27th European Conference on Artificial Intelligence, 19–24 October 2024, Santiago de Compostela, Spain–Including 13th Conference on Prestigious Applications of Intelligent Systems (PAIS 2024)*, pages 202–209. SAGE Publications Pvt. Ltd 1 Oliver's Yard, 55 City Road, London, EC1Y 1SP, 2024.

[22] Damjan Kalajdzievski. A rank stabilization scaling factor for fine-tuning with lora. *arXiv preprint arXiv:2312.03732*, 2023.

[23] Ronald Kemker, Marc McClure, Angelina Abitino, Tyler Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[24] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *ACM computing surveys (CSUR)*, 54(10s):1–41, 2022.

[25] Zhenglun Kong, Dongkuan Xu, Zhengang Li, Peiyan Dong, Hao Tang, Yanzhi Wang, and Subhabrata Mukherjee. Autovit: Achieving real-time vision transformers on mobile via latency-aware coarse-to-fine search. *Int. J. Comput. Vis.*, 133(9):6170–6186, 2025.

[26] Dawid J Kopiczko, Tijmen Blankevoort, and Yuki M Asano. Vera: Vector-based random matrix adaptation. *arXiv preprint arXiv:2310.11454*, 2023.

[27] A Krizhevsky. Learning multiple layers of features from tiny images. *Master's thesis, University of Tront*, 2009.

[28] Jaejun Lee, Raphael Tang, and Jimmy Lin. What would elsa do? freezing layers during transformer fine-tuning. *arXiv preprint arXiv:1911.03090*, 2019.

[29] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, 2021.

[30] Zhengang Li, Alec Lu, Yanyue Xie, Zhenglun Kong, Mengshu Sun, Hao Tang, Zhong Jia Xue, Peiyan Dong, Caiwen Ding, Yanzhi Wang, Xue Lin, and Zhenman Fang. Quasar-vit: Hardware-oriented quantization-aware architecture search for vision transformers. In *ICS*, pages 324–337. ACM, 2024.

[31] Yan-Shuo Liang and Wu-Jun Li. Inflora: Interference-free low-rank adaptation for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23638–23647, 2024.

[32] Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. In *Forty-first International Conference on Machine Learning*, 2024.

[33] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. *AI open*, 5:208–215, 2024.

[34] Xuan Liu and Xiaobin Chang. Lora subtraction for drift-resistant space in exemplar-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15308–15318, 2025.

[35] Marc Masana, Xialei Liu, Bartłomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost Van De Weijer. Class-incremental learning: survey and performance evaluation on image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5):5513–5533, 2022.

[36] Fanxu Meng, Zhaohui Wang, and Muhan Zhang. Pissa: Principal singular values and singular vectors adaptation of large language models. In *NeurIPS*, 2024.

[37] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71, 2019.

[38] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. Adapterfusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th conference of the European chapter of the association for computational linguistics: main volume*, pages 487–503, 2021.

[39] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[40] Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2):123–146, 1995.

[41] Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. *arXiv preprint arXiv:2103.09762*, 2021.

[42] Yihua Shao, Jia Li, Siyu Chen, Xinyu Luo, Yang Liu, Kecheng Chen, Xinwei Long, Lingyu Zhu, Fanhu Zeng, Maolin Wang, Ziyang Yan, Jingcai Guo, Hao Tang, Nicu Sebe, and Zhenyi Wang. List: Local-simplex test-time lora fusion. *CoRR*, abs/2608.22370, 2026.

[43] Yihua Shao, Xiaofeng Lin, Xinwei Long, Siyu Chen, Minxi Yan, Yang Liu, Ziyang Yan, Ao Ma, Hao Tang, and Jingcai Guo. Icm-fusion: In-context meta-optimized lora fusion for multi-task adaptation. In *AAAI*, pages 8860–8868. AAAI Press, 2026.

[44] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11909–11919, 2023.

[45] Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *arXiv preprint arXiv:2106.10270*, 2021.

