CONTINUAL DATA UNLEARNING IN DIFFUSION MODELS VIA TRANSITION-BASED REGULARIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Data unlearning in diffusion models aims to remove the influence of specific training examples without suppressing the broader concepts they represent. However, when deletion requests arrive sequentially, updates for new requests can degrade generative utility and undermine earlier deletions. We propose a continual data unlearning framework that uses completed deletion transitions as directional references to regularize future updates. For each request, we record changes in denoiser responses on the same fixed noisy inputs before and after unlearning. Rather than matching full post-deletion responses, we apply a one-sided penalty that discourages reversal along the recorded directions relative to the post-deletion references, while leaving orthogonal response changes and progress beyond these references unpenalized. To keep storage independent of the number of requests, we maintain a fixed-capacity bank of representative transition records. Records are selected based on the local sensitivity of progress along their recorded directions to parameter updates, allowing them to be retained even when their penalties are inactive. Empirical evaluations show that the proposed framework achieves a better balance between deletion persistence and generative utility than existing unlearning baselines as requests accumulate, using only a small transition memory.

1 INTRODUCTION

Diffusion models have demonstrated remarkable capabilities in generating high-quality images (Rombach et al., 2022; Ramesh et al., 2022). These capabilities are enabled by training on massive image–text datasets collected from the internet (Schuhmann et al., 2022). However, the inclusion of copyrighted material and personal data in these datasets raises a range of ethical and legal concerns (Carlini et al., 2023; Dubiński et al., 2025). For instance, diffusion models can memorize and reproduce individual training samples, potentially exposing privacy-sensitive information or replicating copyrighted content (Carlini et al., 2023). Such concerns motivate mechanisms for addressing training-data deletion requests. This need is further underscored by privacy laws such as the General Data Protection Regulation (GDPR) and the California Consumer Privacy Act (CCPA), which recognize individuals' rights to request the deletion of their personal data under applicable conditions. A straightforward approach is to exclude request samples from the training set and retrain the model from scratch. However, retraining large-scale diffusion models for every deletion request is computationally prohibitive (Alberti et al., 2025). This motivates *machine unlearning*, which aims to efficiently remove the influence of these samples without full retraining while preserving model utility (Bourtoule et al., 2021; Alberti et al., 2025).

Research on unlearning in diffusion models has explored two distinct objectives: *concept unlearning* and *data unlearning*. Concept unlearning aims to suppress the model's ability to generate particular semantic concepts, such as objects, identities, or artistic styles, as explored by methods such as ESD and Forget-Me-Not (Gandikota et al., 2023; Zhang et al., 2024a). In contrast, data unlearning seeks to remove the influence of specific training examples rather than the broader concepts they represent (Alberti et al., 2025; Shi et al., 2026). Given a training dataset X and a subset $A \subset X$ requested for deletion, the model should ideally behave as though it had been trained from scratch on $X \setminus A$. For example, unlearning a particular photograph of a dog should not eliminate the model's ability to generate dogs. The model should still produce high-quality, text-aligned images for prompts associated with deleted samples, provided that the relevant concepts remain supported by the retained data. Concept-level suppression may therefore restrict generative capabilities beyond the scope of

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

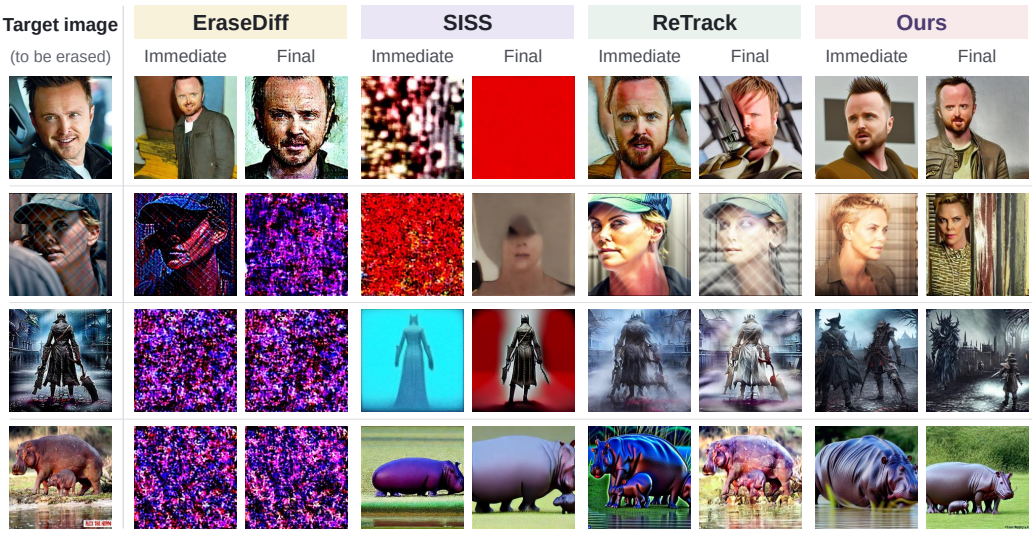

Figure 1: Qualitative comparison of continual data unlearning. Each row shows a target image to be erased and samples generated using its associated prompt. For each method, "Immediate" denotes generation immediately after deleting the target, while "Final" denotes generation after completing the subsequent deletion requests. The prompts used for the generations are provided in Appendix A.

an individual deletion request. Recent methods directly pursue this datapoint-level objective: SISS introduces an importance-sampling-based unlearning objective (Alberti et al., 2025), while ReTrack redirects denoising trajectories toward neighboring examples in the retained data (Shi et al., 2026).

In practical deployments, however, deletion requests may arrive sequentially rather than as a single, predetermined batch, necessitating *continual data unlearning*. Recent work has explored continual concept unlearning (Lee et al., 2026; George et al., 2026). For data unlearning, however, applying existing methods (Alberti et al., 2025; Shi et al., 2026) sequentially raises two key challenges. First, repeated updates may degrade image quality or text–image alignment, including for prompts associated with deleted samples whose underlying concepts remain supported by retained data. Failure to reproduce deleted samples does not, by itself, demonstrate successful unlearning, as it may reflect degraded generative capabilities rather than the selective removal of those samples' influence. Second, updates for new requests may weaken earlier deletions, allowing previously deleted samples to reappear in generated outputs. Figure 1 illustrates both failure modes in existing baselines as deletion requests accumulate. Continual data unlearning must therefore satisfy each new request while preserving generative utility and maintaining the effectiveness of all prior deletions.

In this paper, we propose a framework for continual data unlearning in diffusion models. Our key idea is to preserve the response change induced by a completed deletion rather than match the full post-deletion response. For each request, we record denoiser response changes on the same fixed noisy inputs before and after unlearning. During subsequent requests, a one-sided penalty discourages reversal along the recorded directions relative to the post-deletion references, while leaving orthogonal response changes and further movement along these directions unpenalized. To keep memory usage independent of request count, we maintain a fixed-capacity bank of representative transition records. Records are selected based on how parameter updates locally affect response projections along the recorded directions. This criterion can distinguish records even when their penalties are inactive.

The key contributions of this paper are summarized as follows:

- We introduce a transition-based perspective on continual data unlearning, with a one-sided regularizer that discourages reversal of deletion-induced response changes while leaving orthogonal changes and further progress unpenalized.
- We develop a fixed-capacity transition bank that selects records based on the local sensitivity of directional progress to parameter updates, even when their penalties are inactive.



Figure 2: Difference in SSCD for the first unlearned datapoint across sequential deletion requests on CelebA-HQ. Each path tracks a different example from an independent deletion sequence. The x-axis denotes the number of deletion requests processed. The y-axis shows this datapoint's SSCD after each request minus its SSCD immediately after the first deletion. Positive values indicate that SSCD is higher than it was immediately after the first deletion.

