DOFR: Diffusion-Model One-Iter Forgetting-Based Replay for Compute-Budgeted Continual Learning

Taro Murayama
DENSO CORPORATION, Japan
taro.murayama.j7z@jp.denso.com

Ichiro Takeuchi
Nagoya University, RIKEN, Japan
takeuchi.ichiro.n6@f.mail.nagoya-u.ac.jp

***Abstract*—Diffusion models achieve outstanding image-generation performance, yet practical deployment requires continual knowledge expansion under distribution shift. Most continual learning studies focus on memory constraints (limited access to old data). In practice, however, the primary bottleneck is often compute: GPU time and the number of allowable training iterations (optimizer steps) are limited. To improve the final generation quality with a limited number of iterations, we propose Diffusion-model One-iter Forgetting-based Replay (DOFR). Our idea comes from empirical observation that the loss increase after a single iteration on a new task (one-iteration forgetting) serves as a low-cost yet informative proxy for long-term forgetting. Based on this insight, DOFR estimates the forgetting risk at each diffusion timestep in a simple closed form, enabling replay budgets to be allocated to classes with higher forgetting risk. Experiments with Stable Diffusion v1.5 show that, under the same compute budget, DOFR improves overall generation quality over simple replay baselines, yielding a better compute–performance trade-off.**

***Index Terms*—Diffusion models, Continual learning, Computational efficiency, Sustainable AI**

I. Introduction

Diffusion models [1], [2] are likelihood-based probabilistic generative models. They learn a score function from data perturbed by a forward diffusion process and generate high-quality samples via the reverse process. They are now widely used for image and video generation, as well as image editing. Large-scale offline training has produced highly capable models that are deployed in practice [3].

In deployment, data arrive sequentially and their distribution changes over time. Fine-tuning only on newly arrived data tends to cause catastrophic forgetting [4]. In contrast, full retraining on all data is often unsustainable due to compute budget and update latency. Full retraining requires large compute budgets (GPU time, electricity, and monetary cost) and increases update latency (turnaround time until the model is refreshed). Continual Learning (CL) is therefore an important learning paradigm, and many strategies, including experience replay and regularization, have been proposed [5].

Most CL studies emphasize memory constraints (limited access to old data). In many deployments, however, compute often constitutes the primary bottleneck: the number of training iterations and GPU time are limited even when old data remain accessible. While compute-budgeted CL has been studied for discriminative models [6], diffusion models—despite their heavy training cost and timestep-dependent structures—still lack clear design principles for this setting.

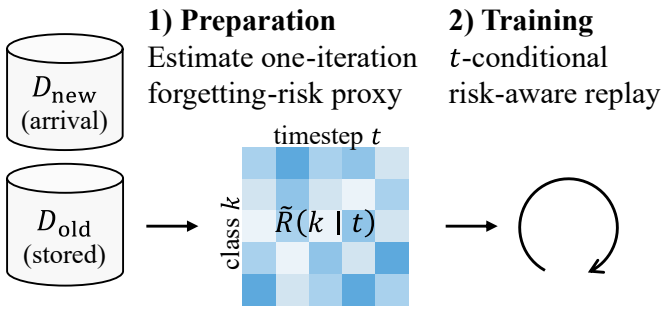

Fig. 1. Conceptual overview of DOFR (Diffusion-model One-iter Forgetting-based Replay). Under a limited training-iteration budget, DOFR improves replay efficiency with no extra diffusion-network forward passes. 1) Preparation phase: upon task arrival, compute a closed-form one-iteration forgetting-risk proxy $\tilde{R}(k \mid t)$ from class-wise summary statistics. 2) Training phase: sample old classes non-uniformly at each timestep, prioritizing classes with higher proxy scores.

To address this gap, we study compute-budgeted CL for diffusion models. We adopt a class-incremental setting as a representative scenario. With a limited number of iterations, the key question is how to allocate replay budget across old classes at each timestep. The ideal allocation would minimize long-term forgetting to preserve the final generation quality, but forecasting it before training is computationally hard. Our basic idea is to focus on the loss increase after a single iteration on the new task (one-iteration forgetting), which provides a cheap early signal of how strongly a new-task update interferes with old-task representations and thus correlates with long-term forgetting. This correlation suggests that one-iteration forgetting is a low-cost proxy for future forgetting risk.

Based on this insight, we propose Diffusion-model One-iter Forgetting-based Replay (DOFR). DOFR estimates the timestep-wise forgetting risk in closed form from class-wise summary statistics (means and per-dimension variances) and uses it to perform non-uniform replay with no extra diffusion-network forward passes (Fig. 1).

We evaluated DOFR using Stable Diffusion v1.5 [3]. We conducted class-incremental task sequences on ImageNet-100 [7] and Food-101-Subset (based on Food-101 [8]). The results showed that, under the same compute budget, DOFR improves overall generation quality compared to simple replay

baselines. These results indicate a better compute–performance trade-off under compute constraints.

Unlike prior compute-budgeted CL work, which mostly targets discriminative models, DOFR leverages the timestep structure of diffusion models to perform class-wise allocation of the replay budget. Our contributions are as follows:

- We show empirically that one-iteration forgetting correlates with long-term forgetting and can serve as a low-cost proxy.
- We propose DOFR, which computes the forgetting-risk proxy in closed form from class statistics and performs risk-aware non-uniform replay.
- Using Stable Diffusion v1.5, we demonstrate improved generation quality under the same compute budget compared to simple replay baselines.

II. Background & Related Work

This section reviews diffusion models and introduces the notation used in Sec. III. It then summarizes related work and positions our contribution.

A. Diffusion Models

We briefly review Variance-Preserving (VP) diffusion models and introduce the notation used in Sec. III. We focus on the discrete-time formulation of DDPM [1], which corresponds to the discretization of the VP SDE [9]. This formulation is widely adopted in modern diffusion models, including Stable Diffusion [3]. Our notation is independent of the specific reverse-process sampler (e.g., DDPM or DDIM) and the output parameterization.

1) Forward Process: Let $x_0 \in \mathbb{R}^d$ denote a clean sample drawn from a distribution $p(x_0)$. We consider a discrete diffusion timestep $t \in \{1, \dots, T\}$, which indexes the noise level in the forward diffusion process. The forward process is defined by a predefined noise schedule $\{(\bar{\alpha}_t, \bar{\sigma}_t)\}_{t=1}^T$, where $\bar{\alpha}_t \in (0,1)$ controls the signal strength and $\bar{\sigma}_t \in (0,1)$ controls the noise variance at timestep t. In the VP setting, these coefficients satisfy $\bar{\alpha}_t + \bar{\sigma}_t = 1$. Given this schedule, the forward diffusion process gradually perturbs the sample by adding Gaussian noise. Specifically, the conditional distribution of the noisy sample x_t given x_0 is $q(x_t \mid x_0) = \mathcal{N}(\sqrt{\bar{\alpha}_t} x_0, \bar{\sigma}_t I)$, which can be reparameterized as:

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{\bar{\sigma}_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0, I). \tag{1}$$

Let $p_t(x_t)$ denote the marginal distribution of x_t obtained by drawing $x_0 \sim p(x_0)$ and applying the forward process. For a class c with class-conditional data distribution $p_c(x_0)$, we define the corresponding noisy distribution at timestep t as $p_{c,t}(x_t) \coloneqq \int q(x_t \mid x_0) p_c(x_0)\, dx_0$.