[46] Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. Lst: Ladder side-tuning for parameter and memory efficient transfer learning. *Advances in Neural Information Processing Systems*, 35:12991–13005, 2022.

[47] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.

[48] Gido M Van de Ven, Tinne Tuytelaars, and Andreas S Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 4(12):1185–1197, 2022.

[49] Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuan-Jing Huang. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10658–10671, 2023.

[50] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European conference on computer vision*, pages 631–648. Springer, 2022.

[51] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 139–149, 2022.

[52] Yichen Wu, Hongming Piao, Long-Kai Huang, Renzhen Wang, Wanhua Li, Hanspeter Pfister, Deyu Meng, Kede Ma, and Ying Wei. Sd-lora: Scalable decoupled low-rank adaptation for class incremental learning. In *ICLR*. OpenReview.net, 2025.

[53] Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanqi Tan, Baobao Chang, Songfang Huang, and Fei Huang. Raise a child in large language model: Towards effective and generalizable fine-tuning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pages 9514–9528, 2021.

[54] Hong Xuan, Abby Stylianou, and Robert Pless. Improved embeddings with easy positive triplet mining. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2474–2482, 2020.

[55] Fanhu Zeng, Zhen Cheng, Fei Zhu, Hongxin Wei, and Xu-Yao Zhang. Local-prompt: Extensible local prompts for few-shot out-of-distribution detection. In *The Thirteenth International Conference on Learning Representations*, 2025.

[56] Fanhu Zeng, Haiyang Guo, Fei Zhu, Li Shen, and Hao Tang. Robustmerge: Parameter-efficient model merging for mllms with direction robustness. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[57] Fanhu Zeng, Fei Zhu, Haiyang Guo, Xu-Yao Zhang, and Cheng-Lin Liu. Modalprompt: Towards efficient multimodal continual instruction tuning with dual-modality guided prompt. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 12137–12152, 2025.

[58] Gengwei Zhang, Liyuan Wang, Guoliang Kang, Ling Chen, and Yunchao Wei. Slca: Slow learner with classifier alignment for continual learning on a pre-trained model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19148–19158, 2023.

[59] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*, 2023.

[60] Da-Wei Zhou, Hai-Long Sun, Han-Jia Ye, and De-Chuan Zhan. Expandable subspace ensemble for pre-trained model-based class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23554–23564, 2024.

[61] Hao Zhu, Yifei Zhang, Junhao Dong, and Piotr Koniusz. Bilora: almost-orthogonal parameter spaces for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 25613–25622, 2025.

# A Full Theoretical Details of Rank-Aware Scaling

## A.1 Setup and Notation

Consider a single linear layer with frozen pre-trained weights $W_0 \in \mathbb{R}^{d\times d}$. In a continual learning setting, after learning tasks $1, \ldots, t-1$ via LoRA, the accumulated low-rank adaptations form an effective update $C_{t-1} \in \mathbb{R}^{d\times d}$ with $\mathrm{rank}(C_{t-1}) = R$, so that the current effective weight is:

$$W^{(t-1)} = W_0 + C_{t-1}.$$

Task $t$ introduces an additional LoRA increment $\Delta_t \in \mathbb{R}^{d\times d}$, and the updated weight becomes:

$$W^{(t)} = W^{(t-1)} + \Delta_t = W_0 + C_{t-1} + \Delta_t.$$

Each task $t$ is equipped with an input design matrix $X_t \in \mathbb{R}^{d\times n_t}$ with positive-definite covariance $\Sigma_t = X_t X_t^\top \succ 0$. The task-$t$ objective is the Frobenius-norm regression loss:

$$\mathcal{L}_t(\Delta_t) = \frac{1}{2}\|(W^{(t-1)} + \Delta_t)X_t - Y_t\|_F^2,$$

optimized via full-batch gradient descent with fixed-step projected gradient descent.

