ReTrack: Data Unlearning in Diffusion Models Through Redirecting the Denoising Trajectory

Qitan Shi **Cheng Jin** **Jiawei Zhang** **Yuantao Gu**
Department of Electronic Engineering
Tsinghua University
Beijing, China

Abstract

Diffusion models excel at generating high-quality, diverse images but also suffer from undesirable training data memorization, raising critical privacy and safety concerns. Data unlearning has emerged to mitigate this issue by removing the influence of specific data through fine-tuning rather than retraining from scratch. We propose ReTrack, a fast and effective data unlearning method for diffusion models. ReTrack employs importance sampling to construct a more efficient unbiased fine-tuning loss. This loss is further approximated by retaining only the dominant terms, thereby reducing computational cost. This yields an interpretable objective that redirects denoising trajectories toward the k-nearest neighbors, enabling efficient unlearning while preserving generative quality. Experiments on MNIST T-Shirt, CelebA-HQ, CIFAR-10, and Stable Diffusion show that ReTrack achieves state-of-the-art performance, striking the best trade-off between unlearning strength and generation quality preservation.

1 INTRODUCTION

In recent years, diffusion models have brought about a revolutionary breakthrough in image generation. Representative models such as Denoising Diffusion Probabilistic Models (Ho, Jain, and Abbeel, 2020) and Stable Diffusion (Rombach et al., 2022) have completely transformed the field of generative artificial intelligence with their exceptional image generation quality and powerful expressive capabilities. These models can generate highly realistic images, demonstrating unprecedented potential in various applications including artistic creation (Wu, 2022; Zhang et al., 2023), image inpainting (Lugmayr et al., 2022; Manukyan et al., 2023), super-resolution (Li et al., 2022; Saharia et al., 2022) and data augmentation (Alimisis et al., 2025).

With their growing capacity, diffusion models have shown a pronounced tendency to memorize training data. Several studies have demonstrated that these models can unintentionally reproduce specific training samples (Gu et al., 2023; Somepalli et al., 2023; Carlini et al., 2023; Chen et al., 2024). Given that the training data of such models are typically sourced from heterogeneous web scrapes, user uploads, or open datasets, they often contain potentially harmful, sensitive, or copyrighted content. The memorization of such data raises privacy, legal, and ethical concerns.

In order to eliminate the influence of these data on the model, the ideal approach would be to retrain the model after removing these data from the training dataset. However, this approach incurs prohibitively large computational cost and is often impractical (Xu et al., 2024; Wang et al., 2024). A research field aiming to modify a pretrained model so that it behaves as though it had never been trained on some data in the training dataset without retraining the whole model is known as *machine unlearning*. Although there have been some early works in the machine unlearning field, most of these works either requires modifications to the pretraining phase (Wu, Dobriban, and Davidson, 2020; Bourtoule et al., 2021; Graves, Nagisetty, and Ganesh, 2021) or is specifically designed for supervised machine learning (Golatkar, Achille, and Soatto, 2020; Izzo et al., 2021; Wang et al., 2022; Perifanis et al., 2024) and cannot be directly applied to pretrained diffusion models.

To handle this problem, recent works have begun ex-

Proceedings of the 29[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

ploring machine unlearning methods specifically designed for diffusion models. These works can be broadly categorized into *concept unlearning* and *data unlearning* (Alberti et al., 2025), with this paper focusing on the latter. Unlike concept unlearning which aims to forget harmful or inappropriate abstract concepts in text-conditioning such as "nudity" and "violence" (Fan et al., 2023; Gandikota et al., 2023, 2024; Zhang et al., 2024; Wu et al., 2025), data unlearning seeks to achieve a sample-level unlearning, i.e., to remove the influence of specific illicit or privacy-sensitive data in the training dataset on the diffusion models, while preserving the influence of other remaining data. For data unlearning, there are two relatively intuitive yet limited baseline methods: the vanilla method which directly fine-tunes the model on the remaining data by applying the standard diffusion loss, and NegGrad (Golatkar, Achille, and Soatto, 2020) which performs gradient ascent on the data to be unlearned. The former often fails to achieve true forgetting, and the latter usually leads to over-forgetting, causing a serious degradation of model generation quality. To address these shortcomings, more data unlearning methods have been proposed in recent years, including: Variational Diffusion Unlearning (VDU) (Panda et al., 2024) which incorporates variational inference techniques into the data unlearning for diffusion models and enables the removal of specific data without having access to the remaining data by approximating the posterior distribution over the parameters of the unlearned model; Subtracted Importance Sampled Scores (SISS) (Alberti et al., 2025) which leverages importance sampling by introducing a mixture distribution to combine the vanilla loss and the NegGrad loss, thereby enabling selective unlearning of specific data while preserving the model's generation quality.

In this work, we propose *ReTrack*, a data unlearning method for diffusion models by **Re**directing the denoising **Tra**je**c**tory to the **k**-nearest neighbors. First, based on the observation that the vanilla fine-tuning loss is unbiased but highly inefficient in sampling, we propose a new loss function derived from the vanilla loss that applies importance sampling to focus the unlearning process on samples that need to be unlearned, thereby greatly accelerating unlearning. Second, to address the challenge of exactly evaluating the full loss function, we approximate it with the k most important terms, which correspond to the k-nearest neighbors of the samples to be unlearned. Intuitively, the approximated loss function fine-tunes the model such that denoising trajectories originally directed toward samples to be unlearned are redirected to their k-nearest neighbors within all the remaining data. Since these neighbors are close in distance to the unlearned samples in data space, the required correction is small, which accelerates training and improves stability. Moreover, redirecting toward real and plausible targets helps maintain the quality of generated outputs during unlearning. Finally, we conduct experiments on MNIST T-Shirt, CelebA-HQ, CIFAR-10 and Stable Diffusion, and demonstrate that our method achieves the best unlearning results among all existing methods that are able to preserve the model's generation quality, meaning that our method strikes the best trade-off between unlearning strength and generation quality preservation.

Our main contributions can be summarized as follows:

- We propose a novel data unlearning fine-tuning loss that preserves unbiasedness while substantially improving sampling efficiency during fine-tuning, enabling more effective unlearning.
- We derive a truncated approximation to the proposed loss by retaining only the dominant terms, thereby reducing computational cost. We provide an intuitive explanation for this approximation.
- We conduct extensive experiments on MNIST T-Shirt, CelebA-HQ, CIFAR-10, and Stable Diffusion, demonstrating that our method achieves efficient unlearning while maintaining the generative quality of the models. Code is available at `https://github.com/sqt24/ReTrack`.

2 PRELIMINARY

2.1 Diffusion Models

Diffusion models are a class of powerful generative frameworks that learn complex data distributions by coupling two complementary processes: a *forward diffusion process*, which gradually perturbs the data with Gaussian noise, and a *reverse denoising process*, which learns to reconstruct the data from noise (Ho, Jain, and Abbeel, 2020).

Suppose the training dataset is A where the data dimension is d. In the forward process, given a training sample $\boldsymbol{a} \in A$ and a diffusion timestep $t \in \{1, 2, \cdots, T\}$ where T denotes the total number of diffusion steps, a noisy sample $\boldsymbol{x}_t$ at timestep t is produced according to

$$\boldsymbol{x}_t = \gamma_t \boldsymbol{a} + \sigma_t \boldsymbol{\epsilon},$$

where γ_t and σ_t are time-dependent coefficients specified by a predefined noise schedule, and $\boldsymbol{\epsilon} \sim \mathcal{N}(\cdot; \mathbf{0}, \mathbf{I}_d)$ denotes standard Gaussian noise. Conditioned on the clean input $\boldsymbol{a}$, the noisy sample $\boldsymbol{x}_t$ follows

$$q_t(\boldsymbol{x}_t|\boldsymbol{a}) = \mathcal{N}(\boldsymbol{x}_t; \gamma_t \boldsymbol{a}, \sigma_t^2 \mathbf{I}_d).$$

As t increases, the noisy sample $\boldsymbol{x}_t$ contains less and less data and more and more noise, and eventually be completely changed to Gaussian noise.

The reverse process is modeled by a neural network that aims to approximate the conditional expectation of the noise term $\boldsymbol{\epsilon}$ given the perturbed input $\boldsymbol{x}_t$ and the timestep t. Formally, the model $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$ parameterized by $\boldsymbol{\theta}$ is trained to minimize the expected squared error

$$\mathcal{L}_{\text{train}}(\boldsymbol{\theta}) = \mathbb{E}_{t,\boldsymbol{a}\sim A,\boldsymbol{x}_t\sim q_t(\cdot|\boldsymbol{a})}\left[\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)-\boldsymbol{\epsilon}\|^2\right],$$

This training objective encourages the model to approximate the true posterior mean $\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{\epsilon}|\boldsymbol{x}_t,t]$, which characterizes the reverse process in a variational framework. By training the denoising model $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$, the data distribution can be fitted indirectly.

At inference time, sample generation proceeds by initializing with a Gaussian noise $\boldsymbol{x}_T \sim \mathcal{N}(\cdot;\mathbf{0},\mathbf{I}_d)$ and iteratively applying the learned denoising model according to a time-discretized approximation of the reverse diffusion process, obtaining the denoising trajectory $\boldsymbol{x}_T, \boldsymbol{x}_{T-1}, \cdots, \boldsymbol{x}_1, \boldsymbol{x}_0$. This process gradually transforms Gaussian noise $\boldsymbol{x}_T$ into a data sample $\boldsymbol{x}_0$ following the learned data distribution, thereby enabling data generation.

2.2 Data Unlearning for Diffusion Models

2.2.1 Definition

The data unlearning problem is mainly studied for unconditional diffusion models, which can be formulated as follows: given a training dataset A and a diffusion model $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$ pretrained on A, our goal is to fine-tune the model with relatively modest computational cost so that it forgets the influence of a subset $A_u \subset A$ (referred to as the *unlearning set*), while preserving the influence of the remaining data $A_r = A \setminus A_u$ (referred to as the *remaining set*). In other words, we seek to obtain a fine-tuned model $\boldsymbol{\epsilon}_{\boldsymbol{\theta}^*}$ with comparatively low computational cost, such that it behaves as though it had been pretrained solely on A_r.

2.2.2 Prior Methods

Vanilla Vanilla serves as a baseline method that simply fine-tune the pretrained model only on remaining set A_r using the standard diffusion loss:

$$\mathcal{L}_{\text{vanilla}}(\boldsymbol{\theta}) = \mathbb{E}_{t,\boldsymbol{a}_r\sim A_r,\boldsymbol{x}_t\sim q_t(\cdot|\boldsymbol{a}_r)}\left[\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)-\boldsymbol{\epsilon}\|_2^2\right].$$

While intuitive, this method completely ignores any information about the unlearning set A_u. As a result, even after many fine-tuning steps, it often fails to effectively remove the influence of A_u.

NegGrad NegGrad (Golatkar, Achille, and Soatto, 2020) directly performs gradient ascent on samples from the unlearning set A_u to push the model's predictions away from the true noise estimations, which is equivalent to performing gradient descent on

$$\mathcal{L}_{\text{NegGrad}}(\boldsymbol{\theta}) = -\mathbb{E}_{t,\boldsymbol{a}_u\sim A_u,\boldsymbol{x}_t\sim q_t(\cdot|\boldsymbol{a}_u)}\left[\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)-\boldsymbol{\epsilon}\|_2^2\right].$$

However, NegGrad cannot guarantee that the model's performance on the remaining set A_r is preserved. It often leads to over-forgetting, whereby the model also forgets A_r while attempting to forget A_u.