- We demonstrate a better balance between deletion persistence and generative utility than existing baselines as requests accumulate, using a small transition memory.

2 Preliminaries and Problem Setup

2.1 Data Unlearning in Diffusion Models

Data unlearning aims to remove the influence of selected training examples from a pretrained diffusion model (Alberti et al., 2025). We consider a diffusion model θ pretrained on a dataset $X = \{x_1, \ldots, x_n\}$ of n datapoints. A deletion request specifies a subset $A = \{a_1, \ldots, a_k\} \subset X$ of k datapoints to be unlearned. The goal is to efficiently update θ to approximate the behavior of a model retrained from scratch on the retained set $R = X \setminus A$. We consider this goal in both unconditional and text-to-image diffusion models. In particular, for text-to-image models, each datapoint consists of an image and its associated prompt. Removing a target image's influence should not compromise the model's ability to generate images aligned with the associated prompt, provided that the relevant concepts remain supported by the retained data.

2.2 Continual Data Unlearning

In practice, deletion requests may arrive sequentially rather than as a single batch. We consider a sequence of requests, where the t-th request provides a target datapoint $a_t^\star \in X$. Starting from the pretrained model θ_0, we update θ_{t-1} to θ_t upon receiving the t-th deletion request. *Once a request is completed, its raw datapoint is discarded and is no longer available for subsequent updates.*

To specify the cumulative unlearning objective, we define

$$A_t = \{a_1^\star, \ldots, a_t^\star\}, \qquad R_t = X \setminus A_t. \tag{1}$$

Here, A_t denotes the cumulative deletion set, not a dataset stored for subsequent updates. At each stage, the goal is to remove the influence of the current target while maintaining the effectiveness of earlier deletions and preserving generative utility, without accessing previously deleted raw datapoints. After all k requests, the final model ϵ_{θ_k} should therefore approximate the behavior of a model retrained from scratch on R_k, accounting for all requested deletions rather than only the most recent one.

3 Transition-Based Regularization for Continual Unlearning

Motivation. Continual data unlearning must preserve generative utility while maintaining the effects of earlier deletions as requests accumulate. However, repeated unlearning can degrade image quality (Figure 1), and the SSCD score for the first deletion target can rebound after an initial decline (Figure 2). This rebound suggests that later updates can partially reverse the effects of an earlier deletion.

Overview. For each deletion request, we record the changes in the model's denoising responses on fixed probes. During subsequent requests, bounded gradient corrections discourage updates that counteract these recorded changes, without requiring the full responses to remain unchanged. To limit storage, we select a fixed-size memory of records based on their local sensitivity to parameter updates. The overall algorithm is given in Appendix B.

3.1 Recording Response Transitions

Building on continual-learning methods that regularize future updates using stored model responses (Buzzega et al., 2020), we record the response changes induced by each deletion and use them as references for subsequent requests. To characterize these changes, while the target $a_i^\star$ is available, we construct Q fixed evaluation instances, termed *probes*, denoted by $\xi_{i,q}$, $q = 1, \dots, Q$. Each probe consists of a diffusion timestep, a noisy target obtained by forward noising $a_i^\star$ to that timestep, and, when applicable, its conditioning. These components remain fixed throughout all evaluations, such that only the model parameters vary.

Let θ_{i-1} and θ_i denote the model parameters immediately before and after processing request i, respectively. We denote the corresponding denoiser output for probe $\xi_{i,q}$ by $\Phi(\theta; \xi_{i,q})$. The corresponding pre- and post-request responses are

$$y_{i,q}^- = \Phi(\theta_{i-1}; \xi_{i,q}), \qquad y_{i,q}^+ = \Phi(\theta_i; \xi_{i,q}). \tag{2}$$

We record

$$d_{i,q} = y_{i,q}^+ - y_{i,q}^-, \qquad \tau_{i,q} = \langle y_{i,q}^+, d_{i,q} \rangle, \tag{3}$$

where $\langle \cdot, \cdot \rangle$ denotes the Euclidean inner product. The vector $d_{i,q}$ captures the change in the probe response over request i. The scalar $\tau_{i,q}$ is a post-request reference to detect later reversal along $d_{i,q}$.

The probes and their recorded quantities form the transition record

$$\mathcal{T}_i = \{(\xi_{i,q}, d_{i,q}, \tau_{i,q}) : \|d_{i,q}\|_2 > 0\}. \tag{4}$$

Later checks use $\mathcal{T}_i$ and the current model without accessing the raw deleted datapoint or an earlier model checkpoint. Probe construction and storage details are given in Appendix C.

3.2 Measuring and Penalizing Reversal

Consider a probe whose response changed from $y_{i,q}^-$ to $y_{i,q}^+$ during deletion. We ask whether a later response has moved back along this change, using the post-request response $y_{i,q}^+$ as the reference point. We measure this movement by the signed margin

$$m_{i,q}(\theta) = \left\langle \Phi(\theta; \xi_{i,q}) - y_{i,q}^+, \frac{d_{i,q}}{\|d_{i,q}\|_2} \right\rangle = \frac{\langle \Phi(\theta; \xi_{i,q}), d_{i,q} \rangle - \tau_{i,q}}{\|d_{i,q}\|_2}. \tag{5}$$

The margin is zero immediately after deletion and equals $-\|d_{i,q}\|_2$ at the model state immediately before processing request i. A negative margin indicates that the response has moved backward along the recorded transition, even if it has not returned to the pre-request response. The second expression allows this margin to be evaluated without storing $y_{i,q}^+$.

We penalize only the negative margins.

$$G_{i,q}(\theta) = [-m_{i,q}(\theta)]_+^2, \qquad [x]_+ = \max\{x, 0\}. \tag{6}$$

Unlike full-response matching, this penalty leaves changes orthogonal to $d_{i,q}$ and further movement along $d_{i,q}$ unpenalized.

3.3 Applying Corrections

At each optimizer step during request t, we first check the next probe in the newest record $\mathcal{T}_{t-1}$. If its margin is negative, indicating reversal, we use the corresponding penalty gradient to form the correction. Otherwise, we select an older retained record according to the service-weight schedule described in Section 3.4 and check its next probe. We cycle through the probes within each record and apply at most one correction per step.

Let $L_t^{\text{base}}(\theta)$ denote the ReTrack-based unlearning objective (Shi et al., 2026) for the current request, with gradient $g^{\text{base}} = \nabla_\theta L_t^{\text{base}}(\theta)$. For a selected probe $\xi_{i,q}$ with $m_{i,q}(\theta) < 0$, we apply the historical correction

$$g = g^{\text{base}} + \text{clip}_{\rho\|g^{\text{base}}\|_2}\left(\omega \nabla_\theta G_{i,q}(\theta)\right), \tag{7}$$

where clip_c limits the Euclidean norm to at most c, $\omega \geq 0$ scales the correction, and $\rho > 0$ bounds its norm relative to the base gradient. If no checked probe has a negative margin, $g = g^{\text{base}}$. This clipped soft correction discourages reversal without enforcing a nonnegative margin at every step.

3.4 Maintaining a Fixed-Size Memory

We keep the newest record $\mathcal{T}_t$ separately throughout request $t+1$ before considering it for long-term storage. Let $\mathcal{I}_t$ index the older records retained after request t, with $|\mathcal{I}_t| \leq K_{\text{mem}}$. At the end of request t, the previous bank and the preceding record $\mathcal{T}_{t-1}$ form the candidate set:

$$\mathcal{C}_t = \mathcal{I}_{t-1} \cup \{t-1\}, \qquad t \geq 2, \tag{8}$$

with $\mathcal{C}_1 = \mathcal{I}_1 = \emptyset$.