After completing task $t-1$, compute the compact SVD of the accumulated LoRA adaptation:

$$C_{t-1} = U\Sigma V^\top = \sum_{i=1}^{R} \sigma_i u_i v_i^\top, \qquad R = \mathrm{rank}(C_{t-1}). \tag{5}$$

Let $V_\perp = [v_{R+1}, \ldots, v_d]$ span the right nullspace (orthogonal complement of the row space) of $C_{t-1}$. We constrain the task-$t$ LoRA increment to the feasible subspace:

$$\mathcal{N}_t \triangleq \{MV_\perp^\top : M \in \mathbb{R}^{d\times(d-R)}\}.$$

Equivalently, $\Delta_t V_{[:R]} = 0$, which implies $\Delta_t x_{\mathrm{old}} = 0$ for all old-task inputs $x_{\mathrm{old}} \in \mathrm{span}(V_{[:R]})$. Under exact arithmetic, old-task forward passes are therefore preserved.

Let $P_{\mathcal{N}_t}$ denote the Frobenius-orthogonal projection onto $\mathcal{N}_t$, which acts as $P_{\mathcal{N}_t}(G) = GV_\perp V_\perp^\top$ for any $G \in \mathbb{R}^{d\times d}$. The ideal task-$t$ update is:

$$\delta\Delta_t = -\frac{\eta}{s} P_{\mathcal{N}_t}(\nabla_{\Delta_t}\mathcal{L}_t),$$

where $\eta$ is the base learning rate, $s > 0$ is the step-size denominator, and $\nabla_{\Delta_t}\mathcal{L}_t = ((W^{(t-1)} + \Delta_t)X_t - Y_t)X_t^\top$ is evaluated at the current iterate.

## A.2 Leakage and Sensitivity Assumptions

In practice, finite-precision arithmetic and the imperfect implementation of the projector cause the effective update to leak into the old-task subspace $\mathcal{S} = \mathrm{span}\{u_i v_i^\top\}_{i=1}^R$.

**Assumption A.1** (Anisotropic Leakage)**.** After a scaled projected gradient step, decompose the residual component of the effective weight update in the SVD basis as:

$$P_{\mathcal{S}}(\delta\Delta_t) = \sum_{i=1}^{R} \epsilon_i\, u_i v_i^\top.$$

The per-direction leakage satisfies:

$$\mathbb{E}[\epsilon_i^2] = \frac{\eta^2\gamma_i^2}{s^2}, \qquad \forall i \in \{1, \ldots, R\}, \tag{6}$$

where $\gamma_i > 0$ captures the direction-dependent base leakage variance. The leakage is independent across directions.

**Assumption A.2** (Direction-Dependent Quadratic Sensitivity)**.** Near the optimum $W^{(t-1)}$, the old-task loss is locally quadratic with curvature $\lambda_i > 0$ along the $i$-th singular direction:

$$\mathcal{L}_{t-1}(W^{(t-1)} + \delta W) - \mathcal{L}_{t-1}(W^{(t-1)}) \approx \frac{1}{2}\sum_{i=1}^{R} \lambda_i \langle u_i v_i^\top, \delta W\rangle_F^2.$$

These assumptions characterize residual leakage along historical directions, rather than the full displacement between task optima. For deep ViTs with LoRA, small projected leakage motivates a local quadratic approximation, but its validity is not established by this analysis.

### A.3 The Forgetting–Progress Trade-off

Under Assumptions A.1–A.2, the per-step expected forgetting and first-order new-task progress are given by Eqns. (1) and (2).

Eqn. (1) shows that scaling down the step by $s$ reduces forgetting *quadratically*: halving the step cuts forgetting by a factor of four. Meanwhile, Eqn. (2) shows that new-task progress degrades only *linearly*: halving the step merely halves the progress. This asymmetric sensitivity is the fundamental mechanism behind the $\sqrt{R/c}$ scaling law: we can pay a small linear price in learning speed for a large quadratic reduction in catastrophic forgetting.

Proofs of Lemmas A.3 and A.4 are given below in Section A.4.