2) Learning Objective and Score-based Representation: Diffusion models are commonly trained to predict the noise added in the forward process. Let $\epsilon_\theta(x_t, t, c)$ denote a neural network parameterized by θ, where c is the class condition. The noise-prediction loss for each sample is defined as

$$\ell_{x_0,t}(\theta) \coloneqq \mathbb{E}_{\epsilon \sim \mathcal{N}(0,I)}\left[\|\epsilon - \epsilon_\theta(x_t, t, c)\|_2^2\right]. \tag{2}$$

For our analysis in Sec. III, we adopt a score-based view to keep the notation agnostic to the model output parameterization. For each class c and timestep t, we define the (marginal) score as $s_{c,t}(x_t) \coloneqq \nabla_{x_t} \log p_{c,t}(x_t)$. In the forward process (Eq. (1)), the conditional score satisfies $\nabla_{x_t} \log q(x_t \mid x_0) = -(x_t - \sqrt{\bar{\alpha}_t} x_0)/\bar{\sigma}_t = -\epsilon/\sqrt{\bar{\sigma}_t}$ [2], [10]. This motivates the score parameterization $s_\theta(x_t, t, c) \coloneqq -\frac{1}{\sqrt{\bar{\sigma}_t}}\epsilon_\theta(x_t, t, c)$, under which the standard noise-prediction loss can be interpreted as a weighted denoising score matching objective [1]. Using this representation, we define the class-and-timestep loss as

$$\begin{aligned} L_{c,t}(\theta) &\coloneqq \mathbb{E}_{x_0 \sim p_c}\left[\ell_{x_0,t}(\theta)\right] \\ &= \bar{\sigma}_t\, \mathbb{E}_{x_t \sim p_{c,t}}\left[\|s_{c,t}(x_t) - s_\theta(x_t, t, c)\|_2^2\right] + \text{const.} \end{aligned} \tag{3}$$

B. Related Work

This subsection reviews three related areas: (i) compute-budgeted CL in discriminative models, (ii) CL in diffusion models, and (iii) efficient diffusion training. Taken together, these bodies of work highlight that diffusion-model CL lacks a perspective on principled compute-budget allocation.

1) Compute-Budgeted Continual Learning: In standard CL, memory constraints limit how much old data can be stored. A growing body of work focuses on compute-budgeted settings, where the number of training iterations or available GPU time is the primary constraint. For example, under a fixed compute budget, simple random replay can be a strong baseline compared to more complex regularization methods [6]. In addition, prior studies have examined conditions under which leveraging pretrained parameters can outperform training from scratch [11], and have proposed adaptive replay schemes that estimate data importance and prioritize useful samples [12]. However, these discussions are mostly limited to discriminative models (classifiers). Generative models, especially recent diffusion models, incur large training costs, but a principled compute-budget allocation strategy that accounts for their timestep dependence remains largely unexplored. This work extends the compute-budgeted perspective to diffusion models and proposes a replay strategy tailored to their structure.

2) Continual Learning in Diffusion Models: Prior work on CL of diffusion models can be broadly categorized into two directions. One direction focuses on continually training diffusion models to expand their knowledge, including evaluations of CL properties [13], distillation-based generative replay [14], benchmark proposals [15], and continual personalization for specific user concepts [16]–[18]. The other direction uses diffusion models to support CL of classifiers via generative replay [19]–[21]. However, most studies target memory constraints or assume sufficient update steps, leaving compute-budgeted CL underexplored.

3) Efficient Training of Diffusion Models: Many studies aim to improve the training efficiency of diffusion models. In particular, prior work proposed redesigning timestep-wise loss weighting [22], [23], non-uniform timestep sampling [24], [25], and improved noise schedules [26]. These methods address the question of which timesteps should be prioritized, and they are closely related to effective use of computational

resources. However, they improve efficiency by prioritizing timesteps within a single task. In contrast, this work introduces the efficiency perspective into CL and mitigates forgetting by prioritizing classes at each timestep. While both approaches share the motivation of efficient resource allocation, they prioritize different axes.

III. Method

In compute-budgeted CL of diffusion models, a central challenge is how to allocate a limited number of iterations across old classes. We address this by designing a timestep-conditional replay distribution over old classes and deriving an efficient proxy score to guide the allocation.

A. Problem Setup

1) Compute-budgeted CL setting: In this work, we consider a *compute-budgeted CL* setting where access to old datasets is not restricted, but the compute budget for additional training (i.e., the number of iterations) is limited.

Let C_{old} and C_{new} be the disjoint sets of old and new classes. For each old class $k \in C_{\text{old}}$, let $p_k(x_0)$ denote its stationary data distribution. For the new task, we define $p_{\text{new}}(x_0)$ as the marginal distribution aggregated over the new classes in C_{new}. We assume access to datasets sampled from these distributions: $D_{\text{old}} = \bigcup_{k \in C_{\text{old}}} D_{\text{old}}^{(k)}$ and D_{new}.

We write the sampling distributions for new-task data and timesteps as $q_{\text{new}}(x_0)$ and $q_{\text{new}}(t)$, and denote by $q_{\text{old}}(x_0, t)$ the replay distribution over old data and timesteps. Then, the replay-based CL objective is

$$\begin{aligned}\mathcal{L}_{\text{CL}}(\theta) = (1-\rho)\,\mathbb{E}_{(x_0,t)\sim q_{\text{old}}(x_0,t)}[\ell_{x_0,t}(\theta)] \\ + \rho\,\mathbb{E}_{x_0\sim q_{\text{new}}(x_0),\, t\sim q_{\text{new}}(t)}[\ell_{x_0,t}(\theta)],\end{aligned} \tag{4}$$

where $\ell_{x_0,t}(\theta)$ is the per-sample loss in Eq. (2) and ρ corresponds to the fraction of new-task data in a mini-batch. We take $q_{\text{new}}(x_0)$ to be the empirical distribution on D_{new} and treat $q_{\text{new}}(t)$ as a fixed, pre-specified distribution. Under this setup, the problem reduces to designing the replay distribution $q_{\text{old}}(x_0, t)$.

2) Replay distribution: We design the replay distribution $q_{\text{old}}(x_0, t)$ under the following two constraints. These constraints leave only the choice of *which old class to replay at each timestep*.