Selection by local sensitivity. With limited memory, we prioritize records whose margins are sensitive to different update directions rather than retaining many with similar responses. However, current penalty gradients vanish for probes with nonnegative margins and thus cannot distinguish their sensitivity to later updates, which may still reverse earlier deletions. We therefore examine the margin gradients $\nu_{i,q} = \nabla_\theta m_{i,q}(\theta_t)$. For a small parameter update $\Delta\theta$,

$$m_{i,q}(\theta_t + \Delta\theta) \approx m_{i,q}(\theta_t) + \langle \nu_{i,q}, \Delta\theta \rangle. \tag{9}$$

A negative inner product indicates a decrease in the margin, even when the current penalty is zero. Similar gradient directions therefore identify margins that tend to decrease under similar local updates. We use this similarity to reduce redundancy among retained records.

To compare these directions efficiently, we apply the same CountSketch-style linear map to all candidate margin gradients S_t (Wang et al., 2026). We normalize the sketched gradients to compare directions rather than magnitudes, then concatenate them in a fixed probe order to form each record's signature:

$$v_{i,q} = \frac{S_t \nu_{i,q}}{\|S_t \nu_{i,q}\|_2}, \qquad u_i = \frac{1}{\sqrt{Q}}[v_{i,1}; \ldots; v_{i,Q}], \tag{10}$$

where concatenation follows a fixed probe order.

For a nonempty candidate set, we select records so that convex combinations of their signatures approximate all candidate signatures.

$$\mathcal{I}_t \in \arg \min_{\substack{\mathcal{J} \subseteq \mathcal{C}_t \\ |\mathcal{J}| = \min(K_{\text{mem}}, |\mathcal{C}_t|)}} \sum_{i \in \mathcal{C}_t} \min_{\eta_i \in \Delta_{\mathcal{J}}} \left\| u_i - \sum_{j \in \mathcal{J}} \eta_{i,j} u_j \right\|_2^2, \tag{11}$$

where $\Delta_{\mathcal{J}}$ is the simplex over the selected records. The objective selects signatures that jointly approximate the candidates' local sensitivity directions.

Allocating future checks. Each record carries a positive *service weight* μ_i for scheduling future checks. When a candidate record is discarded, we redistribute its weight among the selected records according to its fitted coefficient vector $\eta_i^\star$. Let $\boldsymbol{\mu}$ collect the selected records' weights before redistribution:

$$\boldsymbol{\mu}^+ = \boldsymbol{\mu} + \sum_{i \in \mathcal{C}_t \setminus \mathcal{I}_t} \mu_i \eta_i^\star. \tag{12}$$

The fitted coefficients sum to one for each discarded record, so this redistribution preserves total service weight. The normalized weights determine the next request's check schedule, not record selection.

For fixed K_{mem} and Q, we retain at most $K_{\text{mem}} + 1$ transition records between requests and compute at most $(K_{\text{mem}} + 1)Q$ progress gradients per memory update. Both bounds are independent of the number of deletion requests.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Diffusion models and datasets. Following prior work on data unlearning in diffusion models (Alberti et al., 2025; Shi et al., 2026), we evaluate our method on unconditional diffusion models for CIFAR-10 and CelebA-HQ, and on Stable Diffusion v1.4 for text-to-image generation. For the unconditional models, each deletion sequence consists of 50 image deletions. For Stable Diffusion v1.4, we perform sequential unlearning on 48 memorized prompt–image pairs from SISS and assess general utility using a separate set of 1,000 COCO captions (Lin et al., 2014).

Evaluation metrics. We assess unlearning effectiveness using the Self-Supervised Copy Detection (SSCD) score (Pizzi et al., 2022), which measures object-level similarity between generated images and the corresponding deletion targets. For unconditional diffusion models, we evaluate generation quality using Fréchet Inception Distance (FID) (Heusel et al., 2017). On CelebA-HQ, we additionally compute the negative log-likelihood (NLL) of deletion targets using the exact likelihood formulation of Song et al. (2021), with higher NLL indicating lower likelihood assigned to these targets. For text-to-image generation, we evaluate image quality using CLIP-IQA (Wang et al., 2023) and image–text alignment using the CLIP score (Radford et al., 2021), which measures semantic consistency between generated images and their corresponding prompts. Following SISS, we also report the unlearning success rate. For each deletion target, we generate 16 samples from the final model using its corresponding prompt. Unlearning is considered successful for a target if none of these samples is identified as memorized and their mean CLIP-IQA score is at least 0.35. To assess general utility, we further report CLIP-IQA and CLIP scores for images generated from the COCO captions.

Baselines. We compare our method with Naive (retain-only fine-tuning), NegGrad (Golatkar et al., 2020), EraseDiff (Wu et al., 2025), SISS (Alberti et al., 2025), and ReTrack (Shi et al., 2026). We match the pretrained initialization, deletion orders, and the number of base optimization steps per request across methods. Each method uses its own updated model as the starting point for the next request. For all methods, raw datapoints from completed deletion requests are no longer available for subsequent updates.

Implementation details. For CIFAR-10 and CelebA-HQ, we use pretrained diffusion models from Ho et al. (2020). For SSCD evaluation, we apply forward noising to each target image at diffusion timestep 250 using a fixed noise, then reconstruct it with 250 DDPM denoising steps, following SISS (Alberti et al., 2025) and ReTrack (Shi et al., 2026). For Stable Diffusion v1.4, we generate 16 images per prompt variant using 50-step DDIM sampling with a guidance scale of 7.5 and the same initial noise across methods. Further details are in Appendix D.

4.2 CONTINUAL UNLEARNING PERFORMANCE

Table 1: Results after sequentially unlearning 50 datapoints on CelebA-HQ and CIFAR-10. We report mean and standard deviation over 3 runs with randomly sampled target sets and deletion orders.

	CelebA-HQ			CIFAR-10	
Method	SSCD ↓	FID ↓	NLL ↑	SSCD ↓	FID ↓
Naive	0.8595 ± 0.0017	20.41 ± 4.84	1.3281 ± 0.0296	0.5436 ± 0.0099	11.38 ± 1.02
NegGrad	0.1131 ± 0.0153	441.47 ± 95.44	18.4781 ± 1.0963	0.0539 ± 0.0507	431.07 ± 64.01
EraseDiff	0.2008 ± 0.0123	141.31 ± 87.25	5.8061 ± 0.1735	0.4567 ± 0.0110	33.72 ± 23.26
SISS	0.6781 ± 0.0267	38.80 ± 31.13	1.3819 ± 0.0301	0.4893 ± 0.0135	$\mathbf{11.16} \pm 2.73$
ReTrack	0.4110 ± 0.0686	31.48 ± 9.53	2.2861 ± 0.1426	0.4539 ± 0.0240	16.80 ± 2.56
Ours	$\mathbf{0.3965} \pm 0.0739$	$\mathbf{25.51} \pm 6.18$	$\mathbf{2.3421} \pm 0.1286$	$\mathbf{0.4461} \pm 0.0253$	16.04 ± 2.80

Continual unlearning on unconditional models. Table 1 reports the results after sequentially unlearning 50 datapoints on CelebA-HQ and CIFAR-10. Naive yields relatively low FID but the

highest SSCD on both datasets, indicating greater reconstruction similarity to the deletion targets. Conversely, NegGrad achieves the lowest SSCD and, on CelebA-HQ, the highest target NLL, but its FID exceeds 430 on both datasets. These results highlight that favorable unlearning metrics alone can coincide with severe degradation in generation quality. EraseDiff exhibits a similar trade-off on CelebA-HQ, achieving low SSCD at an FID of 141.31.