### A.4 Proofs of Main Lemmas

**Lemma A.3** (Expected Forgetting per Step)**.** *Under Assumptions A.1–A.2, the expected old-task loss increase from one scaled step is:*

$$\mathbb{E}[\Delta\mathcal{L}_{t-1}] = \frac{\eta^2 R\Gamma^2}{2s^2}, \qquad \textit{where} \quad \Gamma^2 \triangleq \frac{1}{R}\sum_{i=1}^{R}\lambda_i\gamma_i^2. \tag{7}$$

*Proof.* Assumption A.2 together with the orthonormality $\langle u_i v_i^\top, u_j v_j^\top\rangle_F = \delta_{ij}$ yields

$$\Delta\mathcal{L}_{t-1} \approx \frac{1}{2}\sum_{i=1}^{R}\lambda_i\epsilon_i^2.$$

Taking expectation and applying Assumption A.1, we obtain:

$$\mathbb{E}[\Delta\mathcal{L}_{t-1}] = \frac{1}{2}\sum_{i=1}^{R}\lambda_i\mathbb{E}[\epsilon_i^2] = \frac{1}{2}\sum_{i=1}^{R}\lambda_i\frac{\eta^2\gamma_i^2}{s^2} = \frac{\eta^2}{2s^2}\sum_{i=1}^{R}\lambda_i\gamma_i^2 = \frac{\eta^2 R\Gamma^2}{2s^2}. \qquad \square$$

**Lemma A.4** (New-Task Progress per Step)**.** *For the projected gradient $g = P_{\mathcal{N}_t}(\nabla_{\Delta_t}\mathcal{L}_t)$ with squared norm $G^2 = \|g\|_F^2$, a step $\delta\Delta_t = -\frac{\eta}{s}g$ yields first-order progress:*

$$\Delta\mathcal{L}_t \approx -\frac{\eta G^2}{s}. \tag{8}$$

*Hence the magnitude of new-task progress is $\eta G^2/s$.*

*Proof.* By first-order Taylor expansion of $\mathcal{L}_t$ around the current iterate:

$$\Delta\mathcal{L}_t \approx \langle\nabla_{\Delta_t}\mathcal{L}_t, \delta\Delta_t\rangle_F = \left\langle g, -\frac{\eta}{s}g\right\rangle_F = -\frac{\eta}{s}\|g\|_F^2 = -\frac{\eta G^2}{s}. \qquad \square$$

**Lemma A.5** (Optimal Scaling Factor for a Given Budget)**.** *Consider the constrained optimization problem that maximizes new-task progress subject to a per-step forgetting budget $\epsilon_0 > 0$:*

$$\max_{s>0}\ \frac{\eta G^2}{s} \quad \textit{subject to} \quad \frac{\eta^2 R\Gamma^2}{2s^2} \le \epsilon_0.$$

*The unique optimal scale is:*

$$s^* = \eta\Gamma\sqrt{\frac{R}{2\epsilon_0}}. \tag{9}$$

*Proof.* The objective $\eta G^2/s$ is strictly decreasing in $s$. The constraint yields the feasible set $s \ge \eta\Gamma\sqrt{\frac{R}{2\epsilon_0}}$. Because the objective decreases with $s$, the optimum is attained at the left boundary, giving $s^* = \eta\Gamma\sqrt{\frac{R}{2\epsilon_0}}$. $\square$

**Definition A.6** (Average Per-Direction Noise Floor)**.** The average per-direction noise floor $\epsilon_{\text{dir}}$ is the mean expected old-task forgetting from one unscaled step ($s = 1$) leaking into a single singular direction:

$$\epsilon_{\text{dir}} \triangleq \frac{1}{R} \sum_{i=1}^{R} \frac{\lambda_i \eta^2 \gamma_i^2}{2} = \frac{\eta^2 \Gamma^2}{2}. \tag{10}$$

We set the budget to $c$ times this average floor:

$$\epsilon_0 = c\, \epsilon_{\text{dir}} = \frac{c\, \eta^2 \Gamma^2}{2}. \tag{11}$$