($C1$) **Matched timestep marginal.** To simplify the problem and ease interpretation, we match the timestep marginal on the old side to that on the new side and introduce bias only through class selection: $q_{\text{old}}(t) = q_{\text{new}}(t)$.

($C2$) **Uniform sampling within each class.** For computational efficiency, conditioned on selecting class k, we sample x_0 uniformly from the class dataset $D_{\text{old}}^{(k)}$: $q_{\text{old}}(x_0 \mid t) = \sum_k \mathcal{U}(x_0 \mid k)\, q_{\text{old}}(k \mid t)$, where $\mathcal{U}(x_0 \mid k)$ is the uniform distribution over $D_{\text{old}}^{(k)}$.

Consequently, designing the replay distribution reduces to specifying the timestep-conditional class distribution $q_{\text{old}}(k \mid t)$. We bias $q_{\text{old}}(k \mid t)$ toward classes that are more susceptible to interference at timestep t, using a forgetting-risk proxy $R(k \mid t)$:

$$q_{\text{old}}(k \mid t) \propto R(k \mid t)^{\beta}, \tag{5}$$

where $\beta > 0$ is a hyperparameter for sharpness. We derive this proxy $R(k \mid t)$ based on the one-iteration forgetting analysis in Sec. III-B.

B. One-Iter Forgetting Proxy

Definition 1 (I-iteration forgetting): Let θ_{old} be the parameters after learning the old task, and let $\theta^{(I)}$ be the parameters obtained after I training iterations on the new-task distribution, starting from $\theta^{(0)} = \theta_{\text{old}}$. Then we define the I-iteration forgetting at timestep t for an old class k as

$$\Delta L_{k,t}^{(I)} := L_{k,t}(\theta^{(I)}) - L_{k,t}(\theta_{\text{old}}), \tag{6}$$

where $L_{k,t}(\theta)$ is the class-and-timestep loss defined in Eq. (3).

1) Proxy from One-Iteration Forgetting Analysis: The key result in this section is that one-iteration forgetting can be approximated by the product of a "distance" and the overlap between the old and new distributions.

Proxy expression for one-iter forgetting. Under the assumptions and approximations in the derivation outline below, the one-iteration forgetting $\Delta L_{k,t}^{(1)}$ can be approximated using the Fisher divergence D_F, the overlap OV, and $c_t > 0$ that depends only on the timestep t:

$$\Delta L_{k,t}^{(1)} \lesssim c_t\, D_F(p_{\text{new},t} \| \, p_{k,t}) \cdot \text{OV}\big(p_{k,t} * \mathcal{N}(0, \bar{\sigma}_t I),\, p_{\text{new},t}\big). \tag{7}$$

Here, $*$ denotes the convolution operation. The terms are defined as follows:

$$D_F(p \| \, q) := \int \|\nabla_x \log p(x) - \nabla_x \log q(x)\|^2 \, p(x)\, \mathrm{d}x,$$

$$\text{OV}(p, q) := \langle p, q \rangle_{L^2} = \int p(x) q(x)\, \mathrm{d}x.$$

We now provide an intuitive interpretation of this proxy-score expression. The Fisher divergence D_F captures the mismatch between the scores of the new and old distributions. In contrast, the overlap term OV is proportional to the smoothed old-data density. This term reflects how strongly the new-task update hits regions where old-task data density is high, i.e., regions where updates are most damaging for the old task. Therefore, the proxy score can be understood as the product of "update magnitude" (the distributional "distance" D_F) and "where the update lands" (the distributional overlap OV).

This multiplicative form is consistent with prior findings that forgetting is maximized at a moderate level of task similarity [27]–[29]. When the new and old distributions are too similar, D_F becomes small; when they are too dissimilar, OV becomes small. Therefore, their product tends to be maximized at an intermediate distance.

Since we only need the relative ranking across classes, we ignore the constant c_t and define the theoretical forgetting-risk proxy score as $R(k \mid t) := D_F(p_{\text{new},t} \| p_{k,t}) \cdot \text{OV}\big(p_{k,t} * \mathcal{N}(0, \bar{\sigma}_t I),\, p_{\text{new},t}\big)$, which depends only on data distributions.

2) Derivation outline: The approximation in Eq. (7) follows from the five steps below, under several simplifying assumptions.

Step 1: Second-order Taylor approximation. We take one gradient descent step on the new-task loss with learning rate η: $\theta^{(1)} = \theta_{\text{old}} - \eta g_{\text{new},t}(\theta_{\text{old}})$, where $g_{\text{new},t}(\theta) := \nabla_\theta L_{\text{new},t}(\theta)$. By Definition 1, $\Delta L_{k,t}^{(1)} = L_{k,t}(\theta^{(1)}) - L_{k,t}(\theta_{\text{old}})$. Assuming score consistency for the old class k at θ_{old}, $s_{\theta_{\text{old}}}(x_t, t, k) \approx s_{k,t}(x_t)$, by Eq. (3) we have $L_{k,t}(\theta_{\text{old}}) \approx \text{const}$ and $\nabla_\theta L_{k,t}(\theta_{\text{old}}) \approx 0$. Thus, a second-order Taylor expansion of $L_{k,t}(\theta)$ around θ_{old} yields

$$\Delta L_{k,t}^{(1)} \approx \frac{\eta^2}{2} \|g_{\text{new},t}(\theta_{\text{old}})\|^2_{H_{k,t}(\theta_{\text{old}})}, \tag{8}$$

where $\|v\|_A^2 := v^\top A v$, and

$$g_{\text{new},t}(\theta_{\text{old}}) = 2\bar{\sigma}_t \mathbb{E}_{x_t \sim p_{\text{new},t}} \left[J_{\text{new}}(x_t)^\top e_{\text{new},t}(x_t) \right],$$
$$H_{k,t}(\theta_{\text{old}}) = 2\bar{\sigma}_t \mathbb{E}_{x_t \sim p_{k,t}} \left[J_k(x_t)^\top J_k(x_t) \right].$$

Here, $J_c(x_t) := \frac{\partial s_\theta(x_t, t, c)}{\partial \theta}\big|_{\theta=\theta_{\text{old}}}$ and $e_{\text{new},t}(x_t) := s_{\theta_{\text{old}}}(x_t, t, \text{new}) - s_{\text{new},t}(x_t)$.