EraseDiff Although EraseDiff (Wu et al., 2025) is originally designed for concept unlearning, following the setting described by Alberti et al. (2025), it can also be applied to data unlearning task by guiding the model's predictions on A_u toward pure noise:

$$\begin{aligned}\mathcal{L}_{\text{EraseDiff}}(\boldsymbol{\theta}) &= \mathbb{E}_{t,\boldsymbol{a}_r\sim A_r,\boldsymbol{x}_t\sim q_t(\cdot|\boldsymbol{a}_r)}\left[\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)-\boldsymbol{\epsilon}\|_2^2\right]\\ &+\lambda\mathbb{E}_{t,\boldsymbol{a}_u\sim A_u,\boldsymbol{x}_t\sim q_t(\cdot|\boldsymbol{a}_u),\boldsymbol{\epsilon}_u\sim\mathcal{U}[0,1]^d}\left[\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)-\boldsymbol{\epsilon}_u\|_2^2\right],\end{aligned}$$

where the first term is used to preserve A_r, and the second term is used to unlearn A_u by pushing the model's prediction toward pure noise. By solving an optimization problem on λ, EraseDiff ensures that the overall gradient descent direction of the loss function simultaneously serves as the descent direction for both of its constituent terms, thereby forgetting A_u while preserving A_r.

SISS SISS (Alberti et al., 2025) strikes a balance between the vanilla fine-tuning loss and the NegGrad loss, and improves computational efficiency by employing importance sampling:

$$\begin{aligned}\mathcal{L}_{\text{SISS}}(\boldsymbol{\theta}) = \mathbb{E}_{t,\boldsymbol{a}_r\sim A_r,\boldsymbol{a}_u\sim A_u,\boldsymbol{x}_t\sim q_t(\cdot|\boldsymbol{a}_r,\boldsymbol{a}_u)}&\\ \left[\frac{q_t(\boldsymbol{x}_t|\boldsymbol{a}_r)}{q_t(\boldsymbol{x}_t|\boldsymbol{a}_r,\boldsymbol{a}_u)}\left\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)-\frac{\boldsymbol{x}_t-\gamma_t\boldsymbol{a}_r}{\sigma_t}\right\|_2^2\right.&\\ \left.-s\frac{q_t(\boldsymbol{x}_t|\boldsymbol{a}_u)}{q_t(\boldsymbol{x}_t|\boldsymbol{a}_r,\boldsymbol{a}_u)}\left\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)-\frac{\boldsymbol{x}_t-\gamma_t\boldsymbol{a}_u}{\sigma_t}\right\|_2^2\right],&\end{aligned}$$

where the first term corresponds to the vanilla fine-tuning loss, the second term corresponds to the NegGrad loss, and

$$q_t(\boldsymbol{x}_t|\boldsymbol{a}_r,\boldsymbol{a}_u) = (1-\lambda)q_t(\boldsymbol{x}_t|\boldsymbol{a}_r)+\lambda q_t(\boldsymbol{x}_t|\boldsymbol{a}_u)$$

is a mixture distribution between $q_t(\boldsymbol{x}_t|\boldsymbol{a}_r)$ and $q_t(\boldsymbol{x}_t|\boldsymbol{a}_u)$ parameterized by $\lambda \in [0,1]$, and $s>0$ is another hyperparameter controlling the strength of unlearning.

VDU VDU (Panda et al., 2024) performs data unlearning by introducing a variational Bayesian framework. However, it relies on access to multiple independently pretrained diffusion models on the same

dataset, which is impractical under our experimental settings. Therefore, we exclude it from our subsequent comparisons.

3 METHODOLOGY

3.1 Motivation

We first review vanilla fine-tuning method, which directly performs denoising matching training on the remaining dataset, with the corresponding loss function given by

$$\mathcal{L}_{\text{vanilla}}(\boldsymbol{\theta}) = \mathbb{E}_{t,\boldsymbol{a}_r\sim A_r,\boldsymbol{x}_t\sim q_t(\cdot|\boldsymbol{a}_r)}\left[\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)-\boldsymbol{\epsilon}\|_2^2\right].$$

If one were to train a model from scratch with this objective, the resulting model would converge to the ideal solution trained only on A_r. Thus, $\mathcal{L}_{\text{vanilla}}$ is an *unbiased* fine-tuning objective, a property that is important for stability and for preserving generative quality after unlearning.

However, $\mathcal{L}_{\text{vanilla}}$ is highly *inefficient* for unlearning. Samples $\boldsymbol{x}_t$ are always generated from $\boldsymbol{a}_r \in A_r$, so training concentrates on regions surrounding A_r. In contrast, the regions where unlearning must occur are precisely the neighborhoods of A_u. As a result, the fine-tuned model's behavior near A_u remains largely unchanged, and the influence of A_u persists even after many steps of fine-tuning. This observation motivates us to improve upon the vanilla method by enhancing sampling efficiency, while retaining its favorable property of unbiasedness.

3.2 Importance-weighted Unlearning Objective

Inspired by the idea of importance sampling, we modify the sampling distribution of $\mathcal{L}_{\text{vanilla}}$ by shifting the sampling of $\boldsymbol{x}_t$ from regions concentrated around A_r to regions concentrated around A_u while maintaining the unbiasedness of the fine-tuning loss. This modification improves the sampling efficiency of the objective during fine-tuning and thus enables faster unlearning. The proposed fine-tuning loss is

$$\mathcal{L}_{\text{unlearn}}(\boldsymbol{\theta}) = \mathbb{E}_{t,\boldsymbol{a}_u\sim A_u,\boldsymbol{x}_t\sim q_t(\cdot|\boldsymbol{a}_u)} \left[\sum_{\boldsymbol{a}_r\in A_r} w_t(\boldsymbol{x}_t|\boldsymbol{a}_r)\left\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)-\frac{\boldsymbol{x}_t-\gamma_t\boldsymbol{a}_r}{\sigma_t}\right\|_2^2\right],$$

where the weight term

$$w_t(\boldsymbol{x}_t|\boldsymbol{a}_r) = \frac{q_t(\boldsymbol{x}_t|\boldsymbol{a}_r)}{\sum_{\boldsymbol{a}'_r\in A_r} q_t(\boldsymbol{x}_t|\boldsymbol{a}'_r)}$$

is introduced by the importance sampling procedure. Note that, via importance sampling, we change the sampling distribution of $\boldsymbol{x}_t$ from $\boldsymbol{x}_t \sim q_t(\cdot|\boldsymbol{a}_r)$ to $\boldsymbol{x}_t \sim q_t(\cdot|\boldsymbol{a}_u)$. The unbiasedness of $\mathcal{L}_{\text{unlearn}}$ is guaranteed by the following proposition:

Proposition 1. *The two fine-tuning loss functions $\mathcal{L}_{\text{vanilla}}$ and $\mathcal{L}_{\text{unlearn}}$ are equivalent.*

The proof is provided in the appendix. Although the two loss functions $\mathcal{L}_{\text{vanilla}}$ and $\mathcal{L}_{\text{unlearn}}$ are theoretically equivalent, in practice, when approximating the expectation via sampling, our method achieves better unlearning performance within a limited number of fine-tuning steps by optimizing the sampling strategy to focus more on the regions around samples in the unlearning set A_u instead of the remaining set A_r. More importantly, different from some *biased* loss function terms employed in previous work like gradient ascent term in $\mathcal{L}_{\text{NegGrad}}$ and $\mathcal{L}_{\text{SISS}}$ or pure noise guidance term in $\mathcal{L}_{\text{EraseDiff}}$, $\mathcal{L}_{\text{unlearn}}$ is an unbiased loss. This property is crucial for preserving model stability and avoiding the quality degradation commonly observed with biased methods.

3.3 Tractable Approximation with Intuitive Interpretation

Although theoretically valid, directly evaluating $\mathcal{L}_{\text{unlearn}}$ can be time-consuming. For a given timestep t, a datum $\boldsymbol{a}_u$ sampled from the unlearning set A_u, and the corresponding noisy sample $\boldsymbol{x}_t \sim q_t(\cdot|\boldsymbol{a}_u)$, it requires computing a weighted average over the entire remaining set A_r with weights $w_t(\boldsymbol{x}_t|\boldsymbol{a}_r) \propto q_t(\boldsymbol{x}_t|\boldsymbol{a}_r)$. When the size of A_r is large, this becomes computationally expensive, which motivates approximation.

Note that

$$q_t(\boldsymbol{x}_t|\boldsymbol{a}_r) \propto \exp\left(-\frac{\|\boldsymbol{x}_t-\gamma_t\boldsymbol{a}_r\|_2^2}{2\sigma_t^2}\right),$$

therefore, as $\gamma_t\boldsymbol{a}_r$ moves away from $\boldsymbol{x}_t$, the likelihoods $q_t(\boldsymbol{x}_t|\boldsymbol{a}_r)$ decay exponentially with the squared Euclidean distance $\|\boldsymbol{x}_t-\gamma_t\boldsymbol{a}_r\|_2^2$. Consequently, in the weighted average most terms receive nearly negligible weights, and the value is determined primarily by a small subset with the largest weights. It is therefore natural to truncate and estimate the full weighted average by computing it only over the k items with the largest weights.

In fact, the k elements in A_r that maximize the weights on average can be identified by the following proposition:

Proposition 2. *Given $\boldsymbol{a}_u$ sampled from the unlearning set A_u, the k samples in the remaining set A_r with the largest expected likelihoods*

$$\mathbb{E}_{t,\boldsymbol{x}_t\sim q_t(\cdot|\boldsymbol{a}_u)}\left[q_t(\boldsymbol{x}_t|\boldsymbol{a}_r)\right]$$

are the k-nearest neighbors of $\boldsymbol{a}_u$ under the Euclidean metric.

The proof is provided in the appendix. Based on this proposition, we truncate the summation in the original $\mathcal{L}_{\text{unlearn}}$ and approximate the full sum using only the largest k terms, which yields the following approximate fine-tuning loss

$$\mathcal{L}_{\text{unlearn}}(\boldsymbol{\theta}; k) = \mathbb{E}_{t,\boldsymbol{a}_u\sim A_u,\boldsymbol{x}_t\sim q_t(\cdot|\boldsymbol{a}_u)} \left[\sum_{\boldsymbol{a}_r\in\mathcal{S}_k(\boldsymbol{a}_u)} \tilde{w}_t(\boldsymbol{x}_t|\boldsymbol{a}_r) \left\| \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \frac{\boldsymbol{x}_t - \gamma_t \boldsymbol{a}_r}{\sigma_t} \right\|_2^2 \right],$$

where the truncated weights are defined as

$$\tilde{w}_t(\boldsymbol{x}_t|\boldsymbol{a}_r) = \frac{q_t(\boldsymbol{x}_t|\boldsymbol{a}_r)}{\sum_{\boldsymbol{a}'_r\in\mathcal{S}_k(\boldsymbol{a}_u)} q_t(\boldsymbol{x}_t|\boldsymbol{a}'_r)},$$

and $\mathcal{S}_k(\boldsymbol{a}_u)$ denotes the set of $\boldsymbol{a}_u$'s k-nearest neighbors in A_r. This can be viewed as computing a weighted average only over the k-nearest neighbors of $\boldsymbol{a}_u$ using the truncated weights $\tilde{w}_t(\boldsymbol{x}_t|\boldsymbol{a}_r)$. Owing to the exponential decay of the likelihood, in practice the choice of k can be much smaller than the size of A_r, which substantially simplifies the evaluation of the loss function.