**Lemma A.7** (The $\sqrt{R/c}$ Scaling Law)**.** *With the natural budget $\epsilon_0 = \frac{c\,\eta^2\Gamma^2}{2}$, the optimal scale simplifies to:*

$$s^* = \sqrt{\frac{R}{c}}. \tag{12}$$

*Proof.* Substitute $\epsilon_0 = \frac{c\,\eta^2\Gamma^2}{2}$ from Eqn. (11) into Eqn. (9):

$$s^* = \eta\Gamma \sqrt{\frac{R}{2 \cdot \frac{c\,\eta^2\Gamma^2}{2}}} = \eta\Gamma \sqrt{\frac{R}{c\,\eta^2\Gamma^2}} = \sqrt{\frac{\eta^2\Gamma^2 \cdot R}{c\,\eta^2\Gamma^2}} = \sqrt{\frac{R}{c}}. \qquad \square$$

### A.5 Properties of the Scaling Law

Substituting $s^* = \sqrt{R/c}$ into Lemma A.3 gives:

$$\mathbb{E}[\Delta\mathcal{L}_{t-1}] = \frac{\eta^2 R \Gamma^2}{2(\sqrt{R/c})^2} = \frac{\eta^2 R \Gamma^2}{2 \cdot R/c} = \frac{c\,\eta^2\Gamma^2}{2} = \epsilon_0.$$

Thus the per-step forgetting exactly equals the prescribed budget $c\,\epsilon_{\text{dir}}$, regardless of the specific directions that constitute the current rank-$R$ subspace.

Any $s < \sqrt{R/c}$ violates the budget because $\mathbb{E}[\Delta\mathcal{L}_{t-1}] > \epsilon_0$. Any $s > \sqrt{R/c}$ is suboptimal because, by Lemma A.4, new-task progress $\eta G^2/s$ is strictly smaller than achievable. Hence $s^*$ is the unique Pareto-optimal operating point for this budget-constrained pacing problem under the stated model, rather than for the full PaLoRA system with auxiliary regularization.

## B Additional Experimental Results

### B.1 Implementation Details

Our method is implemented in PyTorch. Following prior work [31, 34], we insert LoRA modules into the key and value projections of the attention layers in ViT. Across all datasets and task settings, we fix the SVD compression threshold $\tau$ to 0.99 and fix $C$ to 4 in the pacing factor. The regularization term is weighted by 0.5 with a margin of 2.0.

The main accuracy tables report means and standard deviations over seeds 0–4; the additional analyses report point estimates unless otherwise stated. Experiments are conducted on a single NVIDIA GeForce RTX 4080 SUPER GPU. Unless otherwise specified, we use a ViT-B/16-IN21K pretrained backbone. Training is performed using the Adam optimizer with a learning rate of 0.0005 and $(\beta_1, \beta_2) = (0.9, 0.999)$. We train for 50 epochs on ImageNet-R and ImageNet-A, and 20 epochs on CIFAR-100, with a batch size of 128.

### B.2 Additional Accuracy Curves

The main text shows representative trajectories from all three datasets. Figure 6 provides the full results across datasets and task splits, showing that PaLoRA maintains strong performance throughout the learning sequences.