With $K_{\text{mem}} = 4$, our method improves all reported metrics over ReTrack on both datasets. On CelebA-HQ, it reduces FID by 19.0% relative to ReTrack while achieving lower SSCD and higher target NLL, and also outperforms SISS across all metrics. Although SISS achieves a lower FID on CIFAR-10, our method attains lower SSCD while maintaining generation quality comparable to ReTrack. Overall, these results support a favorable balance between unlearning effectiveness and generation quality, without the severe quality degradation observed with NegGrad.

Table 2: Results after sequentially unlearning 48 memorized datapoints from Stable Diffusion v1.4. We evaluate unlearning performance using prompts corresponding to the deletion targets and general generation utility using COCO captions. We report the mean and standard deviation over three runs with different deletion orders.

	Unlearning performance				General utility (COCO)	
Method	Success (%) ↑	CLIP-IQA ↑	SSCD ↓	CLIP score ↑	CLIP-IQA ↑	CLIP score ↑
Pretrained	0.00	0.8112	0.6548	28.6885	0.8866	26.6049
Naive	25.69 ±15.91	0.7169 ±0.0365	0.2521 ±0.0133	32.0518 ±0.2843	0.8871 ±0.0031	27.0232 ±0.0699
NegGrad	0.00 ±0.00	0.1870 ±0.0484	0.0207 ±0.0074	18.2460 ±0.5542	0.2027 ±0.0291	14.0139 ±0.3815
EraseDiff	43.75 ±5.51	0.5312 ±0.0492	0.1226 ±0.0224	27.4282 ±1.1531	0.8887 ±0.0081	26.8076 ±0.0649
SISS	87.50 ±4.17	0.6728 ±0.0505	**0.1454** ±0.0106	26.1410 ±0.6199	0.8907 ±0.0059	**26.8203** ±0.1474
ReTrack	91.67 ±2.09	0.7679 ±0.0515	0.2049 ±0.0312	25.8044 ±0.7549	**0.8913** ±0.0122	26.7416 ±0.1601
Ours	**98.61** ±1.20	**0.8052** ±0.0349	0.1485 ±0.0189	**26.5527** ±0.3633	0.8878 ±0.0143	26.7499 ±0.2079

Continual unlearning for text-to-image diffusion. Table 2 reports the results after sequentially unlearning 48 memorized datapoints from Stable Diffusion v1.4, evaluating unlearning on deletion-target prompts and general utility on COCO. Naive largely preserves COCO utility but achieves limited unlearning success. NegGrad yields the lowest SSCD, yet its success rate is zero, with severe degradation in image quality and prompt alignment. EraseDiff also attains low SSCD but achieves only 43.75% success. These results highlight that low target similarity alone does not ensure successful unlearning with acceptable image quality.

With $K_{\text{mem}} = 2$, our method achieves the highest unlearning success rate of 98.61%. It also achieves lower SSCD and higher CLIP-IQA and CLIP scores than ReTrack on deletion-target prompts. Although SISS achieves slightly lower SSCD, our method attains a higher success rate and better image quality and prompt alignment. On COCO, our method's CLIP-IQA and CLIP scores remain close to the pretrained reference.

Table 3: Continual unlearning performance on CelebA-HQ across different deletion sequence lengths and memory capacities. We report SSCD after 20, 50, and 100 deletions and FID after 100 deletions, with $K_{\text{mem}} \in \{2, 3, 4\}$ for our method. Results are the mean and standard deviation over three runs with different target sets and deletion orders.

	SSCD ↓			FID ↓
Method	20 deletions	50 deletions	100 deletions	100 deletions
SISS	0.6936 ± 0.0312	0.6826 ± 0.0209	0.6873 ± 0.0103	**17.70** ± 2.02
ReTrack	0.5493 ± 0.0354	0.4792 ± 0.0395	0.5030 ± 0.0258	27.35 ± 4.50
Ours ($K_{\text{mem}} = 2$)	**0.4900** ± 0.0198	0.4497 ± 0.0528	0.4846 ± 0.0254	25.98 ± 2.13
Ours ($K_{\text{mem}} = 3$)	0.4921 ± 0.0236	0.4465 ± 0.0370	0.4778 ± 0.0302	22.18 ± 1.71
Ours ($K_{\text{mem}} = 4$)	0.4913 ± 0.0236	**0.4434** ± 0.0377	**0.4654** ± 0.0347	22.73 ± 1.30

4.3 Robustness and Design Analysis

We further analyze our method from two perspectives. For robustness, we study longer deletion sequences under different memory capacities, with additional analyses of deletion-order sensitivity and unlearning persistence provided in Appendix E. For design analysis, we examine the choice of response-preservation penalty and memory-selection rule. Resource costs are also reported in Appendix F.

Longer deletion sequences and memory capacity. Table 3 evaluates different deletion sequence lengths and memory capacities on CelebA-HQ. Our method achieves lower SSCD than both SISS and ReTrack at all evaluated sequence lengths, even with $K_{\mathrm{mem}} = 2$. The three capacities yield similar SSCD after 20 deletions, whereas increasing memory capacity yields progressively lower SSCD after 50 and 100 deletions. At 100 deletions, $K_{\mathrm{mem}} = 4$ reduces SSCD from ReTrack's 0.5030 to 0.4654, a 7.5% reduction. All three capacities also achieve lower FID than ReTrack after 100 deletions, showing that improved target suppression is accompanied by better generation quality. Overall, our method maintains its advantage over longer deletion sequences with a small, fixed memory capacity, while additional memory provides greater benefits as the sequence length increases.

Table 4: Comparison of response-preservation penalties on CelebA-HQ and Stable Diffusion v1.4 after 50 and 48 deletions, respectively. All variants use $K_{\mathrm{mem}} = 2$. We report the mean and standard deviation over three runs with different deletion orders.

	CelebA-HQ		Stable Diffusion v1.4			
Penalty	SSCD ↓	FID ↓	Success (%) ↑	CLIP-IQA ↑	SSCD ↓	CLIP score ↑
Full-output	0.4326 ± 0.0404	33.69 ± 9.26	92.36 ± 1.20	0.8013 ± 0.0309	0.1155 ± 0.0168	$\mathbf{26.6624} \pm 0.2519$
Symmetric	0.4391 ± 0.0279	30.03 ± 10.79	93.75 ± 3.61	0.7937 ± 0.0299	$\mathbf{0.1061} \pm 0.0190$	26.4982 ± 0.1673
Ours	$\mathbf{0.3975} \pm 0.0680$	$\mathbf{27.62} \pm 9.03$	$\mathbf{98.61} \pm 1.20$	$\mathbf{0.8052} \pm 0.0349$	0.1485 ± 0.0189	26.5527 ± 0.3633

Choice of response-preservation penalty. We compare three response-preservation penalties (Table 4). Full-output preservation penalizes all deviations from the post-deletion response, whereas symmetric directional preservation penalizes both reversal and further movement along the recorded deletion direction. Our one-sided penalty penalizes only reversal, allowing movement beyond the post-deletion reference and orthogonal changes. On CelebA-HQ, our one-sided penalty achieves the lowest SSCD and FID among the three variants. On Stable Diffusion v1.4, it achieves the highest success rate of 98.61%, compared with 92.36% for full-output preservation and 93.75% for symmetric directional preservation. It also yields the highest CLIP-IQA while maintaining comparable prompt alignment. Although the other penalties attain lower SSCD, this does not translate into higher unlearning success. Overall, these results support penalizing only reversal rather than imposing broader response constraints in the evaluated settings.