Step 2: Jensen's inequality. Because $H_{k,t}(\theta_{\text{old}})$ is positive semidefinite, applying Jensen's inequality converts the parameter-space quadratic form in Eq. (8) into an expectation under the new-task input distribution $p_{\text{new},t}$. Defining the induced curvature matrix in score space $B_{k,t}(x_t) := J_{\text{new}}(x_t) H_{k,t}(\theta_{\text{old}}) J_{\text{new}}(x_t)^\top$ and absorbing t-dependent constants into $c_t > 0$, we obtain

$$\Delta L_{k,t}^{(1)} \lesssim c_t \mathbb{E}_{x_t \sim p_{\text{new},t}} \left[\|e_{\text{new},t}(x_t)\|^2_{B_{k,t}(x_t)} \right]. \tag{9}$$

Step 3: Local RBF approximation of the NTK. Next, we approximate $B_{k,t}(x_t)$ as a function of the density in the diffusion-model input space. We define the empirical cross neural tangent kernel (NTK) as $K_{\text{new},k}(x_t, x_t') := J_{\text{new}}(x_t) J_k(x_t')^\top$ [30], yielding $B_{k,t}(x_t) \propto \mathbb{E}_{x_t' \sim p_{k,t}} \left[K_{\text{new},k}(x_t, x_t') K_{\text{new},k}(x_t, x_t')^\top \right]$. Assuming locality and isotropy, we approximate $K_{\text{new},k}(x_t, x_t') K_{\text{new},k}(x_t, x_t')^\top$ by an isotropic RBF kernel [31]. We set the bandwidth to $\sqrt{\bar{\sigma}_t}$ to match the typical scale induced by the forward process. With this approximation, $B_{k,t}(x_t)$ becomes proportional to a Gaussian-smoothed version of the old distribution:

$$B_{k,t}(x_t) \approx \mathbb{E}_{x_t' \sim p_{k,t}} \left[\exp\left(-\tfrac{1}{2\bar{\sigma}_t} \|x_t - x_t'\|^2\right) \right] I \propto w_{k,t}(x_t) I, \tag{10}$$

where $w_{k,t}(x_t) := \left(p_{k,t} * \mathcal{N}(0, \bar{\sigma}_t I)\right)(x_t)$. Thus, we approximate $B_{k,t}(x_t)$ by the isotropic form $w_{k,t}(x_t) I$, discarding directional NTK information:

$$\Delta L_{k,t}^{(1)} \lesssim c_t \mathbb{E}_{x_t \sim p_{\text{new},t}} \left[\|e_{\text{new},t}(x_t)\|^2 \, w_{k,t}(x_t) \right].$$

Step 4: Overlap-Localized Condition Indistinguishability. We next separate the score error into the old–new score difference and the pre-update model's condition-mismatch term, using score consistency on class k: $e_{\text{new},t}(x_t) \approx \Delta s(x_t) + \delta(x_t)$, where $\Delta s(x_t) := s_{k,t}(x_t) - s_{\text{new},t}(x_t)$ and $\delta(x_t) := s_{\theta_{\text{old}}}(x_t, t, \text{new}) - s_{\theta_{\text{old}}}(x_t, t, k)$. By Young's inequality, $\|e_{\text{new},t}(x_t)\|^2 \lesssim 2\|\Delta s(x_t)\|^2 + 2\|\delta(x_t)\|^2$. Because $w_{k,t}$ localizes the expectation to the overlap region, we assume that in this region the pre-update model does not distinguish the new condition from class k, so $\mathbb{E}_{x_t \sim p_{\text{new},t}}[w_{k,t}(x_t)\|\delta(x_t)\|^2] \approx 0$.

$$\Delta L_{k,t}^{(1)} \lesssim c_t \mathbb{E}_{x_t \sim p_{\text{new},t}} \left[\|\Delta s(x_t)\|^2 \, w_{k,t}(x_t) \right].$$

Step 5: Ignoring covariance. Finally, we ignore the covariance between $\|\Delta s(x_t)\|^2$ and $w_{k,t}(x_t)$ and factorize $\mathbb{E}\left[\|\Delta s\|^2 w_{k,t}\right] \approx \mathbb{E}\left[\|\Delta s\|^2\right] \mathbb{E}\left[w_{k,t}\right]$, which yields Eq. (7). The two terms can be negatively correlated, so the factorization often overestimates the joint expectation in practice.

C. Empirical Evidence for the Proxy

Complementing the theoretical analysis in Sec. III-B, we provide empirical evidence that one-iteration forgetting is a useful low-cost proxy for replay allocation. Ideally, a replay strategy in CL should minimize long-term forgetting and preserve the final generation quality. However, evaluating such long-term outcomes in advance is difficult under compute constraints. To address this issue, we hypothesize that the loss change after one training iteration reflects long-term forgetting tendencies.

In preliminary experiments, we observed that the one-iteration forgetting $\Delta L_{k,t}^{(1)}$ exhibits a positive correlation with the long-term forgetting $\Delta L_{k,t}^{(I_{\text{final}})}$. We used Stable Diffusion v1.5 and an ImageNet-100 class-incremental setting (see Sec. IV for details), and measured, for each timestep t, the rank correlation (Spearman) and linear correlation (Pearson) across classes. As shown in Fig. 2, we observed positive correlations at many timesteps, and also a moderate correlation on average across timesteps (rank correlation ≈ 0.35, linear correlation ≈ 0.38). This result suggests that even one-iteration forgetting can predict long-term forgetting tendencies to some extent.

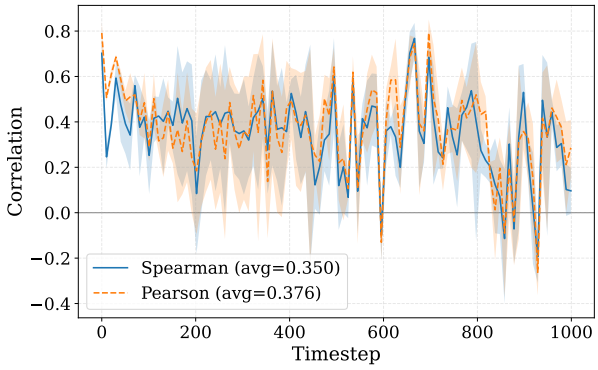

Fig. 2. Correlation between one-iteration forgetting $\Delta L_{k,t}^{(1)}$ and long-term forgetting $\Delta L_{k,t}^{(I_{\text{final}})}$ after $I_{\text{final}} = 2000$ iterations. We simulated the first incremental task (10 new classes) of ImageNet-100 with Stable Diffusion v1.5; training setup follows Sec. IV. We used 16-step gradient accumulation for the $\Delta L_{k,t}^{(1)}$ update. Losses were estimated using 256 samples per (k, t) with fixed random seeds. Correlations were computed across 10 classes for each of 100 equal-width timestep bins over $t \in [1, 1000]$. We averaged correlations across 3 seeds via Fisher's z transform (shaded: standard error).

D. Practical Risk Estimation

Since the theoretical forgetting-risk proxy score $R(k \mid t)$ involves intractable high-dimensional integrals, we derive a

closed-form approximation efficiently computable using class-wise summary statistics (means and per-dimension variances). To this end, we work in a globally standardized space and adopt a diagonal-Gaussian approximation for each class distribution.