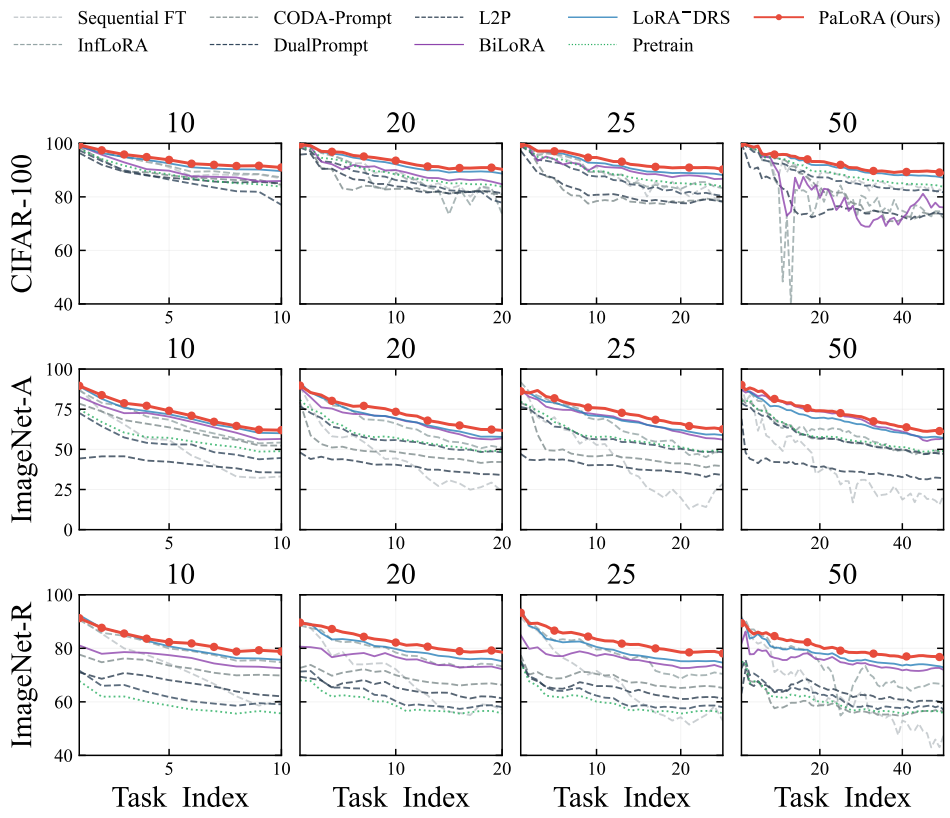
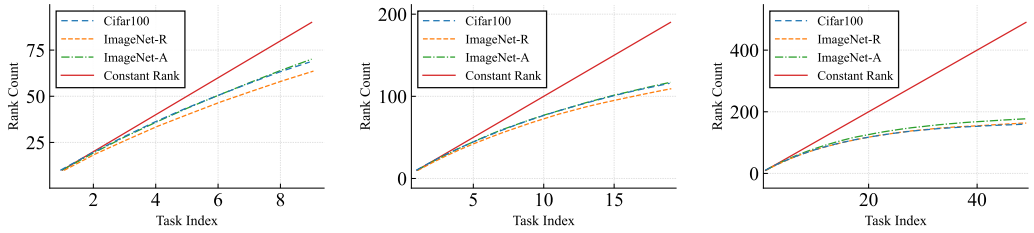



Figure 6: Accuracy trajectories across CIFAR-100, ImageNet-A, and ImageNet-R under 10-, 20-, 25-, and 50-task splits. PaLoRA maintains strong performance across datasets and task horizons.




a) 10 Tasks b) 20 Tasks c) 50 Tasks

Figure 7: SVD compression across datasets with 10, 20, and 50 tasks. At $\tau = 0.99$, the effective rank grows more slowly than the uncompressed $R = rt$ reference and varies with the task sequence.

### B.3 Effectiveness of SVD Compression

We evaluate the effectiveness of the SVD compression module across all three datasets under different task settings. The results are illustrated in Figure 7.

The **Constant Rank** curve denotes a predefined rank budget, where $R = r\times$ (number of tasks). The other three curves represent the actual effective rank of the accumulated knowledge after each task on the three datasets. From the results, it is clear that under a fixed threshold $\tau = 0.99$, our method achieves adaptive rank allocation according to task difficulty. Instead of growing linearly with the number of tasks, the effective rank adjusts dynamically, demonstrating the efficiency of the proposed compression mechanism.