Table 5: Comparison of memory-selection rules on CelebA-HQ and Stable Diffusion v1.4 after 50 and 48 deletions, respectively. All variants use $K_{\mathrm{mem}} = 2$. We report the mean and standard deviation over three runs with different deletion orders.

	CelebA-HQ		Stable Diffusion v1.4			
Selection	SSCD ↓	FID ↓	Success (%) ↑	CLIP-IQA ↑	SSCD ↓	CLIP ↑
FIFO	0.4400 ± 0.0269	32.78 ± 10.48	93.06 ± 3.18	0.7907 ± 0.0293	$\mathbf{0.1205} \pm 0.0150$	26.2898 ± 1.0175
Random	0.4379 ± 0.0288	31.39 ± 9.26	93.75 ± 5.51	0.7916 ± 0.0458	0.1216 ± 0.0111	$\mathbf{27.1066} \pm 0.4785$
Ours	$\mathbf{0.3975} \pm 0.0680$	$\mathbf{27.62} \pm 9.03$	$\mathbf{98.61} \pm 1.20$	$\mathbf{0.8052} \pm 0.0349$	0.1485 ± 0.0189	26.5527 ± 0.3633

Choice of memory-selection rule. We compare our gradient-signature selection with FIFO (first-in, first-out) and Random (Table 5). Each rule selects up to two records from the previous memory and the preceding deletion record. FIFO retains the most recent records, whereas Random samples uniformly without replacement from this candidate set. Previously discarded records are not reconsidered.

On CelebA-HQ, our selection achieves the lowest SSCD and FID, improving both target suppression and generation quality. On Stable Diffusion v1.4, it achieves the highest success rate of 98.61%, compared with 93.06% for FIFO and 93.75% for Random, together with the highest CLIP-IQA. Its success rate also exhibits the smallest standard deviation across deletion orders. These results support the effectiveness of gradient-signature selection over recency-based or random selection, enabling better unlearning performance with a small, fixed memory budget.

5 Related Work

Unlearning in Diffusion Models. Unlearning in diffusion models has been studied at both the concept and datapoint levels. Concept unlearning methods fine-tune pretrained diffusion models to suppress target concepts while preserving non-target generative capabilities (Gandikota et al., 2023; Kumari et al., 2023; Heng & Soh, 2023; Zhang et al., 2024a). Other approaches employ saliency-guided parameter updates (Fan et al., 2024), trainable concept-specific adapters (Lyu et al., 2024), or LoRA fine-tuning for multi-concept erasure (Lu et al., 2024). Further work incorporates adversarial training to improve the robustness of concept erasure against adversarial prompts (Kim et al., 2024; Zhang et al., 2024b; Srivatsan et al., 2025). In contrast, data unlearning aims to remove the influence of specific training examples without suppressing the broader concepts they represent. Representative approaches include EraseDiff's forgetting–retention optimization (Wu et al., 2025), SISS's importance-sampled objective (Alberti et al., 2025), ReTrack's redirection of target denoising trajectories toward retained examples (Shi et al., 2026).

Continual Unlearning in Diffusion Models. In practical deployments, deletion requests may arrive sequentially rather than as a single, predefined batch. However, successful unlearning need not persist under subsequent model updates: previously unlearned concepts can re-emerge after fine-tuning, even on benign data (George et al., 2025; Suriyakumar et al., 2026). This instability motivates continual unlearning, which requires earlier deletions to remain effective as new requests are processed. While Meta-Unlearning (Gao et al., 2025) seeks to make concept erasure resistant to downstream fine-tuning, recent studies explicitly address sequential concept deletion. Lee et al. (2026) investigate regularization and semantic-aware gradient projection to mitigate cumulative parameter drift, while LACU (George et al., 2026) employs locality-aware replacement targets and replay to protect neighboring retained concepts. The same challenge arises in data unlearning, where each new request must be satisfied while maintaining the effectiveness of earlier deletions and preserving generative utility. Unlike concept-level suppression, this requires removing individual samples' influence without suppressing their associated concepts, provided that these concepts remain supported by retained data.

6 Limitations

Our framework uses a fixed-capacity transition bank with a memory budget set in advance. Adaptive capacity allocation could improve flexibility when the number of deletion requests is unknown. Each transition is represented by a finite set of probes, which may not capture the deletion-induced response change exhaustively. Maintaining the bank also adds memory and computation, although these costs remain bounded for fixed capacity and probe count.

7 Conclusion

We presented a transition-based framework for sustaining data unlearning as deletion requests accumulate. Across unconditional and text-to-image diffusion models, our method strengthens deletion persistence while preserving generative utility. On Stable Diffusion, it achieves the highest unlearning success among the evaluated methods, with high-quality, prompt-aligned generation and general utility close to the pretrained reference. These gains are achieved with a small, fixed transition memory whose storage does not grow with the number of deletion requests. Overall, our results support preventing the reversal of deletion-induced response changes, rather than matching full post-deletion responses, as an effective approach to continual unlearning.

AI USE STATEMENT

Generative AI tools were used to assist with identifying related work, method discussion, developing and checking mathematical proofs, implementation and debugging, experiment planning and result analysis, and manuscript drafting and revision. The authors are responsible for the submission and its claims.

ETHICS STATEMENT

This work aims to improve the persistence of data unlearning in diffusion models. Although subsequent updates do not access previously deleted raw images, the method retains target-derived probes and response transitions. Our empirical evaluations do not establish certified erasure or protection against reconstruction.

REPRODUCIBILITY STATEMENT

Section 4 describes the datasets, baselines, and evaluation procedures. Appendix B provides pseudocode, Appendix C details probe construction and stored state, and Appendix D specifies training hyperparameters and implementation details. Appendix F describes the resource measurement protocol and reports memory and runtime costs.

REFERENCES

Silas Alberti, Kenan Hasanaliyev, Manav Shah, and Stefano Ermon. Data unlearning in diffusion models. In *International Conference on Learning Representations*, volume 2025, pp. 3084–3100, 2025.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE symposium on security and privacy (SP)*, pp. 141–159. IEEE, 2021.

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

Nicolas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwag, Florian Tramer, Borja Balle, Daphne Ippolito, and Eric Wallace. Extracting training data from diffusion models. In *32nd USENIX security symposium (USENIX Security 23)*, pp. 5253–5270, 2023.

Jan Dubiński, Antoni Kowalczuk, Franziska Boenisch, and Adam Dziedzic. Cdi: Copyrighted data identification in diffusion models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18674–18684. IEEE, 2025.

Chongyu Fan, Jiancheng Liu, Yihua Zhang, Eric Wong, Dennis Wei, and Sijia Liu. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. In *International Conference on Learning Representations*, volume 2024, pp. 53643–53673, 2024.

Rohit Gandikota, Joanna Materzynska, Jaden Fiotto-Kaufman, and David Bau. Erasing concepts from diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 2426–2436, 2023.

Hongcheng Gao, Tianyu Pang, Chao Du, Taihang Hu, Zhijie Deng, and Min Lin. Meta-unlearning on diffusion models: Preventing relearning unlearned concepts. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2131–2141. IEEE, 2025.

Naveen George, Karthik Nandan Dasaraju, Rutheesh Reddy Chittepu, and Konda Reddy Mopuri. The illusion of unlearning: The unstable nature of machine unlearning in text-to-image diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13393–13402, June 2025.