1) Setup and Approximations: Approximating high-dimensional distributions with a simple diagonal Gaussian is generally coarse because it ignores cross-dimensional correlations. To keep the approximation practically accurate, we model distributions in a globally standardized space.

Global standardization. Let μ_{glob} and $\sigma^2_{\text{glob}} \in \mathbb{R}^d$ be the mean and per-dimension variance of all data observed up to the current task; we denote the corresponding standard deviation by σ_{glob}. We standardize raw data x^{raw} by $x = (x^{\text{raw}} - \mu_{\text{glob}}) \odot \sigma^{-1}_{\text{glob}}$, where $\odot$ denotes element-wise product. This per-dimension rescaling mitigates scale imbalance across dimensions, preventing features with large variances from dominating the proxy score. Because the same linear transform is applied to all classes and timesteps, it largely preserves the relative ranking of $R(k \mid t)$; any residual scale change can be absorbed into β.

Diagonal-Gaussian approximation. In the standardized space, we approximate each class distribution p_c and its distribution at timestep t, $p_{c,t}$, by diagonal Gaussians. Let $p_c \approx \mathcal{N}(\mu_{c,0}, \text{diag}(\sigma^2_{c,0}))$, where $\mu_{c,0}$ and $\sigma^2_{c,0}$ are the mean and per-dimension variance of class c in the standardized space; we denote the per-dimension standard deviation by $\sigma_{c,0}$. Then, by the property of the diffusion forward process (Eq. (1)), $p_{c,t}$ becomes a diagonal Gaussian with the following mean and per-dimension variance:

$$\mu_{c,t} = \sqrt{\bar{\alpha}_t}\mu_{c,0}, \quad \sigma^2_{c,t} = \bar{\alpha}_t\sigma^2_{c,0} + \bar{\sigma}_t\mathbf{1}, \tag{11}$$

where $\mathbf{1} \in \mathbb{R}^d$ denotes the all-ones vector. As t increases, $p_{c,t}$ approaches $\mathcal{N}(0, I)$, which further improves the diagonal approximation.

2) Closed-form Expression:

Definition 2 (Practical forgetting-risk proxy score): In the standardized space, we define the practical forgetting-risk proxy score $\tilde{R}(k \mid t)$ using the Fisher divergence D_F and the scaled overlap $\widetilde{\text{OV}}(p_{k,t}, p_{\text{new},t})$.

$$\tilde{R}(k \mid t) \coloneqq D_F(p_{\text{new},t} \| \, p_{k,t}) \cdot \widetilde{\text{OV}}(p_{k,t}, p_{\text{new},t})^{\gamma}, \tag{12}$$

where $\gamma > 0$ is an exponent parameter controlling the contribution of the overlap term. Here, $\widetilde{\text{OV}}$ normalizes and dimension-scales the L^2 inner product using a cosine-similarity form to mitigate extreme values in high dimensions.

$$\widetilde{\text{OV}}(p_{c,t}, p_{c',t}) \coloneqq \left(\frac{\langle \tilde{p}_{c,t}, \tilde{p}_{c',t} \rangle_{L^2}}{\|\tilde{p}_{c,t}\|_{L^2} \, \|\tilde{p}_{c',t}\|_{L^2}} \right)^{1/\sqrt{d}}, \tag{13}$$

where $\tilde{p}_{c,t} \coloneqq p_{c,t} * \mathcal{N}(0, (\bar{\sigma}_t/2)I)$.

Unlike the model-dependent one-iteration forgetting $\Delta L^{(1)}_{k,t}$, the practical proxy in Eq. (12) is parameter-independent, depends only on class-wise summary statistics, and can be computed with no additional diffusion-network forward passes.

Algorithm 1 Continual training with DOFR

Require: Initial params θ_{old}, Datasets $D_{\text{old}}, D_{\text{new}}$, Old raw stats $\{\mu^{\text{raw}}_{k,0}, \sigma^{2,\text{raw}}_{k,0}\}_k$, Running global stats $(\mu_{\text{glob}}, \sigma^2_{\text{glob}})$, Hyperparams β, γ

// Phase 1: Preparation (once per task)

Compute $\mu^{\text{raw}}_{\text{new},0}, \sigma^{2,\text{raw}}_{\text{new},0}$ from D_{new}.
Update $\mu_{\text{glob}}, \sigma^2_{\text{glob}}$ via Welford's alg.
Compute proxy-score table $\tilde{R}(k \mid t)$ via Eq. (14), (15) using standardized stats.
Construct sampling table $q_{\text{old}}(k \mid t) \propto \tilde{R}(k \mid t)^{\beta}$.
Store raw stats $\{\mu^{\text{raw}}_{c,0}, \sigma^{2,\text{raw}}_{c,0}\}_c$ for next task.

// Phase 2: Training

for each training iteration **do**
  Sample a mini-batch of timesteps $t \sim q_{\text{new}}(t)$.
  Sample old classes $k \sim q_{\text{old}}(k \mid t)$ for each t.
  Sample data $x_{\text{old}} \sim D^{(k)}_{\text{old}}$ and $x_{\text{new}} \sim D_{\text{new}}$ uniformly.
  Update θ using mini-batch gradients of Eq. (4).
end for

Closed-form expression for the practical risk. Under the diagonal-Gaussian approximation in the standardized space, the practical forgetting-risk proxy score $\tilde{R}(k \mid t)$ is computed in closed form using only the class statistics $\mu_{c,t}$ and $\sigma^2_{c,t}$:

$$\begin{aligned} D_F(p_{\text{new},t} \| \, p_{k,t}) = \left\| \sigma^{-2}_{k,t} \odot (\mu_{\text{new},t} - \mu_{k,t}) \right\|_2^2 \\ + \left\| \sigma_{\text{new},t} \odot \left(\sigma^{-2}_{k,t} - \sigma^{-2}_{\text{new},t} \right) \right\|_2^2, \end{aligned} \tag{14}$$

$$\begin{aligned} \widetilde{\text{OV}}(p_{k,t}, p_{\text{new},t}) = \frac{|\,\text{diag}(2\sigma'_{k,t})|^{\frac{1}{4\sqrt{d}}} \, |\,\text{diag}(2\sigma'_{\text{new},t})|^{\frac{1}{4\sqrt{d}}}}{|\,\text{diag}(\sigma'_{\text{mix},t})|^{\frac{1}{2\sqrt{d}}}} \\ \cdot \exp\left(-\frac{1}{2\sqrt{d}} \|\mu_{k,t} - \mu_{\text{new},t}\|^2_{\text{diag}(\sigma'_{\text{mix},t})^{-1}} \right). \end{aligned} \tag{15}$$

Here, $\sigma'_{c,t} = \sigma^2_{c,t} + (\bar{\sigma}_t/2)\mathbf{1}$, $\sigma'_{\text{mix},t} = \sigma'_{k,t} + \sigma'_{\text{new},t}$, and $|\cdot|$ denotes the matrix determinant. Corresponding to the two factors in Eq. (7), Eq. (14) quantifies mismatch between $p_{\text{new},t}$ and $p_{k,t}$, while Eq. (15) captures their normalized overlap after smoothing. In closed form, Eq. (14) depends on differences in means and per-dimension spreads, while Eq. (15) depends on similarity of the smoothed spreads and closeness of the means.