**Different Effective Rank Definitions.** We further analyze the impact of different ways of computing the effective rank $R$. As a comparison, we adopt a fixed-rank SVD strategy that retains the top $t \times k$ components, where $t$ is the number of learned tasks and $k$ is a predefined constant. When $k = 1$,

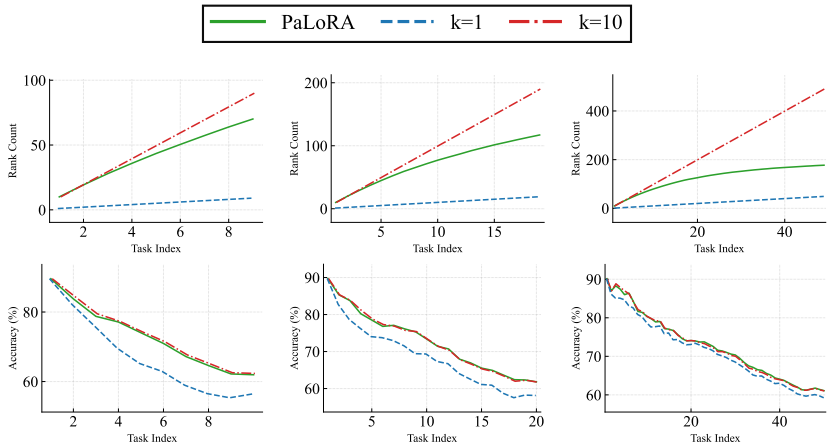



Figure 8: Effective-rank choices on ImageNet-A with 10, 20, and 50 tasks. Top: rank growth; bottom: accuracy. Adaptive SVD retains fewer directions than the $k = r$ reference while maintaining comparable accuracy.

Table 7: Pacing-constant sensitivity on CIFAR-100 with 50 tasks and $\tau = 0.99$ (accuracy in %; point estimates). Performance varies little over the tested range.

| $C$ | $ACC_{50}$ | $\overline{ACC_{50}}$ |
|---|---|---|
| 2.25 | 88.92 | 92.54 |
| 4.00 (default) | 88.86 | 92.53 |
| 6.25 | 88.70 | 92.40 |

this setting corresponds to a lower bound, while $k = r$ represents an upper bound (i.e., no compression). As shown in Figure 8, our method demonstrates strong adaptivity by achieving accuracy curves that are consistently close to the upper bound, while still performing compression according to task difficulty. These results support using adaptive effective rank to balance compression and accuracy.

### B.4 Hyperparameter Sensitivity

We vary the pacing constant $C$ and SVD threshold $\tau$ on CIFAR-100 with 50 tasks. With $\tau = 0.99$, changing $C$ from 2.25 to 6.25 yields final accuracy between $88.70\%$ and $88.92\%$ (Table 7). With $C = 4$, lowering $\tau$ to 0.95 reduces the average final rank to 39.6 but lowers final accuracy to $84.16\%$. Increasing $\tau$ from 0.99 to 0.999 raises the rank from 159.8 to 390.8 with little change in accuracy (Table 8). These results support the default $C = 4$ and $\tau = 0.99$ as a balance between compression and performance.

Table 8: SVD-threshold sensitivity on CIFAR-100 with 50 tasks and $C = 4$ (accuracy in %; point estimates). The default threshold balances retained rank and accuracy.

| $\tau$ | $ACC_{50}$ | $\overline{ACC_{50}}$ | Avg. Final Rank |
|---|---|---|---|
| 0.95 | 84.16 | 90.56 | 39.6 |
| 0.99 (default) | 88.86 | 92.53 | 159.8 |
| 0.999 | 88.74 | 92.55 | 390.8 |

### B.5 Computational Overhead

SVD compression is performed once per task boundary and takes approximately 0.08 seconds per task on an NVIDIA GeForce RTX 4080 SUPER, totaling about 4 seconds over 50 tasks. The triplet loss computes pairwise feature distances with complexity $O(B^2 d)$, where $B = 128$ is the batch size

Table 9: Average training time per task on CIFAR-100 with 50 tasks using an RTX 4080 SUPER. PaLoRA adds 0.34 minutes over Sequence and is faster than LoRA$^-$DRS.