We provide an intuitive analysis of the approximate fine-tuning loss function. During fine-tuning, our method can be viewed as adding noise to $\boldsymbol{a}_u$ to obtain the noisy sample $\boldsymbol{x}_t$, and then training the model to denoise it toward $\boldsymbol{a}_r \in \mathcal{S}_k(\boldsymbol{a}_u)$ as if $\boldsymbol{x}_t$ had been produced by adding noise to $\boldsymbol{a}_r$. For a pretrained model, since information about $\boldsymbol{a}_u$ is already encoded internally, denoising the $\boldsymbol{x}_t$ obtained from $\boldsymbol{a}_u$ will typically drive the denoising trajectory $\boldsymbol{x}_t, \boldsymbol{x}_{t-1}, \cdots, \boldsymbol{x}_0$ back toward $\boldsymbol{a}_u$, especially when the noise level is small. However, this fine-tuning loss forces the model to redirect the denoising trajectory that would converge to $\boldsymbol{a}_u$ toward its nearest neighbors in the remaining set A_r, thereby preventing the fine-tuned model from generating undesirable results and achieving efficient unlearning. Figure 1 provides an intuitive illustration of the redirection. Because $\boldsymbol{a}_r \in \mathcal{S}_k(\boldsymbol{a}_u)$ is close to $\boldsymbol{a}_u$, the correction required in the model output is small, which accelerates the fine-tuning procedure and improves training stability. Moreover, by steering the denoising trajectory toward other real data, the method maintains reasonable outputs during unlearning and thus better mitigates degradation in generation quality. This preservation of realism is consistent with the unbiasedness mentioned in our earlier theoretical analysis.

3.4 Final Objective

Finally, due to the finite capacity of practical networks, fine-tuning only in regions around the unlearning set A_u can cause catastrophic forgetting elsewhere. To mitigate this issue, we introduce the vanilla fine-tuning loss as a regularizer to avoid over-forgetting on the remaining set A_r. The final loss of our ReTrack method is formulated as an interpolation between the unlearning term and the regularization term:

$$\mathcal{L}_{\text{ReTrack}}(\boldsymbol{\theta}) = \lambda\, \mathcal{L}_{\text{unlearn}}(\boldsymbol{\theta}; k) + (1-\lambda)\, \mathcal{L}_{\text{vanilla}}(\boldsymbol{\theta}),$$

where $\lambda \in [0, 1]$ is a hyperparameter that controls the strength of unlearning. The complete fine-tuning process, implemented with stochastic gradient descent, is shown in Algorithm 1. Since the unlearning set A_u and the remaining set A_r are fixed throughout fine-tuning, the k-nearest neighbors can be computed once in advance and stored, and there is no need to recompute at each iteration. Moreover, although the unlearning term sums over k neighbors, these terms share the same network prediction $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$, and thus require only a single function evaluation. As a result, the overall computational cost of our method remains comparable to that of prior methods.

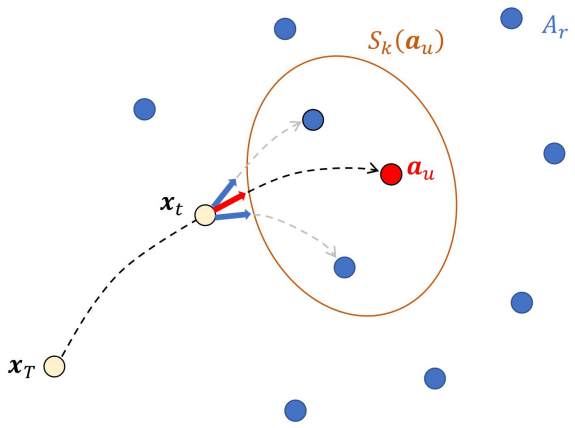

Figure 1: An intuitive schematic of our method. By changing the diffusion model's noise prediction on $\boldsymbol{x}_t$, the denoising trajectory that originally pointed to the unlearning sample $\boldsymbol{a}_u$ is redirected to its k-nearest neighbors, thus achieving fast unlearning while preserving generation quality.

4 EXPERIMENTS

4.1 Experimental Setups

4.1.1 Datasets.

We conduct experiments on four datasets: MNIST T-Shirt, CelebA-HQ, CIFAR-10, and Stable Diffusion. The MNIST T-Shirt dataset is a relatively simple test case constructed by Alberti et al. (2025) by augmenting the 60,000 handwritten digit images of the

Algorithm 1 The fine-tuning process of ReTrack.

Input: Pretrained model $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$, unlearning set A_u, remaining set A_r, fine-tuning steps N.
Output: Unlearned model $\boldsymbol{\epsilon}_{\boldsymbol{\theta}^*}$.

For each $\boldsymbol{a}_u$ in A_u, find its k-nearest neighbors in A_r under the Euclidean metric.
for $n = 1$ **to** N **do**
  Sample a timestep $t \in \{1, 2, \cdots, T\}$.
  Sample $\boldsymbol{a}_u$ from A_u, $\boldsymbol{x}_t \sim q_t(\cdot|\boldsymbol{a}_u)$ and compute $\mathcal{L}_1(\boldsymbol{\theta}) = \sum_{\boldsymbol{a}_r \in \mathcal{S}_k(\boldsymbol{a}_u)} \tilde{w}_t(\boldsymbol{x}_t|\boldsymbol{a}_r) \left\| \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \frac{\boldsymbol{x}_t - \gamma_t \boldsymbol{a}_r}{\sigma_t} \right\|_2^2$.
  Sample $\boldsymbol{a}_r$ from A_r, $\boldsymbol{x}_t \sim q_t(\cdot|\boldsymbol{a}_r)$ and compute $\mathcal{L}_2(\boldsymbol{\theta}) = \|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{\epsilon}\|_2^2$.
  Compute $\mathcal{L}(\boldsymbol{\theta}) = \lambda \mathcal{L}_1(\boldsymbol{\theta}) + (1 - \lambda)\mathcal{L}_2(\boldsymbol{\theta})$.
  Take gradient descent step on $\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta})$.
end for

MNIST dataset (LeCun et al., 1998) with T-Shirt images drawn from Fashion-MNIST dataset (Xiao, Rasul, and Vollgraf, 2017) at a 1% ratio, and the goal of unlearning is to make the model forget the T-Shirt images. For CelebA-HQ dataset (Karras et al., 2017), we use the 256×256 resolution version that comprises 30,000 high quality facial images. For CIFAR-10 (Krizhevsky, Hinton et al., 2009), we use the training split of 50,000 images. The unlearning dataset for the Stable Diffusion is also constructed by Alberti et al. (2025). By fixing the model's text-conditioned input, Stable Diffusion can be treated as an unconditional generative model for a specific concept, and thus can be employed to evaluate data unlearning methods. For each text-image pair appearing in the training dataset of Stable Diffusion, unlearning set and remaining set are required in order to perform data unlearning methods so that the model forgets the original image in the training dataset in this text condition. Therefore, 128 images are sampled using the prompt and then clustered. Those images sufficiently similar to the original training images are designated as the unlearning set, while the remainder formed the remaining set, as demonstrated in Figure 2.

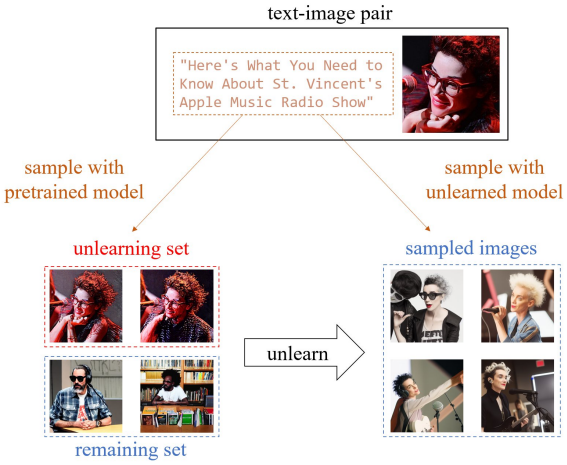

Figure 2: Construction of the Stable Diffusion dataset. For each text-image pair, in order to forget the corresponding image in this text condition, the pretrained model is used to sample images in the given text condition and clustered to construct the unlearning set and remaining set required by the data unlearning method. After unlearning, the model no longer generates the corresponding image in the same text condition.

4.1.2 Evaluation metrics.

To measure whether an unlearning method can effectively forget images in the unlearning set while preserving generation quality, we evaluate each method using both quality metrics and unlearning metrics.

For quality metrics, we use the standard FID (Heusel et al., 2017), IS (Salimans et al., 2016), and CLIP-IQA (Wang, Chan, and Loy, 2023) metrics to measure generation quality of the unlearned model.

For unlearning metrics, we use Frequency, NLL, and SSCD (Pizzi et al., 2022) to quantify the extent to which the model memorizes a given image. The details are as follows: for the MNIST T-Shirt dataset, because it is trivial to distinguish whether images sampled from the unlearned model belong to T-shirts or handwritten digits, we directly assess unlearning performance by calculating the frequency of T-Shirt in the generated images. Following the procedure in Song et al. (2020), we calculate the negative log-likelihood (NLL) to quantify the model's memorization strength on the specific images. For CelebA-HQ and CIFAR-10, following Alberti et al. (2025), instead of generating images from the random Gaussian noise, we inject t steps of noise into the clean image and then use the unlearned model to denoise and reconstruct the clean image. We then compute SSCD between the original training image and the reconstructed image to measure the memorization strength of the unlearned model for this image, as shown in Figure 3. For the Stable Diffusion dataset, due to the limited number of images sampled during each evaluation, we directly calculate the average SSCD between the images sampled by the unlearned model and the original training image to evaluate the unlearning performance.

4.1.3 Implementation details.

For the MNIST T-Shirt dataset constructed by Alberti et al. (2025), we directly employ the pretrained model provided by them. The pretrained models for CelebA-HQ and CIFAR-10 are adopted from Ho, Jain, and Abbeel (2020). For Stable Diffusion, we use v1.4

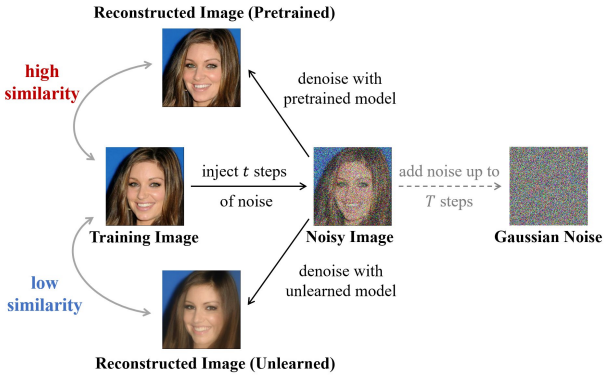

Figure 3: Schematic of the SSCD metric calculation method on CelebA-HQ and CIFAR-10. Instead of adding noise up to the full T steps and generating image from the pure Gaussian noise, we inject noise for only $t < T$ steps and use the diffusion model to denoise and reconstruct the clean image. The SSCD is then calculated to measure the similarity between the original training image and the reconstructed image, thereby indicating how much of the training image's information is retained by the model.

for experiments. The noise injection intensity for SSCD calculation is set to $t = T/4$, consistent with Alberti et al. (2025). In our experiments, unless otherwise stated, we set the hyperparameter k in our method to 10 and adjust λ in each dataset so that the unlearning term and the regularization term are of similar order of magnitude. The hyperparameters for the other methods are set according to their original works. Since Stable Diffusion v1.4 is a latent diffusion model that performs the diffusion process in the latent space, in this experiment our method searches for k-nearest neighbors in the latent space to ensure consistency with the theoretical analysis. For all other experiments with non-latent diffusion models, the k-nearest neighbors are defined in the pixel domain.