E. Algorithm and Complexity

1) Algorithm: We summarize the overall procedure of DOFR in Algorithm 1. DOFR has two phases: preparation (once per task) and training.

Preparation phase. When a new task arrives, before training we construct a replay-probability table $q_{\text{old}}(k \mid t)$ over old classes. First, we compute the mean and per-dimension variance for each new class $c \in C_{\text{new}}$ in the diffusion-model input space. For proxy-score computation, we obtain the aggregate new-task statistics ($\mu^{\text{raw}}_{\text{new},0}$, $\sigma^{2,\text{raw}}_{\text{new},0}$) by pooling per-class statistics (weighted by sample counts). Additionally, we update the global statistics $(\mu_{\text{glob}}, \sigma^2_{\text{glob}})$ using Welford's algorithm [32] to enable accurate streaming updates without reloading old data, and store per-class statistics for future

replay. Using the aggregate new-task statistics, we compute the proxy scores $\tilde{R}(k \mid t)$ associated with Eqs. (14) and (15) to construct the sampling table $q_{\text{old}}(k \mid t)$.

Training phase. Inside the training loop, we sample replay data simply by looking up the precomputed table $q_{\text{old}}(k \mid t)$.

2) Complexity Analysis:

Time complexity. The preparation phase costs $O(N_{\text{new}}d)$ for updating statistics and $O(K_{\text{old}}Td)$ for computing proxy score, where N_{new} is the number of new samples and K_{old} is the number of old classes.

Storage complexity. The additional storage overhead consists of only per-class and global summary statistics, totaling $O(Kd)$, where K is the total number of observed classes.

IV. Experiments

We evaluated DOFR to validate its effectiveness in compute-budgeted CL of diffusion models on ImageNet-100 [7] and Food-101-Subset (based on Food-101 [8]) using Stable Diffusion v1.5 (SDv1.5) [3].

A. Experimental setup

1) Datasets and task sequence: ImageNet-100 contains 100 classes from ImageNet-1k [33], [34] (about 125k training images and 5k test images). Food-101-Subset contains 100 classes randomly sampled from Food-101 (75k training images and 25k test images).

We split the 100 classes into 10 class-incremental tasks of 10 classes each and fixed the task order with one random seed. We retained access to all old data and added new-task data at each task arrival. We trained tasks sequentially and controlled the per-task compute budget by the number of training iterations per task (Iter/Task).

2) Compared methods and compute budgets: We compared methods that differ only in the timestep-conditional replay distribution over old classes, $q_{\text{old}}(k \mid t)$, under the same compute budget. Iter/Task counts only training iterations and excludes auxiliary compute overhead. We excluded online methods that estimate forgetting during training and update the sampling distribution (e.g., [12]), because converting such overhead into training iterations is protocol-dependent. We used two simple replay baselines: Random, which samples uniformly over all old classes, and Recent-task, which samples uniformly over the classes in the most recent old task. Random is known to be a strong baseline in compute-budgeted CL [6]. DOFR samples old classes based on the practical forgetting-risk proxy score: $q_{\text{old}}(k \mid t) \propto \tilde{R}(k \mid t)^{\beta}$. As a reference point, we also report Full-retraining, which trains on the union of all task data (non-CL).

We evaluated three compute budgets, corresponding to 20/40/60% of the per-task equivalent of Full-retraining to cover representative low-to-medium budgets. For ImageNet-100, Full-retraining used 35,000 total iterations, corresponding to 3,500 iterations per task (10 tasks); we set Iter/Task $\in \{700, 1400, 2100\}$. For Food-101-Subset, Full-retraining used 25,000 total iterations; we set Iter/Task $\in \{500, 1000, 1500\}$.

3) Model and training: We used SDv1.5 [3] as the base text-to-image diffusion model. For class-conditioned generation, we used a fixed prompt template, `a photo of {class_name}`, conditioning on the class name only. We applied LoRA [35] to the cross- and self-attention layers of the U-Net. We used LoRA with rank 16 and $\alpha = 16$.

We trained with AdamW (learning rate 10^{-4} and weight decay 10^{-2}) and used a batch size of 64. We used the standard uniform timestep sampling distribution during training. We set ρ in Eq. (4) proportional to dataset sizes, while ensuring that each new-task sample was seen at least once at low Iter/Task. We sampled new-task data without replacement and old-task data with replacement. In DOFR, we computed $\tilde{R}(k \mid t)$ from latents encoded by the fixed VAE of SDv1.5, because the diffusion model operates on that latent space. To simulate a realistic scenario, we fixed the hyperparameters across all tasks. We determined their values empirically based on preliminary experiments on the first incremental task: $(\beta, \gamma) = (1, 1)$ for ImageNet-100 and $(1.5, 0.75)$ for Food-101-Subset. All experiments used four NVIDIA A100 (80GB) GPUs.

4) Metrics and generation: We evaluated the model after the final task, since our goal is to maximize final generation quality under a fixed compute budget. We used CMMD [36] and FID [37] for visual quality, and CLIPScore [38] for prompt fidelity. We reported forgetting only to validate our proxy (Fig. 2), rather than as a primary metric. We prioritized CMMD because it mitigates FID's sensitivity to sample size [36], which is crucial for ImageNet-100 (5k test images).

For evaluation, we generated the same number of images as the test-set size of each dataset. We used the DDIM sampler [39] with 50 steps and the default scaled-linear schedule. We used classifier-free guidance (CFG) [40] for conditional generation. We set the guidance scale to 4, which performed best in a small grid search on the pretrained model.

B. Main results

Table I shows performance after the final task. DOFR improves CMMD over Random across both datasets and keeps FID and CLIPScore comparable. The gain is largest at the lowest budget (e.g., ImageNet-100, Iter/Task = 700: 0.276 → 0.265; Food-101-Subset, Iter/Task = 500: 0.442 → 0.431). DOFR also matches or outperforms Recent-task in CMMD, while consistently improving FID and CLIPScore. These results suggest that DOFR improves the compute–performance trade-off by allocating limited replay budget toward classes with higher estimated forgetting risk at each timestep. However, a gap to Full-retraining still remained.

C. Additional Experiments

We present two additional analyses: (1) an ablation study of the proxy-score components and (2) a wall-clock time breakdown.