| Method | Time per Task (min) |
|---|---|
| L2P | 2.98 |
| DualPrompt | 2.95 |
| CODA-Prompt | 3.02 |
| BiLoRA | 3.45 |
| Sequence | 2.67 |
| Sequence + triplet loss | 2.80 |
| InfLoRA | 3.31 |
| LoRA$^-$DRS | 4.36 |
| PaLoRA w/o triplet loss | 2.81 |
| PaLoRA | 3.01 |

and $d$ is the feature dimension. Table 9 reports average training time per task on CIFAR-100 with 50 tasks. PaLoRA takes 3.01 minutes versus 2.67 for Sequence and 4.36 for LoRA$^-$DRS. Without triplet loss, PaLoRA takes 2.81 minutes, quantifying the cost of compression, projection, and pacing together; the triplet loss adds a further 0.20 minutes within PaLoRA.

### B.6 Compatibility with Other LoRA Variants

The pacing formulation uses the aggregated low-rank task update rather than a specific factor initialization. We evaluate its compatibility with PiSSA [36] on CIFAR-100 with 50 tasks (Table 10). Applying PaLoRA raises the final accuracy of sequential PiSSA adaptation from $71.85\%$ to $89.30\%$, slightly exceeding the $88.86\%$ obtained by PaLoRA with standard LoRA in this comparison.

Table 10: Compatibility with PiSSA on CIFAR-100 with 50 tasks (accuracy in %; point estimates). PaLoRA improves sequential PiSSA adaptation and slightly exceeds its standard-LoRA counterpart.

| Method | $ACC_{50}$ | $\overline{ACC_{50}}$ |
|---|---|---|
| Sequence | 81.72 | 87.99 |
| PaLoRA | 88.86 | 92.53 |
| Sequence + PiSSA | 71.85 | 80.54 |
| PaLoRA + PiSSA | 89.30 | 92.75 |

### B.7 Different Pretrained Backbones

Table 11 compares DINOv1 and iBOT backbones on ImageNet-R and ImageNet-A with 50 tasks. PaLoRA achieves the highest final and average accuracy in all four settings, extending the comparison beyond the default ViT-B/16-IN21K backbone.

Table 11: Final and average accuracy (%; point estimates) with DINOv1 and iBOT backbones on 50-task ImageNet-R and ImageNet-A. PaLoRA outperforms the compared methods for both backbones.

| Method | ImageNet-R (50 tasks) | | | | ImageNet-A (50 tasks) | | | |
|---|---|---|---|---|---|---|---|---|
| | DINOv1 | | iBOT | | DINOv1 | | iBOT | |
| | ACC | $\overline{ACC}$ | ACC | $\overline{ACC}$ | ACC | $\overline{ACC}$ | ACC | $\overline{ACC}$ |
| InfLoRA [31] | 61.52 | 70.10 | 58.52 | 69.32 | 30.55 | 42.48 | 33.31 | 49.40 |
| BiLoRA [61] | 62.85 | 66.73 | 65.85 | 70.42 | 28.57 | 37.82 | 34.69 | 44.50 |
| LoRA$^-$DRS [34] | 61.58 | 67.26 | 65.12 | 70.94 | 30.28 | 40.19 | 34.89 | 45.37 |
| **PaLoRA (Ours)** | **66.78** | **72.80** | **69.85** | **75.66** | **35.94** | **46.51** | **43.25** | **55.18** |

## NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction accurately reflect the paper's contribution and scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations in the main-text section "Limitations and Future Work".

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: We provide the full set of assumptions and a complete proof in the Appendix.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide the code and logs in the supplements and we provide implement details in Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide the code and logs in the supplements.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide implement details in Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The main accuracy tables report means and standard deviations over five random seeds (0–4). Other tables and the existing accuracy curves report point estimates, as stated in the experimental setup.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide implement details (include compute resources) in Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in our paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: The paper has no societal impact.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The creators or original owners of assets (e.g., code, data, models) used in the paper are properly credited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.