4.2 Main Results

This section presents the metric evaluations of each method on each dataset. For clarity, method names in the tables are formatted as follows: *italic* for pretrained models, **bold** for methods that preserve generation quality, and plain font for methods that severely degrade generation quality. The best result among the quality-preserving methods is also highlighted in **bold**.

4.2.1 MNIST T-Shirt.

We conduct experiments under 10 different random seeds and report both the mean and standard deviation of the performance. After fine-tuning for 50 steps, we sample 50,000 images and evaluate the extent of unlearning achieved by each method. The results are presented in Table 1. Among all methods that preserve generation quality, our method achieves the strongest unlearning performance.

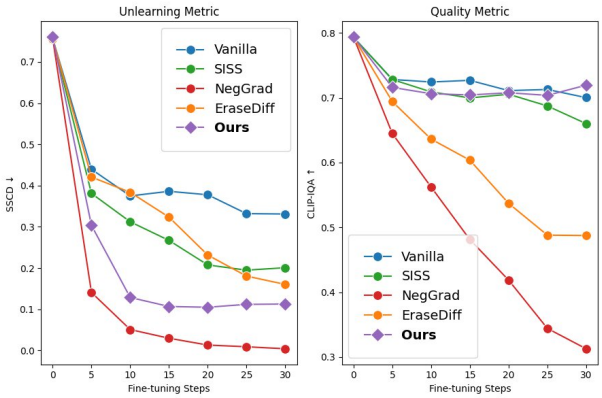

Figure 4: Results on Stable Diffusion.

4.2.2 CelebA-HQ.

We randomly select 10 facial images from the CelebA-HQ dataset, and separately unlearn those chosen images. After fine-tuning for 40 steps, 10,000 images are sampled to evaluate the generation quality. The average unlearning performance and standard deviations are shown in Table 2. The experimental results show that, among all methods that preserve generation quality, our method yields the best evaluation metrics.

4.2.3 CIFAR-10.

We randomly choose 10 images from the CIFAR-10 dataset and separately conduct experiments on them. We sample 10,000 images for generation quality evaluation after 60 fine-tuning steps. The average results are reported in Table 3. The results indicate that our method outperforms all other methods that preserve generation quality.

4.2.4 Stable Diffusion.

We conducted experiments independently on 48 text-image pairs. For each experiment, a total of 30 steps of fine-tuning are performed, with an evaluation every 5 steps. For each evaluation, 16 images are sampled to compute the generation quality metric CLIP-IQA and the unlearning metric SSCD. The averaged results are shown in Figure 4. The results show that the unlearning metric of our method is only slightly lower than that of NegGrad method, which causes serious degradation of the generation quality, implying that our method is the best among all quality-preserving methods.

Table 1: Results on MNIST T-Shirt

Method	Unlearning		Quality	
	Frequency↓	**NLL↑**	**FID↓**	**IS↑**
Pretrained	0.8252% ± 0.0010%	0.81 ± 0.05	1.17 ± 0.00	9.67 ± 0.00
NegGrad	0.0000% ± 0.0000%	14.84 ± 1.04	276.65 ± 14.35	6.60 ± 0.33
EraseDiff	0.0004% ± 0.0012%	8.77 ± 0.29	6.92 ± 5.48	9.14 ± 0.36
Vanilla	0.3598% ± 0.2933%	1.14 ± 0.09	2.16 ± 0.72	**9.52** ± 0.09
SISS	0.0530% ± 0.0747%	4.53 ± 0.50	3.05 ± 1.58	9.37 ± 0.24
Ours	**0.0000%** ± 0.0000%	**8.28** ± 0.03	**2.09** ± 1.04	9.48 ± 0.06

Table 2: Results on CelebA-HQ

Method	Unlearning		Quality
	NLL↑	**SSCD↓**	**FID↓**
Pretrained	1.29 ± 0.13	0.88 ± 0.02	17.99 ± 0.00
NegGrad	6.93 ± 1.12	0.14 ± 0.07	424.62 ± 47.04
EraseDiff	2.59 ± 0.26	0.33 ± 0.05	113.15 ± 19.32
Vanilla	1.28 ± 0.13	0.89 ± 0.03	22.20 ± 2.24
SISS	1.44 ± 0.27	0.50 ± 0.11	22.02 ± 1.64
Ours	**1.51** ± 0.25	**0.41** ± 0.16	**21.26** ± 2.59

Table 3: Results on CIFAR-10

Method	Unlearning		Quality	
	NLL↑	**SSCD↓**	**FID↓**	**IS↑**
Pretrained	3.22 ± 0.36	0.54 ± 0.06	5.27 ± 0.00	9.29 ± 0.00
NegGrad	4.33 ± 0.35	0.23 ± 0.06	81.02 ± 33.19	6.32 ± 1.59
EraseDiff	3.55 ± 0.30	0.48 ± 0.09	29.22 ± 8.68	8.32 ± 0.45
Vanilla	3.22 ± 0.36	0.55 ± 0.08	8.77 ± 0.00	9.14 ± 0.00
SISS	3.21 ± 0.37	0.45 ± 0.06	9.05 ± 0.64	9.29 ± 0.13
Ours	**3.44** ± 0.32	**0.37** ± 0.05	**8.45** ± 0.42	**9.42** ± 0.14

4.3 Ablation Study

In the ablation study, we discuss two important parts in our method: the choice of hyperparameter k and the effect of the regularization term. All experiments conducted in this section are performed on the MNIST T-Shirt dataset.

4.3.1 Choice of hyperparameter k.

We keep all other experimental settings fixed and vary the hyperparameter k in our method. The corresponding results are presented in Table 4. As k increases, the unlearning metrics remains essentially unchanged, while the quality metrics exhibits a slight improvement. Therefore, to strike a balance between the performance of our method and the implementation simplicity of the loss function, we recommend setting $k = 10$ based on the empirical observations.

4.3.2 Effect of the regularization term.

To demonstrate the necessity of incorporating the regularization term in the loss function of our method, we hold all other experimental settings fixed and compare the performance of two variants of our loss function: one including both the unlearning term and the regularization term, and the other retaining only the unlearning term. The results, summarized in Table 5, reveal that although omitting the regularization term yields a marginal improvement in the unlearning metrics, it also causes a pronounced degradation in generation quality. This deterioration indicates that the absence of the regularization term induces over-forgetting, thereby validating the crucial role of the regularization term in preserving the model's genera-

Table 4: Ablation study on the choice of hyperparameter k

k	Unlearning		Quality	
	Frequency↓	**NLL↑**	**FID↓**	**IS↑**
1	0.0000% ± 0.0000%	8.29 ± 0.04	2.22 ± 1.02	9.48 ± 0.06
10	0.0000% ± 0.0000%	8.28 ± 0.03	2.09 ± 1.04	9.48 ± 0.06
100	0.0000% ± 0.0000%	8.28 ± 0.04	2.04 ± 1.02	9.48 ± 0.06
1000	0.0000% ± 0.0000%	8.28 ± 0.04	2.01 ± 1.05	9.49 ± 0.07
Pretrained	0.8252% ± 0.0010%	0.81 ± 0.05	1.17 ± 0.00	9.67 ± 0.00

Table 5: Ablation study on the effect of the regularization term.

Setting	Unlearning		Quality	
	Frequency↓	**NLL↑**	**FID↓**	**IS↑**
w/ reg.	0.0000% ± 0.0000%	8.28 ± 0.03	2.09 ± 1.04	9.48 ± 0.06
w/o reg.	0.0000% ± 0.0000%	9.74 ± 0.09	14.22 ± 3.81	9.40 ± 0.08
Pretrained	0.8252% ± 0.0010%	0.81 ± 0.05	1.17 ± 0.00	9.67 ± 0.00

tive quality.

5 CONCLUSION

In this paper, we propose ReTrack, a novel data unlearning method for diffusion models that redirects the denoising trajectories toward the k-nearest neighbors. Using importance sampling, we first derive an unbiased loss that concentrates the fine-tuning procedure on the regions surrounding the unlearning set. Then, we approximate this loss by keeping only the k largest terms, enabling fast unlearning while preserving generation quality. Across MNIST T-Shirt, CelebA-HQ, CIFAR-10, and Stable Diffusion, ReTrack achieves the best trade-off between unlearning strength and generation quality compared with prior methods.

Acknowledgements

The authors are with the Department of Electronic Engineering, Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing 100084, China. This work was supported by the National Key Research and Development Program of China (Grant No. 2025YFF0515601) and the National Natural Science Foundation of China (NSAF U2230201).

References

Alberti, S.; Hasanaliyev, K.; Shah, M.; and Ermon, S. 2025. Data unlearning in diffusion models. *arXiv preprint arXiv:2503.01034*.

Alimisis, P.; Mademlis, I.; Radoglou-Grammatikis, P.; Sarigiannidis, P.; and Papadopoulos, G. T. 2025. Advances in diffusion models for image data augmentation: A review of methods, models, evaluation metrics and future research directions. *Artificial Intelligence Review*, 58(4): 1–55.

Bourtoule, L.; Chandrasekaran, V.; Choquette-Choo, C. A.; Jia, H.; Travers, A.; Zhang, B.; Lie, D.; and Papernot, N. 2021. Machine unlearning. In *2021 IEEE symposium on security and privacy (SP)*, 141–159. IEEE.

Carlini, N.; Hayes, J.; Nasr, M.; Jagielski, M.; Sehwag, V.; Tramer, F.; Balle, B.; Ippolito, D.; and Wallace, E. 2023. Extracting training data from diffusion models. In *32nd USENIX Security Symposium (USENIX Security 23)*, 5253–5270.

Chen, Y.; Ma, X.; Zou, D.; and Jiang, Y.-G. 2024. Towards a theoretical understanding of memorization in diffusion models. *arXiv preprint arXiv:2410.02467*.

Fan, C.; Liu, J.; Zhang, Y.; Wong, E.; Wei, D.; and Liu, S. 2023. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. *arXiv preprint arXiv:2310.12508*.

Gandikota, R.; Materzynska, J.; Fiotto-Kaufman, J.; and Bau, D. 2023. Erasing concepts from diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2426–2436.

Gandikota, R.; Orgad, H.; Belinkov, Y.; Materzyńska, J.; and Bau, D. 2024. Unified concept editing in diffusion models. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 5111–5120.

Golatkar, A.; Achille, A.; and Soatto, S. 2020. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 9304–9312.