1) Ablation study: We ran an ablation study on ImageNet-100 to isolate the impact of each component in the forgetting-risk proxy definition. We kept the training and evaluation setup fixed and changed only the components used to form

TABLE I

MAIN RESULTS ON (A) IMAGENET-100 AND (B) FOOD-101-SUBSET AFTER THE FINAL TASK. ROWS: METHODS; COLUMNS: ITER/TASK (COMPUTE BUDGET). ARROWS INDICATE BETTER. VALUES ARE MEAN ± STANDARD ERROR OVER 3 SEEDS.

	CMMD (↓)			FID (↓)			CLIPScore (↑)		
Method \ Iter/Task	700	1400	2100	700	1400	2100	700	1400	2100
Recent-task	0.278±0.003	0.283±0.003	0.266±0.001	79.44±0.16	78.14±0.30	78.69±0.34	31.76±0.04	31.75±0.02	31.71±0.02
Random	0.276±0.003	0.255±0.003	0.244±0.006	78.47±0.22	77.58±0.54	**76.80±0.33**	32.03±0.01	32.11±0.05	32.14±0.07
DOFR	**0.265±0.003**	**0.247±0.004**	**0.242±0.001**	**78.27±0.13**	**76.98±0.27**	76.88±0.25	**32.08±0.02**	**32.18±0.04**	**32.16±0.02**
Full-retraining		0.227±0.004			75.12±0.21			32.31±0.02	

(a) ImageNet-100

	CMMD (↓)			FID (↓)			CLIPScore (↑)		
Method \ Iter/Task	500	1000	1500	500	1000	1500	500	1000	1500
Recent-task	0.438±0.004	0.421±0.002	**0.394±0.007**	52.92±0.18	53.69±0.14	52.94±0.33	32.49±0.03	32.53±0.01	32.52±0.01
Random	0.442±0.005	0.414±0.004	0.400±0.002	52.90±0.29	52.32±0.24	52.86±0.27	32.46±0.01	32.51±0.05	**32.60±0.05**
DOFR	**0.431±0.006**	**0.409±0.003**	**0.394±0.003**	**52.57±0.44**	**52.22±0.14**	**52.35±0.31**	**32.61±0.00**	**32.56±0.04**	32.59±0.02
Full-retraining		0.367±0.001			52.90±0.18			32.75±0.03	

(b) Food-101-Subset

TABLE II

ABLATION ON IMAGENET-100 (CMMD ↓). D_F: FISHER DIVERGENCE; $\widetilde{\text{OV}}$: OVERLAP; t-COND.: TIMESTEP-CONDITIONED SAMPLING.

			Iter/Task		
D_F	$\widetilde{\text{OV}}$	t-cond.	700	1400	2100
	✓	✓	0.273±0.003	0.254±0.001	0.249±0.002
✓		✓	0.271±0.006	0.253±0.004	0.247±0.002
✓	✓		0.269±0.001	0.256±0.003	0.246±0.002
✓	✓	✓	**0.265±0.003**	**0.247±0.004**	**0.242±0.000**

TABLE III

PER-TASK WALL-CLOCK TIME BREAKDOWN ON IMAGENET-100 AT ITER/TASK = 1400, AVERAGED OVER TASKS 2–10.

	Avg. Time per Task [s]	% of Total
VAE preprocessing	115.2	6.50
Preparation phase	0.8	0.05
Training phase	1655.6	93.45
Total	1771.6	100

$q_{\text{old}}(k \mid t)$. We tested three variants: (i) D_F only, (ii) $\widetilde{\text{OV}}$ only, and (iii) the marginal distribution $q_{\text{old}}(k)$ instead of timestep-conditioned sampling (the t-average of $q_{\text{old}}(k \mid t)$). We used the same hyperparameters as in the main results. Table II shows that CMMD worsens when either D_F or $\widetilde{\text{OV}}$ is removed, or when timestep-conditioned class sampling is replaced with $q_{\text{old}}(k)$, indicating that each of the three components contributes to DOFR's performance in this setting.

2) Wall-clock time breakdown: To quantify the contribution of the Preparation phase to total wall-clock time, Table III reports the per-task wall-clock time breakdown on ImageNet-100 at Iter/Task = 1400. Because our experiments use latent diffusion, VAE preprocessing is shown separately from DOFR's Preparation and Training phases. The Preparation phase required only 0.8 s/task (0.05% of total wall-clock time), compared with 1655.6 s/task (93.45%) for the Training phase, suggesting that the Preparation phase contributes little to total wall-clock time in this setting.

V. CONCLUSION

We proposed DOFR, a replay method for compute-budgeted continual learning of diffusion models based on one-iteration forgetting. In our experiments, DOFR improved overall generation quality relative to simple replay baselines with no extra diffusion-network forward passes. These results suggest that, in diffusion-model continual learning, choosing which old class to replay at each timestep can be an important factor in compute-budgeted settings.

A natural next step is to integrate DOFR with principled timestep sampling strategies. This work focused on class selection ($q(k \mid t)$), but jointly designing the timestep sampling distribution ($q(t)$) with DOFR could further improve efficiency. Another direction is to refine the proxy score by leveraging conditioning information. DOFR uses summary statistics in the diffusion-model input space, but in conditional generation, relationships among condition vectors may also affect forgetting.

As training costs grow with model scale, we expect compute-efficient continual learning to be a stepping stone toward sustainable high-performance generative AI, including diffusion models.

ACKNOWLEDGMENT

We used Microsoft Copilot [41] to draft and revise the English text from our original manuscript throughout the paper. We are responsible for all technical content and conclusions.

References

[1] J. Ho, A. Jain, and P. Abbeel, "Denoising Diffusion Probabilistic Models," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 6840–6851.

[2] Y. Song and S. Ermon, "Generative Modeling by Estimating Gradients of the Data Distribution," in *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019.

[3] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-Resolution Image Synthesis With Latent Diffusion Models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 684–10 695.

[4] M. McCloskey and N. J. Cohen, "Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem," in *Psychology of Learning and Motivation*. Elsevier, 1989, vol. 24, pp. 109–165.

[5] L. Wang, X. Zhang, H. Su, and J. Zhu, "A Comprehensive Survey of Continual Learning: Theory, Method and Application," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 8, pp. 5362–5383, Aug. 2024.

[6] A. Prabhu, H. A. Al Kader Hammoud, P. K. Dokania, P. H. S. Torr, S.-N. Lim, B. Ghanem, and A. Bibi, "Computationally Budgeted Continual Learning: What Does Matter?" in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 3698–3707.

[7] Y. Tian, D. Krishnan, and P. Isola, "Contrastive Multiview Coding," in *Computer Vision – ECCV 2020*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Cham: Springer International Publishing, 2020, vol. 12356, pp. 776–794.