Naveen George, Naoki Murata, Yuhta Takida, Konda Reddy Mopuri, and Yuki Mitsufuji. Locality-aware continual unlearning for diffusion models. In Paolo Favaro, Zuzana Kukelova, Atsuto Maki, Anna Rohrbach, Konrad Schindler, and Federico Tombari (eds.), *Computer Vision – ECCV 2026*, pp. 389–407, Cham, 2026. Springer Nature Switzerland. ISBN 978-3-032-37167-6.

Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9301–9309. IEEE, 2020.

Alvin Heng and Harold Soh. Selective amnesia: A continual learning approach to forgetting in deep generative models. *Advances in Neural Information Processing Systems*, 36:17170–17194, 2023.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Changhoon Kim, Kyle Min, and Yezhou Yang. Race: Robust adversarial concept erasure for secure text-to-image diffusion model. In *European Conference on Computer Vision*, pp. 461–478. Springer, 2024.

Nupur Kumari, Bingliang Zhang, Sheng-Yu Wang, Eli Shechtman, Richard Zhang, and Jun-Yan Zhu. Ablating concepts in text-to-image diffusion models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22634–22645. IEEE, 2023.

Justin Lee, Zheda Mai, Jinsu Yoo, Chongyu Fan, Cheng Zhang, and Wei-Lun Chao. Continual unlearning for text-to-image diffusion models: A regularization perspective. In *International Conference on Learning Representations*, volume 2026, pp. 30660–30682, 2026.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Shilin Lu, Zilan Wang, Leyang Li, Yanzhu Liu, and Adams Wai-Kin Kong. Mace: Mass concept erasure in diffusion models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6430–6440. IEEE, 2024.

Mengyao Lyu, Yuhong Yang, Haiwen Hong, Hui Chen, Xuan Jin, Yuan He, Hui Xue, Jungong Han, and Guiguang Ding. One-dimensional adapter to rule them all: Concepts, diffusion models and erasing applications. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7559–7568. IEEE, 2024.

Ed Pizzi, Sreya Dutta Roy, Sugosh Nagavara Ravindra, Priya Goyal, and Matthijs Douze. A self-supervised descriptor for image copy detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. URL `https://arxiv.org/abs/2202.10261`.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021. URL `https://proceedings.mlr.press/v139/radford21a.html`.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 10674–10685. ieee, 2022.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in neural information processing systems*, 35:25278–25294, 2022.

Qitan Shi, Cheng Jin, Jiawei Zhang, and Yuantao Gu. Retrack: Data unlearning in diffusion models through redirecting the denoising trajectory. In *The 29th International Conference on Artificial Intelligence and Statistics*, 2026. URL https://openreview.net/forum?id=oTUZfa0iPv.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

Koushik Srivatsan, Fahad Shamshad, Muzammal Naseer, Vishal M Patel, and Karthik Nandakumar. Stereo: A two-stage framework for adversarially robust concept erasing from text-to-image diffusion models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23765–23774. IEEE, 2025.

Vinith Menon Suriyakumar, Rohan Alur, Ayush Sekhari, Manish Raghavan, and Ashia C. Wilson. Unstable unlearning: The hidden risk of concept resurgence in diffusion models. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=Vj0Z2wspQ5.

Jianyi Wang, Kelvin C. K. Chan, and Chen Change Loy. Exploring CLIP for assessing the look and feel of images. In *AAAI Conference on Artificial Intelligence*, 2023. URL https://arxiv.org/abs/2207.12396.

Shaobo Wang, Xuan Ouyang, Tianyi Xu, Yuzheng Hu, Jialin Liu, Guo Chen, Tianyu Zhang, Junhao Zheng, Kexin Yang, Xingzhang Ren, Dayiheng Liu, and Linfeng Zhang. OPUS: Towards efficient and principled data selection in large language model pre-training in every iteration. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=lcLxGrk5N9.

Jing Wu, Trung Le, Munawar Hayat, and Mehrtash Harandi. Erasing undesirable influence in diffusion models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 28263–28273. IEEE, 2025.

Gong Zhang, Kai Wang, Xingqian Xu, Zhangyang Wang, and Humphrey Shi. Forget-me-not: Learning to forget in text-to-image diffusion models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1755–1764. IEEE, 2024a.

Yimeng Zhang, Xin Chen, Jinghan Jia, Yihua Zhang, Chongyu Fan, Jiancheng Liu, Mingyi Hong, Ke Ding, and Sijia Liu. Defensive unlearning with adversarial training for robust concept erasure in diffusion models. *Advances in neural information processing systems*, 37:36748–36776, 2024b.

APPENDIX

Appendix Overview. This appendix provides additional details and analyses supporting the main paper. Appendix A lists the prompts used for the qualitative results in Figure 1. Appendix B presents the overall continual unlearning algorithm, followed by probe construction and stored-state details in Appendix C and implementation details in Appendix D. Appendix E provides additional experiments on request-order sensitivity and unlearning persistence, while Appendix F reports memory and runtime costs. Finally, Appendix G provides a local analysis of the proposed reversal regularization.

A PROMPTS FOR FIGURE 1

The following prompts were used to generate the samples shown in Figure 1:

- Aaron Paul to Play Luke Skywalker at LACMA Reading of The Empire Strikes Back
- Video: Charlize Theron in Trailer for New Gillian Flynn Adaptation, Dark Places
- Bloodborne Video: Sony Explains the Game's Procedurally Generated Dungeons
- Mothers influence on her young hippo

B ALGORITHM OVERVIEW

Algorithm 1 Continual data unlearning: overall procedure

Require: θ: pretrained denoiser parameters, $\mathcal{X}$: original training data
Require: N: steps per request, Q: probes per target, K: maximum older records
Probe: a fixed noisy input, its timestep, and optional conditioning.
Record: probes, response-change directions, post-deletion references, and a check-frequency weight.
Reversal: backward movement from a post-deletion reference along its recorded direction.

retained $\leftarrow \mathcal{X}$
newest $\leftarrow \varnothing$ — *record of the most recent deletion*
bank $\leftarrow \varnothing$ — *older-record memory with capacity K*
for each incoming deletion target a **do**
  retained $\leftarrow$ retained $\setminus \{a\}$
  probes $\leftarrow$ Q fixed probes constructed from a
  before $\leftarrow$ denoiser outputs under θ on probes

  for N optimizer steps **do**
    base $\leftarrow$ ReTrack gradient for $(\theta, a,$ retained$)$
    correction $\leftarrow$ CORRECTREVERSAL$(\theta,$ newest, bank, base$)$
    Bounded penalty gradient, or zero when no reversal is detected.
    $\theta \leftarrow \mathrm{OptimizerStep}(\theta,$ base $+$ correction$)$
  end for

  after $\leftarrow$ denoiser outputs under updated θ on the same probes
  record $\leftarrow$ RECORDTRANSITION(probes, before, after)
  Move newest into bank, if nonempty
  bank $\leftarrow$ UPDATEMEMORY$(\theta,$ bank$, K)$ — *retain up to K representative records*
  newest $\leftarrow$ record — *checked first during the next request*
  Discard raw target a, before, and after
end for
return θ — *model after all deletion requests*

Algorithm 2 Record the response transition after a deletion

Require: $w_0 > 0$: initial check weight
$\mathrm{Project}(y, d) = \langle y, d\rangle / \|d\|_2$, for response y and nonzero change d.

function RECORDTRANSITION(probes, before, after)
  record $\leftarrow \varnothing$
  for each probe $\in$ probes with a nonzero response change **do**
    direction $\leftarrow$ after[probe] $-$ before[probe]
    reference $\leftarrow \mathrm{Project}$(after[probe], direction)
    Append (probe, direction, reference) to record
  end for
  return record with check weight w_0 *future check frequency*
end function