Graves, L.; Nagisetty, V.; and Ganesh, V. 2021. Amnesiac machine learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 11516–11524.

Gu, X.; Du, C.; Pang, T.; Li, C.; Lin, M.; and Wang, Y. 2023. On memorization in diffusion models. *arXiv preprint arXiv:2310.02664*.

Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; and Hochreiter, S. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30.

Ho, J.; Jain, A.; and Abbeel, P. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33: 6840–6851.

Izzo, Z.; Smart, M. A.; Chaudhuri, K.; and Zou, J. 2021. Approximate data deletion from machine learning models. In *International Conference on Artificial Intelligence and Statistics*, 2008–2016. PMLR.

Karras, T.; Aila, T.; Laine, S.; and Lehtinen, J. 2017. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324.

Li, H.; Yang, Y.; Chang, M.; Chen, S.; Feng, H.; Xu, Z.; Li, Q.; and Chen, Y. 2022. Srdiff: Single image super-resolution with diffusion probabilistic models. *Neurocomputing*, 479: 47–59.

Loshchilov, I.; and Hutter, F. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Lugmayr, A.; Danelljan, M.; Romero, A.; Yu, F.; Timofte, R.; and Van Gool, L. 2022. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11461–11471.

Manukyan, H.; Sargsyan, A.; Atanyan, B.; Wang, Z.; Navasardyan, S.; and Shi, H. 2023. Hd-painter: High-resolution and prompt-faithful text-guided image inpainting with diffusion models. In *The Thirteenth International Conference on Learning Representations*.

Panda, S.; Varun, M.; Jain, S.; Maharana, S. K.; and Prathosh, A. 2024. Variational Diffusion Unlearning: a variational inference framework for unlearning in diffusion models. In *Neurips Safe Generative AI Workshop 2024*.

Perifanis, V.; Karypidis, E.; Komodakis, N.; and Efraimidis, P. 2024. SFTC: Machine Unlearning via Selective Fine-tuning and Targeted Confusion. In *Proceedings of the 2024 European Interdisciplinary Cybersecurity Conference*, 29–36.

Pizzi, E.; Roy, S. D.; Ravindra, S. N.; Goyal, P.; and Douze, M. 2022. A self-supervised descriptor for image copy detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14532–14542.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10684–10695.

Saharia, C.; Ho, J.; Chan, W.; Salimans, T.; Fleet, D. J.; and Norouzi, M. 2022. Image super-resolution via iterative refinement. *IEEE transactions on pattern analysis and machine intelligence*, 45(4): 4713–4726.

Salimans, T.; Goodfellow, I.; Zaremba, W.; Cheung, V.; Radford, A.; and Chen, X. 2016. Improved techniques for training gans. *Advances in neural information processing systems*, 29.

Somepalli, G.; Singla, V.; Goldblum, M.; Geiping, J.; and Goldstein, T. 2023. Diffusion art or digital forgery? investigating data replication in diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 6048–6058.

Song, Y.; Sohl-Dickstein, J.; Kingma, D. P.; Kumar, A.; Ermon, S.; and Poole, B. 2020. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*.

Wang, J.; Chan, K. C.; and Loy, C. C. 2023. Exploring clip for assessing the look and feel of images. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 2555–2563.

Wang, J.; Guo, S.; Xie, X.; and Qi, H. 2022. Federated unlearning via class-discriminative pruning. In *Proceedings of the ACM web conference 2022*, 622–632.

Wang, W.; Tian, Z.; Zhang, C.; and Yu, S. 2024. Machine unlearning: A comprehensive survey. *arXiv preprint arXiv:2405.07406*.

Wu, J.; Le, T.; Hayat, M.; and Harandi, M. 2025. Erasing undesirable influence in diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 28263–28273.

Wu, X. 2022. Creative painting with latent diffusion models. *arXiv preprint arXiv:2209.14697*.

Wu, Y.; Dobriban, E.; and Davidson, S. 2020. Deltagrad: Rapid retraining of machine learning models. In *International Conference on Machine Learning*, 10355–10366. PMLR.

Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

Xu, J.; Wu, Z.; Wang, C.; and Jia, X. 2024. Machine unlearning: Solutions and challenges. *IEEE Transactions on Emerging Topics in Computational Intelligence*.

Zhang, G.; Wang, K.; Xu, X.; Wang, Z.; and Shi, H. 2024. Forget-me-not: Learning to forget in text-to-image diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 1755–1764.

Zhang, Y.; Huang, N.; Tang, F.; Huang, H.; Ma, C.; Dong, W.; and Xu, C. 2023. Inversion-based style transfer with diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10146–10156.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Yes]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Supplementary Materials

A PROOFS OF PROPOSITIONS

In this section, we provide the detailed proofs of the propositions stated in the methodology section of the main text.

A.1 Proof of Proposition 1

Proposition 1. *The two fine-tuning loss functions $\mathcal{L}_{\text{vanilla}}$ and $\mathcal{L}_{\text{unlearn}}$ are equivalent.*

Proof. Denoting the discrete uniform distributions on A_r and A_u as

$$q_{A_r}(\boldsymbol{a}) = \frac{1}{|A_r|}\,\mathbf{1}_{A_r}(\boldsymbol{a}),$$
$$q_{A_u}(\boldsymbol{a}) = \frac{1}{|A_u|}\,\mathbf{1}_{A_u}(\boldsymbol{a}),$$

where $|\cdot|$ denotes the cardinality of a finite set, and the indicator function is defined as

$$\mathbf{1}_A(\boldsymbol{a}) = \begin{cases} 1, & \boldsymbol{a} \in A, \\ 0, & \boldsymbol{a} \notin A. \end{cases}$$

Expanding the expectations over $\boldsymbol{a}_r$ and $\boldsymbol{x}_t$ in $\mathcal{L}_{\text{vanilla}}$, we have

$$\begin{aligned}
\mathcal{L}_{\text{vanilla}}(\boldsymbol{\theta}) &= \mathbb{E}_{t,\boldsymbol{a}_r\sim A_r,\boldsymbol{x}_t\sim q_t(\cdot|\boldsymbol{a}_r)}\left[\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)-\boldsymbol{\epsilon}\|_2^2\right] \\
&= \mathbb{E}_t\left[\sum_{\boldsymbol{a}_r\in A_r}\int_{\mathbb{R}^d} q_{A_r}(\boldsymbol{a}_r)q_t(\boldsymbol{x}_t|\boldsymbol{a}_r)\left\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)-\frac{\boldsymbol{x}_t-\gamma_t\boldsymbol{a}_r}{\sigma_t}\right\|_2^2 \mathrm{d}\boldsymbol{x}_t\right] \\
&= \mathbb{E}_t\left[\sum_{\boldsymbol{a}_r\in A_r}\int_{\mathbb{R}^d} q_t(\boldsymbol{x}_t)\frac{q_{A_r}(\boldsymbol{a}_r)q_t(\boldsymbol{x}_t|\boldsymbol{a}_r)}{q_t(\boldsymbol{x}_t)}\left\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)-\frac{\boldsymbol{x}_t-\gamma_t\boldsymbol{a}_r}{\sigma_t}\right\|_2^2 \mathrm{d}\boldsymbol{x}_t\right].
\end{aligned}$$

By defining the weights as

$$\begin{aligned}
w_t(\boldsymbol{x}_t|\boldsymbol{a}_r) &= \frac{q_{A_r}(\boldsymbol{a}_r)q_t(\boldsymbol{x}_t|\boldsymbol{a}_r)}{q_t(\boldsymbol{x}_t)} \\
&= \frac{q_{A_r}(\boldsymbol{a}_r)q_t(\boldsymbol{x}_t|\boldsymbol{a}_r)}{\sum_{\boldsymbol{a}'_r\in A_r} q_{A_r}(\boldsymbol{a}'_r)q_t(\boldsymbol{x}_t|\boldsymbol{a}'_r)} \\
&= \frac{q_t(\boldsymbol{x}_t|\boldsymbol{a}_r)}{\sum_{\boldsymbol{a}'_r\in A_r} q_t(\boldsymbol{x}_t|\boldsymbol{a}'_r)}
\end{aligned}$$

and expanding the first $q_t(\boldsymbol{x}_t)$ in the integral using the law of total probability

$$q_t(\boldsymbol{x}_t) = \sum_{\boldsymbol{a}_u\in A_u} q_{A_u}(\boldsymbol{a}_u)q_t(\boldsymbol{x}_t|\boldsymbol{a}_u),$$

we obtain

$$\begin{aligned}
\mathcal{L}_{\text{vanilla}}(\boldsymbol{\theta}) &= \mathbb{E}_t \left[\sum_{\boldsymbol{a}_r \in A_r} \int_{\mathbb{R}^d} \sum_{\boldsymbol{a}_u \in A_u} q_{A_u}(\boldsymbol{a}_u) q_t(\boldsymbol{x}_t|\boldsymbol{a}_u) w_t(\boldsymbol{x}_t|\boldsymbol{a}_r) \left\| \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \frac{\boldsymbol{x}_t - \gamma_t \boldsymbol{a}_r}{\sigma_t} \right\|_2^2 \mathrm{d}\boldsymbol{x}_t \right] \\
&= \mathbb{E}_t \left[\sum_{\boldsymbol{a}_u \in A_u} \int_{\mathbb{R}^d} q_{A_u}(\boldsymbol{a}_u) q_t(\boldsymbol{x}_t|\boldsymbol{a}_u) \sum_{\boldsymbol{a}_r \in A_r} w_t(\boldsymbol{x}_t|\boldsymbol{a}_r) \left\| \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \frac{\boldsymbol{x}_t - \gamma_t \boldsymbol{a}_r}{\sigma_t} \right\|_2^2 \mathrm{d}\boldsymbol{x}_t \right] \\
&= \mathbb{E}_{t, \boldsymbol{a}_u \sim A_u, \boldsymbol{x}_t \sim q_t(\cdot|\boldsymbol{a}_u)} \left[\sum_{\boldsymbol{a}_r \in A_r} w_t(\boldsymbol{x}_t|\boldsymbol{a}_r) \left\| \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \frac{\boldsymbol{x}_t - \gamma_t \boldsymbol{a}_r}{\sigma_t} \right\|_2^2 \right] \\
&= \mathcal{L}_{\text{unlearn}}(\boldsymbol{\theta}).
\end{aligned}$$

□

A.2 Proof of Proposition 2

Proposition 2. *Given $\boldsymbol{a}_u$ sampled from the unlearning set A_u, the k samples in the remaining set A_r with the largest expected likelihoods*

$$\mathbb{E}_{t, \boldsymbol{x}_t \sim q_t(\cdot|\boldsymbol{a}_u)} \left[q_t(\boldsymbol{x}_t|\boldsymbol{a}_r) \right]$$

are the k-nearest neighbors of $\boldsymbol{a}_u$ under the Euclidean metric.