[8] L. Bossard, M. Guillaumin, and L. Van Gool, "Food-101 – Mining Discriminative Components with Random Forests," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, vol. 8694, pp. 446–461.

[9] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-Based Generative Modeling through Stochastic Differential Equations," in *International Conference on Learning Representations*, Oct. 2020.

[10] P. Vincent, "A Connection Between Score Matching and Denoising Autoencoders," *Neural Computation*, vol. 23, no. 7, pp. 1661–1674, 2011.

[11] E. Verwimp, G. Hacohen, and T. Tuytelaars, "Same accuracy, twice as fast: Continuous training surpasses retraining from scratch," Feb. 2025.

[12] J. S. Smith, L. Valkov, S. Halbe, V. Gutta, R. Feris, Z. Kira, and L. Karlinsky, "Adaptive Memory Replay for Continual Learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 3605–3615.

[13] M. Zając, K. Deja, A. Kuzina, J. M. Tomczak, T. Trzciński, F. Shkurti, and P. Miłoś, "Exploring Continual Learning of Diffusion Models," Mar. 2023.

[14] S. Masip, P. Rodriguez, T. Tuytelaars, and G. M. van de Ven, "Continual Learning of Diffusion Models with Generative Distillation," in *Proceedings of The 3rd Conference on Lifelong Learning Agents*. PMLR, Feb. 2025, pp. 431–456.

[15] H. Zhang, J. Zhou, H. Lin, H. Ye, J. Zhu, Z. Wang, L. Gao, Y. Wang, and Y. Liang, "CLoG: Benchmarking Continual Learning of Image Generation Models," in *NeurIPS 2024 Workshop on Scalable Continual Learning for Lifelong Foundation Models*, Oct. 2024.

[16] J. S. Smith, Y.-C. Hsu, L. Zhang, T. Hua, Z. Kira, Y. Shen, and H. Jin, "Continual Diffusion: Continual Customization of Text-to-Image Diffusion with C-LoRA," *Transactions on Machine Learning Research*, Jan. 2024.

[17] J. Dong, W. Liang, H. Li, D. Zhang, M. Cao, H. Ding, S. Khan, and F. S. Khan, "How to Continually Adapt Text-to-Image Diffusion Models for Flexible Customization?" *Advances in Neural Information Processing Systems*, vol. 37, pp. 130 057–130 083, Dec. 2024.

[18] Ł. Staniszewski, K. Zaleska, and K. Deja, "Low-Rank Continual Personalization of Diffusion Models," Mar. 2025.

[19] R. Gao and W. Liu, "DDGR: Continual Learning with Deep Diffusion-based Generative Replay," in *Proceedings of the 40th International Conference on Machine Learning*. PMLR, Jul. 2023, pp. 10 744–10 763.

[20] Z. Meng, J. Zhang, C. Yang, Z. Zhan, P. Zhao, and Y. Wang, "DiffClass: Diffusion-Based Class Incremental Learning," in *Computer Vision – ECCV 2024*, A. Leonardis, E. Ricci, S. Roth, O. Russakovsky, T. Sattler, and G. Varol, Eds. Cham: Springer Nature Switzerland, 2025, vol. 15145, pp. 142–159.

[21] B. Cywiński, K. Deja, T. Trzciński, B. Twardowski, and Ł. Kuciński, "GUIDE: Guidance-based Incremental Learning with Diffusion Models," May 2024.

[22] J. Choi, J. Lee, C. Shin, S. Kim, H. Kim, and S. Yoon, "Perception Prioritized Training of Diffusion Models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 472–11 481.

[23] T. Hang, S. Gu, C. Li, J. Bao, D. Chen, H. Hu, X. Geng, and B. Guo, "Efficient Diffusion Training via Min-SNR Weighting Strategy," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 7441–7451.

[24] K. Wang, M. Shi, Y. Zhou, Z. Li, Z. Yuan, Y. Shang, X. Peng, H. Zhang, and Y. You, "A Closer Look at Time Steps is Worthy of Triple Speed-Up for Diffusion Model Training," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 12 934–12 944.

[25] M. Kim, D. Ki, S.-W. Shim, and B.-J. Lee, "Adaptive Non-Uniform Timestep Sampling for Accelerating Diffusion Model Training," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 2513–2522.

[26] T. Hang, S. Gu, J. Bao, F. Wei, D. Chen, X. Geng, and B. Guo, "Improved Noise Schedule for Diffusion Training," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025, pp. 4796–4806.

[27] V. V. Ramasesh, E. Dyer, and M. Raghu, "Anatomy of Catastrophic Forgetting: Hidden Representations and Task Semantics," in *International Conference on Learning Representations*, Oct. 2020.

[28] S. Lee, S. Goldt, and A. Saxe, "Continual Learning in the Teacher-Student Setup: Impact of Task Similarity," in *Proceedings of the 38th International Conference on Machine Learning*. PMLR, Jul. 2021, pp. 6109–6119.

[29] D. Goldfarb, I. Evron, N. Weinberger, D. Soudry, and Pa. HAnd, "The Joint Effect of Task Similarity and Overparameterization on Catastrophic Forgetting — An Analytical Model," in *The Twelfth International Conference on Learning Representations*, Oct. 2023.

[30] J. Lee, L. Xiao, S. Schoenholz, Y. Bahri, R. Novak, J. Sohl-Dickstein, and J. Pennington, "Wide Neural Networks of Any Depth Evolve as Linear Models Under Gradient Descent," in *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019.

[31] B. Schölkopf, K. Tsuda, and J.-P. Vert, "A Primer on KernelMethods," in *Kernel Methods in Computational Biology*. MIT Press, 2004.

[32] B. P. Welford, "Note on a Method for Calculating Corrected Sums of Squares and Products," *Technometrics*, vol. 4, no. 3, pp. 419–420, 1962.

[33] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. Miami, FL: IEEE, Jun. 2009, pp. 248–255.

[34] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[35] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-Rank Adaptation of Large Language Models," in *International Conference on Learning Representations*, Oct. 2021.

[36] S. Jayasumana, S. Ramalingam, A. Veit, D. Glasner, A. Chakrabarti, and S. Kumar, "Rethinking FID: Towards a Better Evaluation Metric for Image Generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 9307–9315.

[37] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.

[38] J. Hessel, A. Holtzman, M. Forbes, R. Le Bras, and Y. Choi, "CLIPScore: A Reference-free Evaluation Metric for Image Captioning," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, 2021, pp. 7514–7528.

[39] J. Song, C. Meng, and S. Ermon, "Denoising Diffusion Implicit Models," in *International Conference on Learning Representations*, Oct. 2020.

[40] J. Ho and T. Salimans, "Classifier-Free Diffusion Guidance," in *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, Dec. 2021.

[41] "Microsoft Copilot," https://copilot.microsoft.com.