Algorithm 3 Detect reversal and compute a bounded correction

Require: $\omega \geq 0$: correction scale, $\rho > 0$: norm-cap ratio
Φ_θ: model response, θ: model parameters, g: base gradient.
$\mathrm{Project}(y, d) = \langle y, d\rangle / \|d\|_2$ for nonzero d.

function CORRECTREVERSAL(θ, newest, bank, g)
  for record: newest, then one weight-scheduled record from bank **do**
    Skip missing or empty records
    (probe, direction, reference) $\leftarrow$ next cyclic entry of record
    current $\leftarrow \mathrm{Project}(\Phi_\theta(\text{probe}), \text{direction})$
    reversal $\leftarrow \max(\text{reference} - \text{current}, 0)$
    if reversal > 0 **then**
      $h \leftarrow \omega \nabla_\theta(\text{reversal}^2)$
      Rescale h if needed to have norm at most $\rho\|g\|_2$
      return h
    end if
  end for
  return 0
end function

Algorithm 4 Select representative records and redistribute check weights

θ: model parameters, K: maximum number of older records.

function UPDATEMEMORY(θ, bank, K)
  if |bank| $\leq K$ **then**
    return bank
  end if
  for each record in bank **do**
    gradients $\leftarrow$ gradients of all current probe projections w.r.t. θ
    signature $\leftarrow$ concatenate compressed, normalized gradients in probe order
  end for
  for each record in bank **do**
    Fit its signature by a weighted average of the other signatures
  end for
  Discard the record with the smallest squared fitting error
  Redistribute its check weight using its fitted averaging weights
  return bank
end function

C Probe Construction and Storage

Response construction. Each probe is constructed from the deletion target using a fixed diffusion timestep, noise realization, and, when applicable, conditioning. Writing h for the identity in pixel-space diffusion and the fixed VAE encoding in latent diffusion,

$$z_s(x, \epsilon) = \gamma_s h(x) + \sigma_s \epsilon, \qquad \Phi(\theta, \xi) = \epsilon_\theta(z_s^{\mathrm{tar}}, s, c).$$

Here γ_s and σ_s are the forward-process signal and noise coefficients, and conditioning c is omitted for unconditional models. The noisy input, timestep, and conditioning remain fixed across pre-deletion, post-deletion, and subsequent evaluations. Thus, changes in the recorded response arise from model updates rather than newly sampled probe inputs.

Probe sampling. We use $Q = 4$ probes per record. For CIFAR-10 and CelebA-HQ, timesteps are sampled uniformly from $\{200, \ldots, 999\}$, with an independently sampled Gaussian noise realization for each probe. Stable Diffusion uses timesteps $\{225, 375, 625, 875\}$ and the original target prompt, with a fixed Gaussian noise realization at each timestep. These training probes are distinct from the timestep-250 reconstructions and prompt-conditioned samples used to evaluate unlearning.

Persistent state. The pixel-space implementations store the noisy target inputs, their timesteps, the displacement d, and the scalar post-deletion projection τ for each probe. CIFAR-10 retains these floating-point tensors in FP32. CelebA-HQ stores noisy inputs and displacements in FP16, with projections computed in FP32 using the stored displacements. The separate pre- and post-deletion responses are not needed for later checks. Stable Diffusion instead stores the noisy target latents, text-conditioning embeddings, timesteps, pre-deletion responses y^-, and displacements d in FP32.

Each retained record also carries a service weight and a cyclic probe cursor. The newest record is kept separately for one request before entering memory selection, giving at most $K_{\mathrm{mem}} + 1$ records between requests. Full parameter gradients and their sketches are temporary computations, not persistent historical records.

D IMPLEMENTATION DETAILS

Table 6: Training and correction hyperparameters. K_{mem} counts older records and excludes the separately stored newest record. Memory-capacity and penalty variants override the corresponding entries.

Parameter	CIFAR-10	CelebA-HQ	Stable Diffusion v1.4
Updates per request	60	40	35
Learning rate	2×10^{-5}	5×10^{-6}	10^{-5}
AdamW (β_1, β_2)	$(0.95, 0.999)$	$(0.95, 0.999)$	$(0.9, 0.999)$
Weight decay	10^{-6}	10^{-6}	10^{-2}
AdamW ϵ	10^{-8}	10^{-8}	10^{-8}
Probes Q	4	4	4
Memory capacity K_{mem}	4	4	2
Correction scale ω	1	1	0.5
Relative norm cap ρ	0.20	0.10	0.02

Optimization. Table 6 summarizes the ReTrack-based training and correction settings. We update the denoiser in FP32 and carry its parameters across deletion requests, while reinitializing AdamW for each request. Learning rates are constant, with no warm-up. For Stable Diffusion, the entire UNet is updated while the VAE and text encoder remain fixed. The correction is added to the base gradient before the optimizer step. Its norm is bounded relative to the base gradient, rather than the parameter update produced by AdamW.

Base-update settings. CIFAR-10 uses batches of 128 with one gradient-accumulation step. The CelebA-HQ ReTrack-based runs use batches of four with two accumulation steps. SISS uses batches of four with four accumulation steps. ReTrack uses ten neighbors, with objective coefficient 0.02 on CIFAR-10, 0.005 on CelebA-HQ, and 0.5 on Stable Diffusion. SISS uses mixture coefficient 0.5 and rescaling factors 5 and 500 on CIFAR-10 and CelebA-HQ, respectively. Previously deleted targets are excluded from positive supervision throughout each sequence.

Gradient signatures and selection. We use two CountSketch-style repeats of dimension 8,192 with sketch seed 27182818, concatenated and scaled by $1/\sqrt{2}$. Both repeats use a shared parameter ordering and bucket assignment across records, with independent random signs. Each probe sketch is normalized separately. The four normalized sketches are concatenated and scaled by $1/\sqrt{Q}$. The implementation differentiates normalized progress $p = 1 + m/\|d\|_2$. Its gradient is a positive scalar multiple of the margin gradient, so the normalized sketch direction is unchanged.

The candidate pool contains at most $K_{\mathrm{mem}} + 1$ records. We enumerate subsets of the required size and solve the simplex-constrained least-squares fit for each candidate by enumerating active coefficient sets. The inner solves use FP64 and a feasibility tolerance of 10^{-8}. Discarded records transfer their service weights according to the fitted coefficients. Largest-remainder allocation, with at least one scheduled visit per retained record, converts these weights into a finite checking schedule. Newest-record checks retain priority over this older-record schedule.

E Additional Experiments

E.1 Sensitivity to Request Order

Table 7: Sensitivity to request order on CelebA-HQ and Stable Diffusion v1.4. For each setting, we evaluate five different deletion orders using a fixed target set: 50 targets for CelebA-HQ and 30 targets for Stable Diffusion. We report the mean and standard deviation across these orders. Our method uses $K_{\mathrm{mem}} = 2$ in both settings.

	CelebA-HQ		Stable Diffusion v1.4			
Method	SSCD ↓	FID ↓	Success (%) ↑	CLIP-IQA ↑	SSCD ↓	CLIP score ↑
SISS	0.6985 ±0.0256	**25.06** ±4.78	74.67 ±14.64	0.6782 ±0.0574	0.1611 ±0.0218	26.2896 ±0.8763
ReTrack	0.4292 ±0.0228	29.01 ±2.43	85.33 ±9.31	0.7942 ±0.0346	0.2286 ±0.0163	26.7059 ±0.7214
Ours	**0.4115** ±0.0187	28.90 ±3.24	**90.67** ±2.79	**0.7969** ±0.0171	**0.1545** ±0.0250	**26.7712** ±0.5602

We assess sensitivity to request order by varying the deletion sequence while keeping the target set fixed (Table 7). On CelebA-HQ, our method achieves the lowest SSCD with the smallest standard deviation, while also attaining a slightly lower FID than ReTrack. On Stable Diffusion v1.4, it achieves the highest success rate with substantially lower variability, reaching $(90.67 \pm 2.79)\%$ compared with $(85.33 \pm 9.31)\%$ for ReTrack and $(74.67 \pm 14.64)\%$ for SISS. It also achieves lower SSCD and higher CLIP-IQA and CLIP scores than both baselines, with smaller standard deviations in the two CLIP-based metrics. Overall, our method improves both SSCD on CelebA-HQ and unlearning success on Stable Diffusion v1.4, while reducing their variability across deletion orders and maintaining competitive generation quality.