Proof. Define the auxiliary variable

$$\boldsymbol{z}_t = \frac{\boldsymbol{x}_t - \gamma_t \boldsymbol{a}_r}{\sigma_t}.$$

Since $\boldsymbol{x}_t \sim q_t(\cdot|\boldsymbol{a}_u)$, we have

$$\boldsymbol{z}_t \sim \mathcal{N}(\cdot; \boldsymbol{\mu}_t, \mathbf{I}_d),$$

where the expectation of the Gaussian distribution is

$$\boldsymbol{\mu}_t = \frac{\gamma_t}{\sigma_t}(\boldsymbol{a}_u - \boldsymbol{a}_r).$$

Then, the expected values of the likelihoods are given by

$$\begin{aligned}
\mathbb{E}_{t, \boldsymbol{x}_t \sim q_t(\cdot|\boldsymbol{a}_u)} \left[q_t(\boldsymbol{x}_t|\boldsymbol{a}_r) \right] =& \mathbb{E}_{t, \boldsymbol{x}_t \sim q_t(\cdot|\boldsymbol{a}_u)} \left[\frac{1}{(2\pi\sigma_t^2)^{\frac{d}{2}}} \exp\left(-\frac{\|\boldsymbol{x}_t - \gamma_t \boldsymbol{a}_r\|_2^2}{2\sigma_t^2} \right) \right] \\
=& \mathbb{E}_{t, \boldsymbol{z}_t \sim \mathcal{N}(\cdot; \boldsymbol{\mu}_t, \mathbf{I}_d)} \left[\frac{1}{(2\pi\sigma_t^2)^{\frac{d}{2}}} \exp\left(-\frac{1}{2}\|\boldsymbol{z}_t\|_2^2 \right) \right] \\
\propto& \sum_{t=1}^{T} \sigma_t^{-d} \int_{\mathbb{R}^d} \exp\left(-\frac{1}{2}\|\boldsymbol{z}_t - \boldsymbol{\mu}_t\|_2^2 \right) \exp\left(-\frac{1}{2}\|\boldsymbol{z}_t\|_2^2 \right) \mathrm{d}\boldsymbol{z}_t \\
=& \sum_{t=1}^{T} \sigma_t^{-d} \int_{\mathbb{R}^d} \exp\left(-\|\boldsymbol{z}_t\|_2^2 + \boldsymbol{\mu}_t^\top \boldsymbol{z}_t - \frac{1}{2}\|\boldsymbol{\mu}_t\|_2^2 \right) \mathrm{d}\boldsymbol{z}_t \\
=& \sum_{t=1}^{T} \sigma_t^{-d} \exp\left(-\frac{1}{4}\|\boldsymbol{\mu}_t\|_2^2 \right) \int_{\mathbb{R}^d} \exp\left(-\left\| \boldsymbol{z}_t - \frac{\boldsymbol{\mu}_t}{2} \right\|_2^2 \right) \mathrm{d}\boldsymbol{z}_t \\
\propto& \sum_{t=1}^{T} \sigma_t^{-d} \exp\left(-\frac{1}{4}\|\boldsymbol{\mu}_t\|_2^2 \right) \\
=& \sum_{t=1}^{T} \sigma_t^{-d} \exp\left(-\frac{\gamma_t^2}{4\sigma_t^2}\|\boldsymbol{a}_u - \boldsymbol{a}_r\|_2^2 \right),
\end{aligned}$$

so maximizing the likelihoods $q_t(\boldsymbol{x}_t|\boldsymbol{a}_r)$ in expectation is equivalent to minimizing the Euclidean distance $\|\boldsymbol{a}_u - \boldsymbol{a}_r\|_2$. Consequently, the k samples in A_r with the largest expected likelihoods are the k-nearest neighbors of $\boldsymbol{a}_u$ under the Euclidean metric. □

B COMPARISON WITH PRIOR METHOD

While our proposed ReTrack and the recent SISS (Alberti et al., 2025) both utilize importance sampling for data unlearning in diffusion models, they differ fundamentally in their theoretical formulations and the intrinsic role of importance sampling.

According to the discussion in Appendix A.1 (Stability Analysis and Interpretation of SISS) of the SISS paper, their unlearning objective is equivalent to a vanilla loss and a heavily biased gradient-ascent term:

$$\mathcal{L}_{\text{SISS}}(\boldsymbol{\theta}) = \mathcal{L}_{\text{vanilla}}(\boldsymbol{\theta}) + s\mathcal{L}_{\text{NegGrad}}(\boldsymbol{\theta})$$

where s is a coefficient chosen to dynamically balance the gradient norms. In their method, importance sampling acts primarily as a computational trick to unify the forward-pass computation of these two distinct loss terms, effectively halving the number of function evaluations. As implied by their own ablation studies, importance sampling is not the core driver of unlearning in SISS; the method fundamentally relies on the biased $\mathcal{L}_{\text{NegGrad}}$ component.

In contrast, our unlearning objective is theoretically unbiased and is formulated as a combination of an unlearning term and a regularization term:

$$\mathcal{L}_{\text{ReTrack}}(\boldsymbol{\theta}) = \lambda\mathcal{L}_{\text{unlearn}}(\boldsymbol{\theta}) + (1-\lambda)\mathcal{L}_{\text{vanilla}}(\boldsymbol{\theta})$$

We identify that while the original vanilla loss perfectly preserves generation quality due to its unbiasedness, it suffers from extremely low sampling efficiency for unlearning. Consequently, ReTrack employs importance sampling specifically to shift the sampling distribution of the vanilla loss toward the unlearning set, yielding a mathematically equivalent but vastly more sample-efficient unlearning loss, $\mathcal{L}_{\text{unlearn}}$. In our framework, importance sampling is intrinsic: removing it would reduce our approach back to the vanilla loss, eliminating any meaningful unlearning effect.

Finally, we note that the numerical discrepancies between the results reported in our evaluation and those in the SISS publication are simply due to the different numbers of fine-tuning steps utilized in our experimental setup.

C DETAILED EXPERIMENTAL SETUP

In this section, we present a detailed description of the experimental setup, including the hardware and software environment, the datasets and models, the evaluation metrics, and the training hyperparameter settings employed in the experiments.

C.1 Computational Environment

All experiments were conducted on NVIDIA A100-SXM4-40GB GPU hardware and implemented using the `PyTorch` framework and the Hugging Face `Diffusers` library. The AdamW optimizer (Loshchilov and Hutter, 2017) was employed for network optimization.

C.2 Datasets and Models

We conducted experiments on four datasets: MNIST T-Shirt, CelebA-HQ, CIFAR-10, and Stable Diffusion. For CelebA-HQ and CIFAR-10, we employed the pretrained models provided in Ho, Jain, and Abbeel (2020). The MNIST T-Shirt dataset and its pretrained model were adopted directly from Alberti et al. (2025). For Stable Diffusion, we used version 1.4 and the corresponding dataset constructed by Alberti et al. (2025). All diffusion models use UNet as the backbone, with a total diffusion steps $T = 1000$. We use the EMA version for all pretrained models.

During fine-tuning, all models were initialized from the provided pretrained checkpoints without any additional modifications. For Stable Diffusion model, we trained only the denoising UNet while keeping all other components fixed.

C.3 Evaluation Metrics

For the Frequency metric on MNIST T-Shirt, we used the ℓ_2 distance to determine whether a generated image belongs to T-Shirts. We computed the NLL metric according to the process presented in Song et al. (2020). The SSCD metric was computed using the default `sscd_disc_mixup` model recommended by Pizzi et al. (2022). FID, IS, and CLIP-IQA were all evaluated via the `torchmetrics` library. For IS, we used a ten-class classifier trained on MNIST for the MNIST T-Shirt dataset and the standard Inception V3 classifier for CIFAR-10.

C.4 Training Hyperparameter Settings

On the MNIST T-Shirt, CelebA-HQ, CIFAR-10, and Stable Diffusion datasets, the learning rate during fine-tuning was kept constant at 5×10^{-5}, 5×10^{-6}, 2×10^{-5}, and 1×10^{-5} respectively. These values were chosen to ensure good performance of the vanilla method. The training batch size was set to 128, 64, 128, and 16 respectively. For the MNIST T-Shirt, CelebA-HQ, and CIFAR-10 datasets, we used AdamW with beta parameters set to (0.95, 0.999) and weight decay of 1×10^{-6}. For the Stable Diffusion dataset, beta was set to (0.9, 0.999) and weight decay to 1×10^{-2}. To balance generation quality with inference efficiency, we set the inference steps to 1000 for CIFAR-10 and to 50 for all other datasets. All hyperparameter configurations are available in the YAML files under the `/configs` directory in our code.

D ADDITIONAL RESULTS

In this section, we present more visualization results on each dataset. For image generation quality visualizations on each dataset, see Figure 5, 6, 7, 8, 9 and 10. For the images used during the calculation of the SSCD metric for the CelebA-HQ and CIFAR-10 datasets, see Figure 11 and 12.

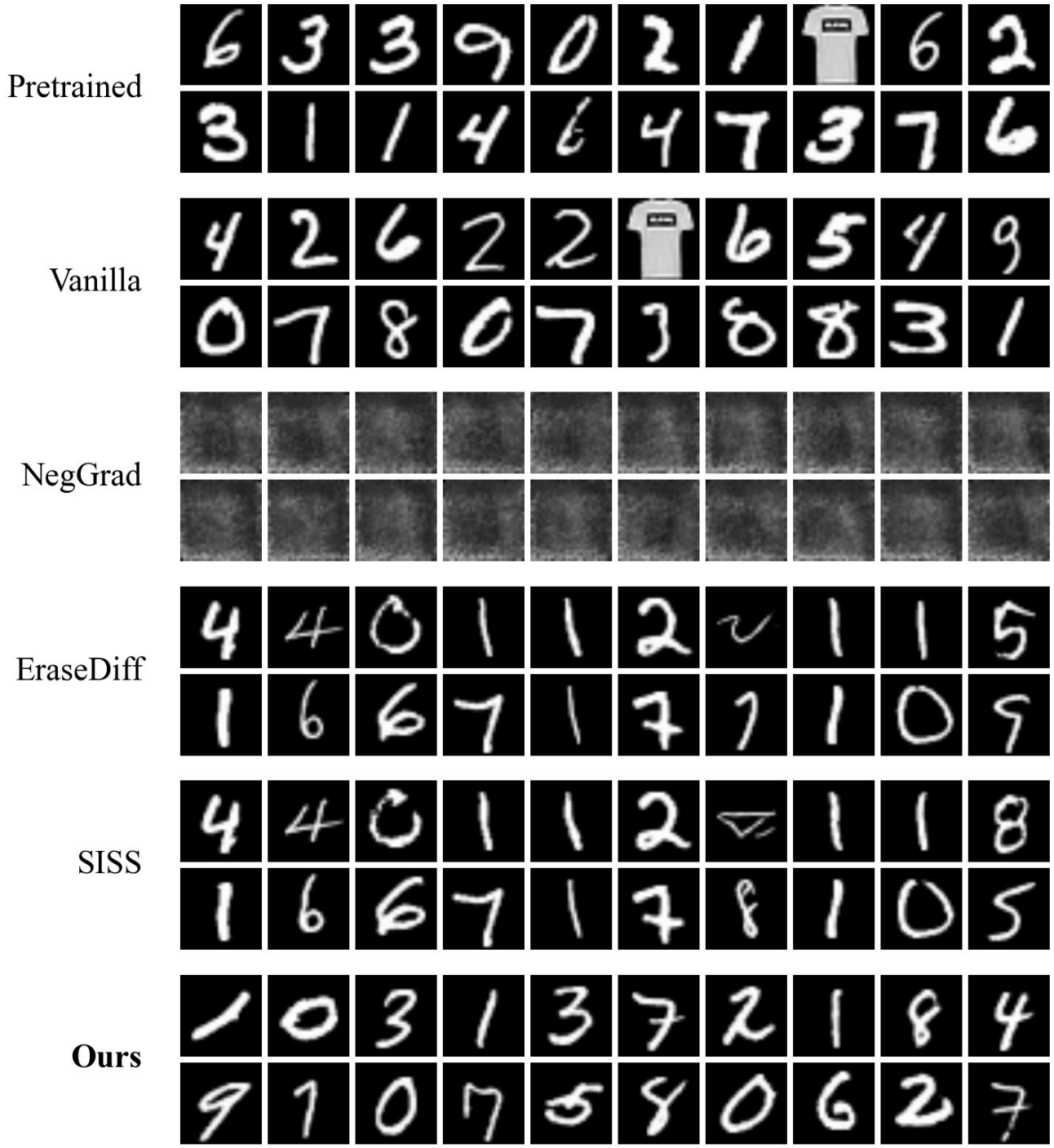

Figure 5: Partial sampling results from pretrained and unlearned models on the MNIST T-Shirt dataset.

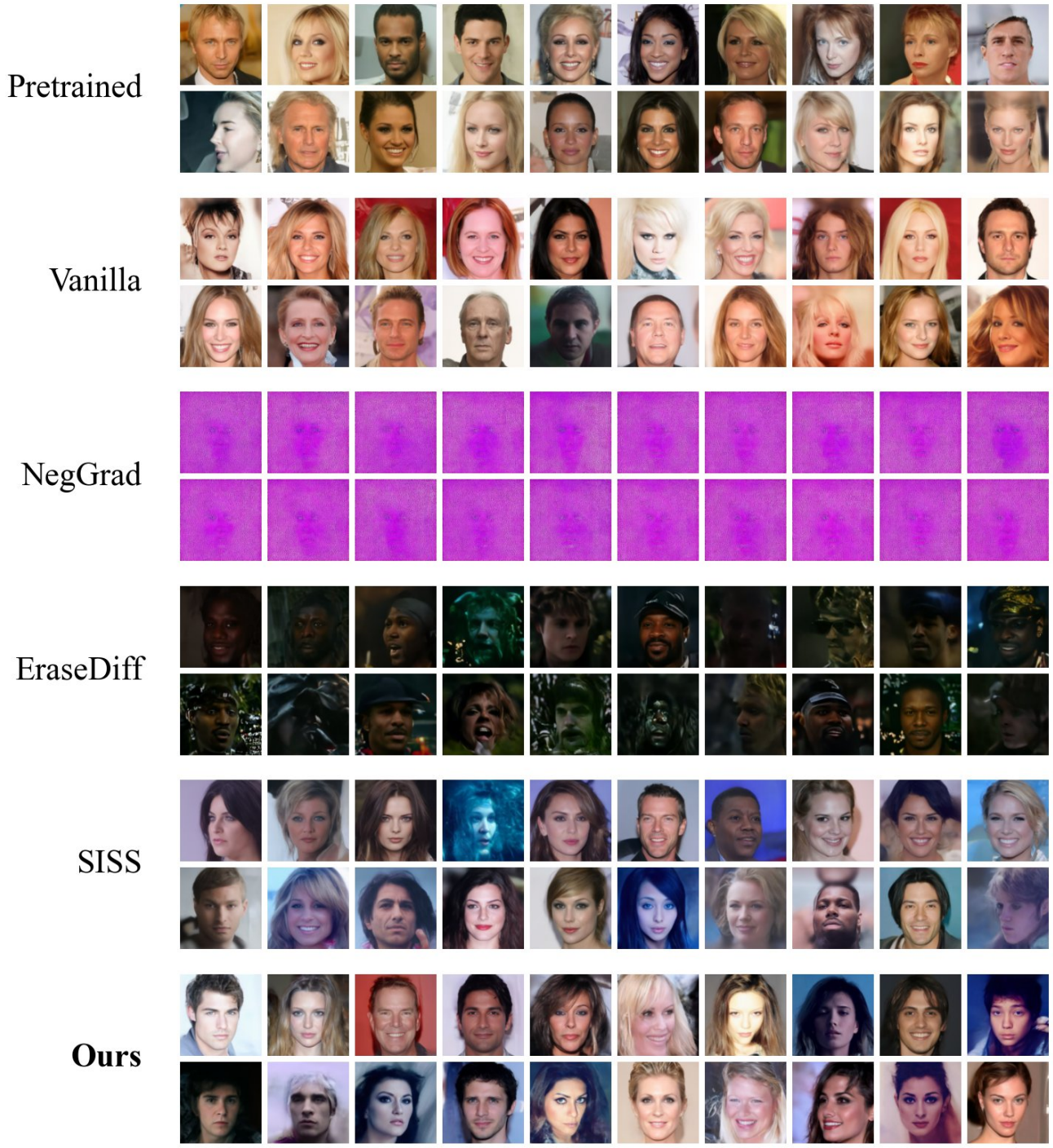

Figure 6: Partial sampling results from pretrained and unlearned models on the CelebA-HQ dataset.

Figure 7: Partial sampling results from pretrained and unlearned models on the CIFAR-10 dataset.

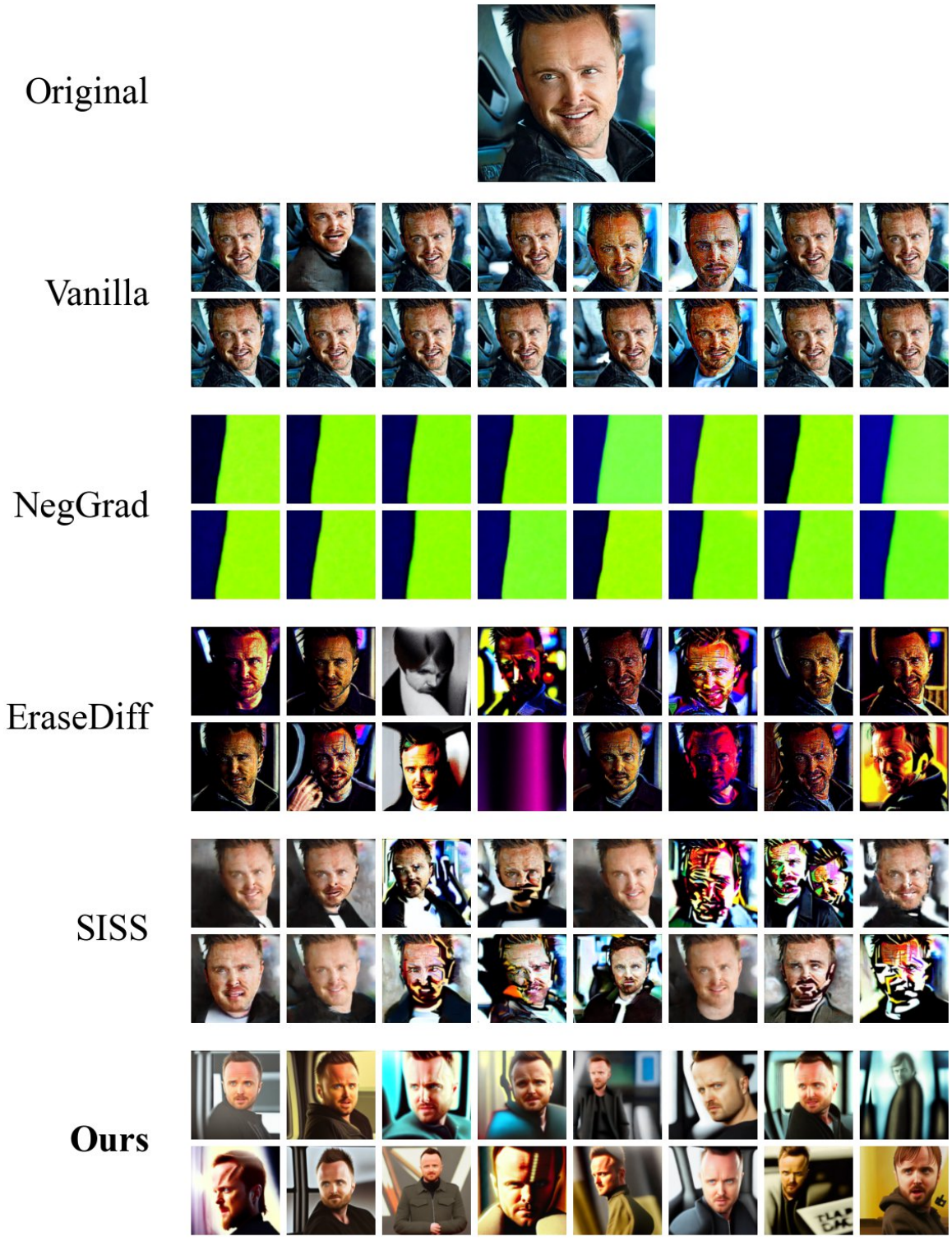

Figure 8: Sampling results from pretrained and unlearned models on the Stable Diffusion dataset conditioned on the prompt "Aaron Paul to Play Luke Skywalker at LACMA Reading of *The Empire Strikes Back*".

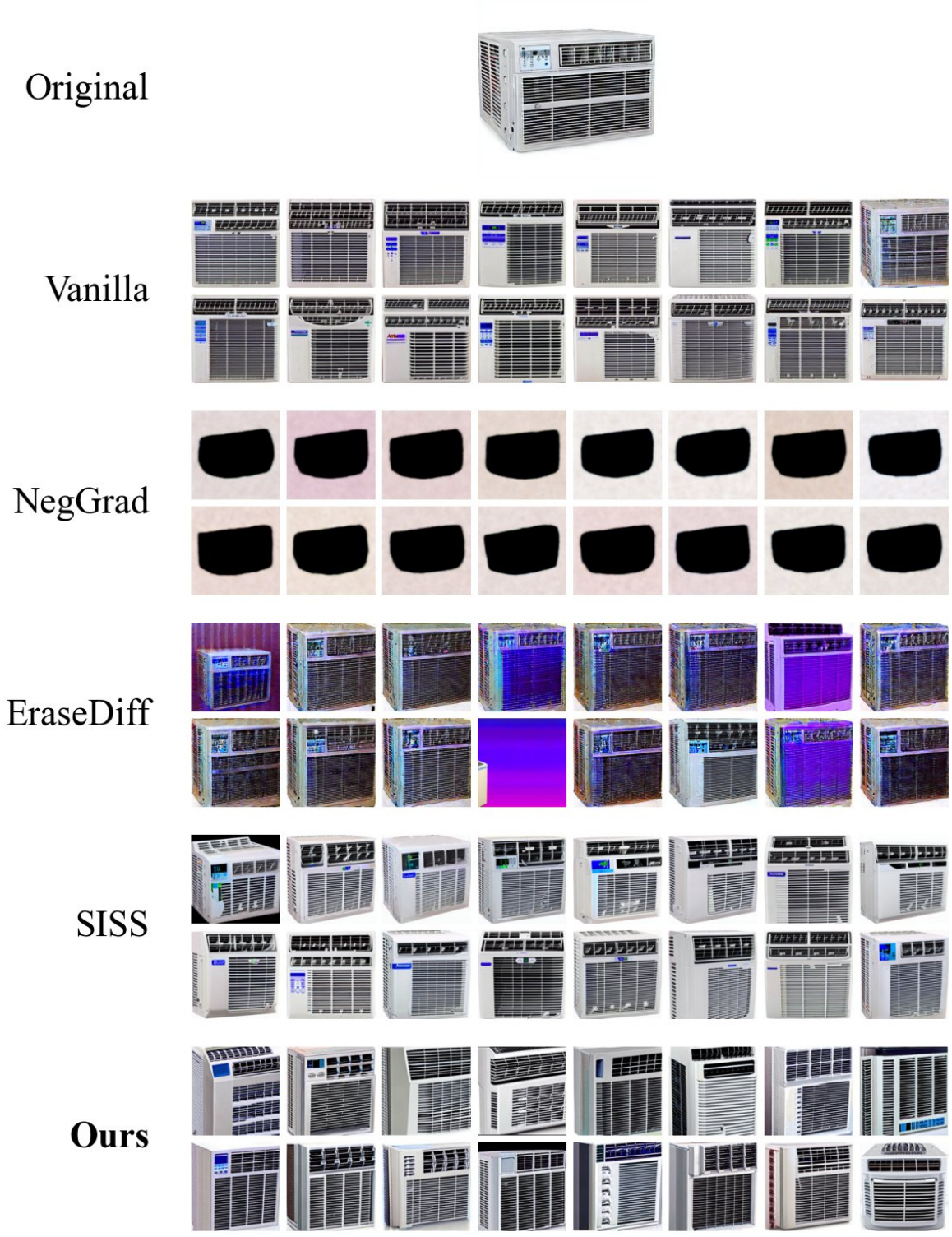

Figure 9: Sampling results from pretrained and unlearned models on the Stable Diffusion dataset conditioned on the prompt "Air Conditioners & Parts".