E.2 Persistence of Unlearning

Table 8: SSCD rebound on CelebA-HQ over three runs of 50 deletions. Ours uses $K_{\mathrm{mem}} = 4$. The lower panel requires immediate SSCD ≤ 0.5 under all methods.

	SSCD ↓		
Method	Immediately after deletion	After 50 deletions	Mean rebound
All evaluated targets			
SISS	0.6069 ±0.0044	0.6814 ±0.0093	**0.0746** ±0.0136
ReTrack	0.3631 ±0.0237	0.4810 ±0.0091	0.1210 ±0.0151
Ours	**0.3529** ±0.0153	**0.4408** ±0.0326	0.1001 ±0.0325
Common targets with low post-deletion SSCD			
SISS	0.3913 ±0.0293	0.5232 ±0.0731	0.1319 ±0.0438
ReTrack	**0.1831** ±0.0245	0.3094 ±0.0409	0.1269 ±0.0190
Ours	0.2127 ±0.0097	**0.3009** ±0.0152	**0.0919** ±0.0182

We measure persistence by averaging positive SSCD increases from immediately after deletion to the final model under matched evaluation conditions, counting decreases as zero (Table 8). Our method reduces rebound relative to ReTrack ($0.1210 \rightarrow 0.1001$) while achieving lower immediate and final SSCD. SISS has the smallest rebound but much higher SSCD at both time points; low rebound alone therefore does not imply effective unlearning.

A small rebound can also reflect weak initial unlearning, as illustrated by SISS's high immediate SSCD. We therefore additionally compare common targets with immediate SSCD ≤ 0.5 under all methods. On this subset, our method reduces rebound by 27.6% relative to ReTrack and achieves lower final SSCD despite starting from higher post-deletion similarity. This supports improved persistence beyond stronger initial suppression on the evaluated subset.

F MEMORY AND RUNTIME

We profile ReTrack and our method on the first ten requests of three CelebA-HQ deletion sequences, using 40 optimizer steps per request, $Q = 4$, and $\rho = 0.20$. Each configuration starts from the same pretrained model and maintains its own evolving parameters and records. Measurements use a Blackwell GPU, 12 CPUs, four computation threads, and FP32 with TF32 disabled. All runs use the same GPU model.

Table 9 summarizes requests 7–10, when all memory capacities are filled. We first average within each sequence and then report the mean and sample standard deviation across three sequences. Request time includes training, neighbor search, process startup, model loading, checkpoint I/O, transition recording, and memory selection, but excludes evaluation. GPU memory is the mean of per-request peak PyTorch allocations during training, including optimizer and correction state. Memory used outside the training steps is not included.

Table 9: CelebA-HQ resource costs across three deletion sequences.

Method	Time/request (s)	Relative time	Training peak (MiB)
ReTrack	224.47 ± 19.38	$1.00\times$	21317.48 ± 0.00
Ours ($K_{\mathrm{mem}} = 2$)	277.42 ± 16.23	$1.24\times$	21331.51 ± 1.16
Ours ($K_{\mathrm{mem}} = 3$)	272.15 ± 6.97	$1.22\times$	21331.90 ± 0.67
Ours ($K_{\mathrm{mem}} = 4$)	285.41 ± 5.37	$1.28\times$	21332.28 ± 0.67

Relative time is the mean ratio to ReTrack over matched deletion sequences. The measured request-time overhead is approximately 22–28%, while training peak allocation increases by approximately 14–15 MiB. Timing includes startup and I/O costs, and no unusually slow training steps are excluded. Runtime does not increase monotonically with capacity in these measurements, so the small differences between capacities should not be interpreted as a consistent speed advantage.

G Local Analysis of Reversal Regularization

Notation. Fix one probe and its recorded quantities $d \neq 0$ and y^+. Omitting request and probe indices, write $y(\theta)$ for the response and define

$$\hat{d} = \frac{d}{\|d\|_2}, \qquad m(\theta) = \langle y(\theta) - y^+, \hat{d} \rangle, \qquad G(\theta) = [-m(\theta)]_+^2.$$

Proposition 1 (Response-space distance)**.** *Let $\mathcal{H} = \{y : \langle y - y^+, \hat{d} \rangle \geq 0\}$. Then*

$$G(\theta) = \operatorname{dist}(y(\theta), \mathcal{H})^2, \qquad \Pi_{\mathcal{H}}(y(\theta)) = y(\theta) + [-m(\theta)]_+ \hat{d}, \tag{13}$$

where $\Pi_{\mathcal{H}}$ is Euclidean projection onto $\mathcal{H}$.

Proof. If $m(\theta) \geq 0$, then $y(\theta) \in \mathcal{H}$ and both distances vanish. Otherwise, any feasible displacement δ satisfies $\langle \delta, \hat{d} \rangle \geq -m(\theta)$. Cauchy–Schwarz gives $\|\delta\|_2 \geq -m(\theta)$, with equality attained by $\delta = -m(\theta)\hat{d}$. Squaring this minimum distance proves the claim. □

Proposition 2 (Local margin effect of a clipped correction)**.** *Suppose $m(\theta) < 0$, $\omega, \rho \geq 0$, and ∇m is L-Lipschitz on a convex neighborhood containing the two update segments below. Set $g = g^{\text{base}}$ and*

$$h = \operatorname{clip}_{\rho\|g\|_2}(\omega \nabla G(\theta)),$$

where clip *denotes Euclidean norm clipping. There exists $\lambda \in [0, \omega]$ with $h = \lambda \nabla G(\theta)$. For $\eta > 0$, define the ordinary gradient-descent steps $\theta^+ = \theta - \eta(g + h)$ and $\theta^0 = \theta - \eta g$. Then*

$$m(\theta^+) - m(\theta^0) = 2\eta\lambda[-m(\theta)]\|\nabla m(\theta)\|_2^2 + r_\eta, \tag{14}$$

with

$$|r_\eta| \leq \frac{L\eta^2}{2} \left(\|g + h\|_2^2 + \|g\|_2^2 \right). \tag{15}$$

Proof. Euclidean norm clipping rescales its input by a factor in $[0, 1]$, giving the stated λ. Since $m(\theta) < 0$, $\nabla G(\theta) = 2m(\theta)\nabla m(\theta)$. Lipschitz continuity of the gradient implies

$$m(\theta - \eta v) = m(\theta) - \eta \langle \nabla m(\theta), v \rangle + R_v, \qquad |R_v| \leq \tfrac{L\eta^2}{2} \|v\|_2^2.$$

Subtract the expansions for $v = g + h$ and $v = g$. The linear term is $-\eta\langle \nabla m, h \rangle = 2\eta\lambda[-m]\|\nabla m\|_2^2$. The triangle inequality bounds $r_\eta = R_{g+h} - R_g$ as stated. □

Scope. Proposition 2 gives a nonnegative first-order contribution relative to the base step. Proposition 1 characterizes response-space geometry.