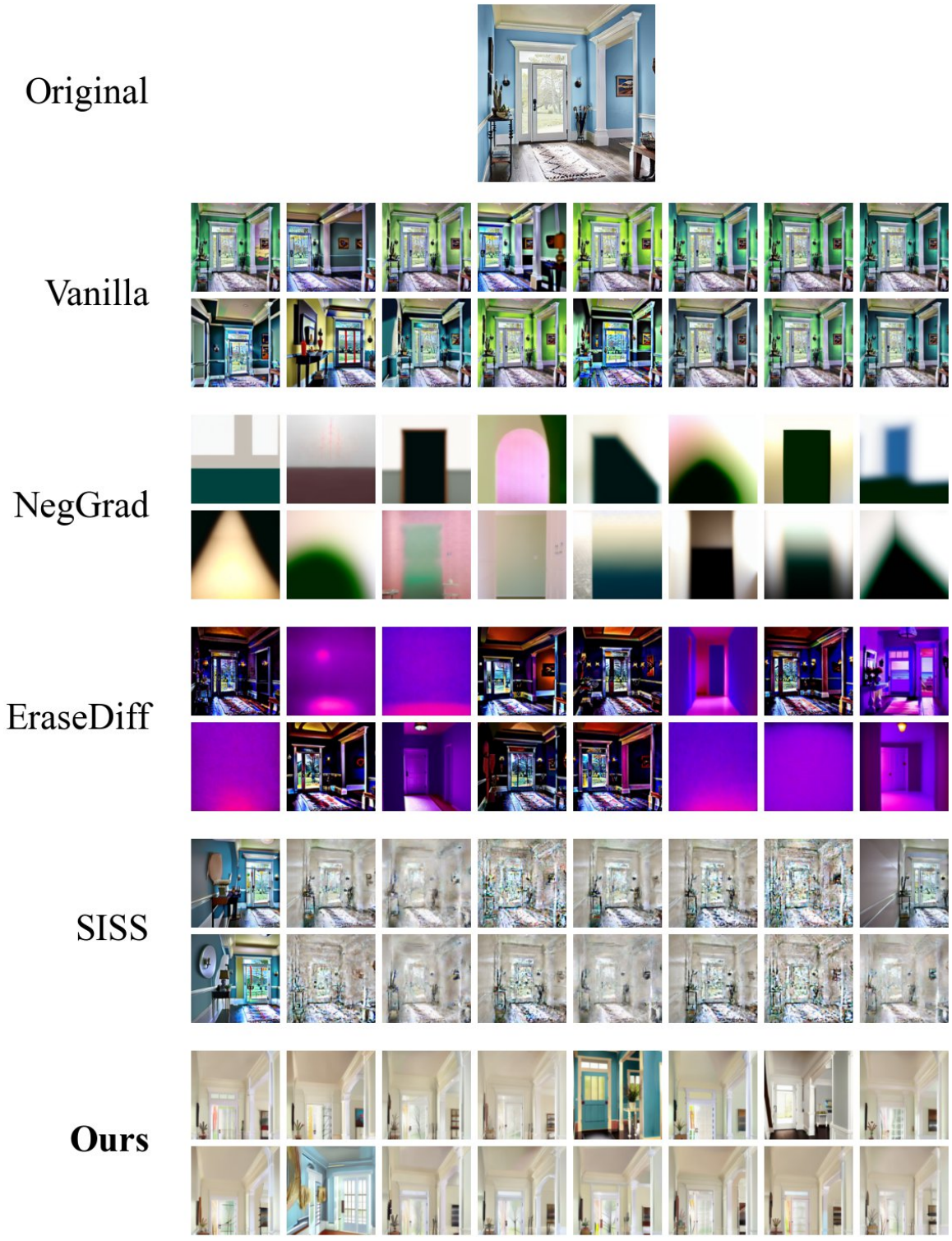

Figure 10: Sampling results from pretrained and unlearned models on the Stable Diffusion dataset conditioned on the prompt "Foyer painted in SALTY AIR".

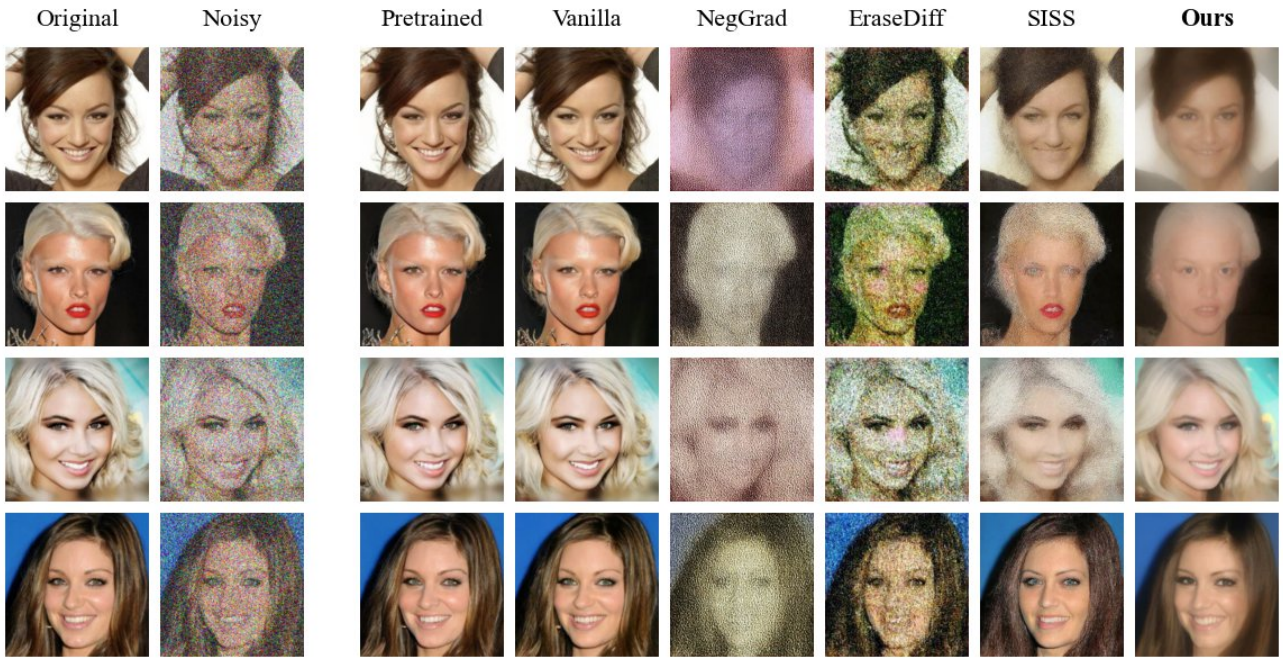

Figure 11: Partial reconstruction images from pretrained and unlearned models on the CelebA-HQ dataset when computing the SSCD metric.

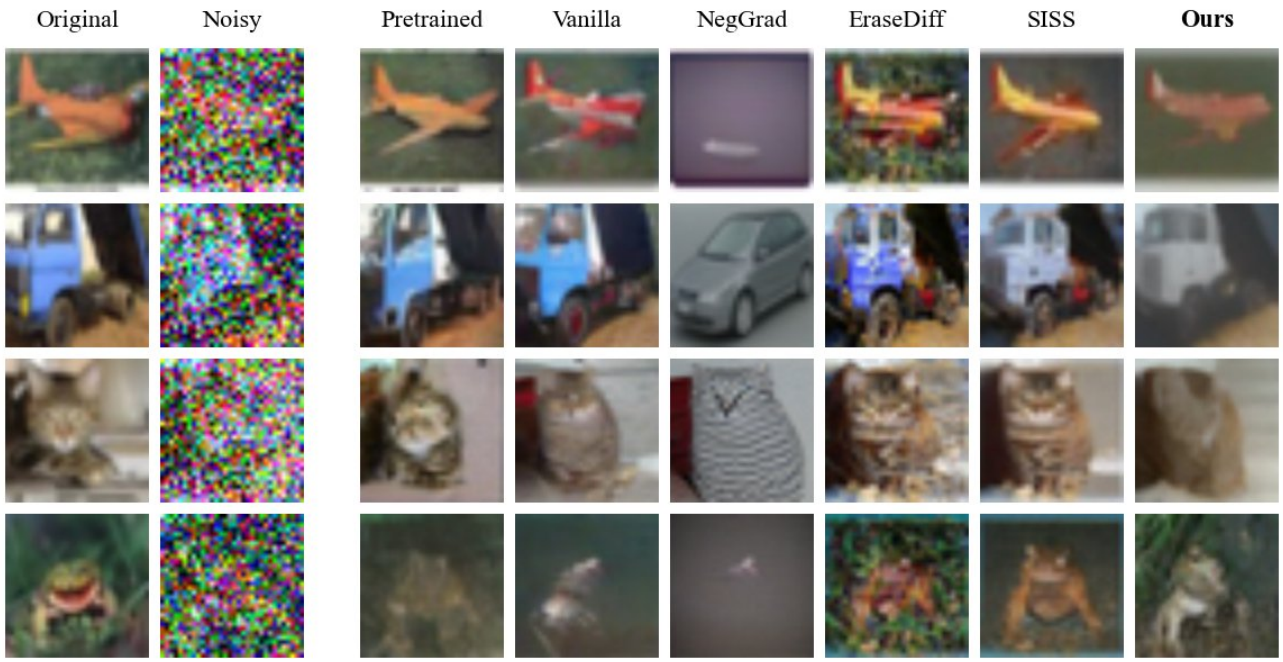

Figure 12: Partial reconstruction images from pretrained and unlearned models on the CIFAR-10 dataset when computing the SSCD metric.