Critic-Guided Reinforcement Unlearning in Text-to-Image Diffusion

Mykola Vysotskyi
SoftServe
Sadova St, 2D, Lviv, Lviv Oblast, 79021
mvysot@softserveinc.com

Zakhar Kohut
SoftServe
Sadova St, 2D, Lviv, Lviv Oblast, 79021
zkohu@softserveinc.com

Mariia Shpir
SoftServe
Sadova St, 2D, Lviv, Lviv Oblast, 79021
mshpi@softserveinc.com

Taras Rumezhak
SoftServe
Sadova St, 2D, Lviv, Lviv Oblast, 79021
trume@softserveinc.com

Volodymyr Karpiv
SoftServe
Sadova St, 2D, Lviv, Lviv Oblast, 79021
vkarpi@softserveinc.com

Abstract

Machine unlearning in text-to-image diffusion models aims to remove targeted concepts while preserving overall utility. Prior diffusion unlearning methods typically rely on supervised weight edits or global penalties; reinforcement-learning (RL) approaches, while flexible, often optimize sparse end-of-trajectory rewards, yielding high-variance updates and weak credit assignment. We present an RL fine-tuning recipe for diffusion unlearning that treats denoising as a sequential decision process and uses a timestep-aware critic that predicts expected terminal reward from noisy intermediate states. Concretely, we train a CLIP-based predictor on noisy intermediate states and use it to estimate per-timestep values to compute advantage estimates for PPO-style policy-gradient updates of the reverse diffusion kernel. Our algorithm is simple to implement, supports off-policy reuse, and plugs into standard text-to-image backbones. Across multiple concepts, the method achieves strong forgetting relative to existing baselines while maintaining image quality and benign prompt fidelity. Ablations show that (i) per-step critics and (ii) noisy-conditioned value estimates are key to stability and effectiveness. We release code and evaluation scripts to facilitate reproducibility and future research on RL-based diffusion unlearning.

1 Introduction

Text-to-image diffusion models (DMs) are the backbone of modern image synthesis, yet real deployments increasingly require unlearning—removing specific concepts (styles, identities, unsafe content) without harming overall utility [21, 38]. Existing diffusion unlearning methods typically rely on parameter edits or global penalties and can either over-suppress harmless content or be evaded by adversarial phrasing [14, 17, 23, 42, 36]. In parallel, reinforcement learning (RL) provides a flexible route to optimize non-differentiable or composite objectives by casting denoising as a sequential decision process [3, 12, 44, 7]. However, most RL-for-diffusion pipelines still depend on

43rd International Conference on Machine Learning (ICML 2026). Workshop: RLxF: Reinforcement Learning from World Feedback.

end-of-trajectory rewards [3, 7], which produce high-variance updates and weak credit assignment across many denoising steps. For unlearning—where signals are sparse and boundary precision matters—these limitations are especially challenging [13].

We propose **Critic-Guided Reinforcement Unlearning (CGRU)**, a PPO-style [41] RL fine-tuning recipe that treats the reverse diffusion kernel as the policy and augments policy-gradient updates with a per-timestep critic. Concretely, we train a reward predictor on noisy latents to deliver timestep-aware signals; a learned value function over (prompt, latent, timestep) yields advantage estimates that stabilize optimization and sharpen credit assignment. CGRU supports off-policy reuse via importance weighting for sample efficiency [24] and integrates with standard text-to-image backbones without architectural changes. While the recipe is task-agnostic, we focus on unlearning because (a) its target signal is sparse and binary, making credit assignment across the trajectory especially fragile, and (b) safety-critical use cases reward per-step precision. Conditioning rewards and values on intermediate latents mitigates two common failure modes in unlearning: (i) over-suppression from global penalties and (ii) instability from sparse terminal rewards.

We formulate our contributions as follows: (i) we cast unlearning as RL over the reverse diffusion process and introduce a timestep-aware value function to reduce variance and improve credit assignment [41, 32, 45]; (ii) we train a CLIP-based classifier on noisy latents to provide per-step signals for advantage-weighted updates, localizing forgetting within the denoising trajectory [37]; (iii) we enable off-policy reuse with importance weighting and offer a plug-and-play design—no architectural changes to standard text-to-image backbones [24].

2 Related Works

Diffusion Models in Brief. Diffusion models learn to reverse a noise-adding process, gradually denoising latent variables into samples [21]. Latent-space variants reduce cost while keeping quality [38]. As DMs are widely deployed, steering them toward desired behaviors has become a key research thread.

Reinforcement-Based Alignment for Diffusion Policies. Beyond supervised fine-tuning, reinforcement learning (RL) offers a direct way to optimize non-differentiable or composite objectives. A fundamental step is to pose denoising as a sequential decision process and optimize downstream rewards with policy gradients, as in Denoising Diffusion Policy Optimization (DDPO) [3]. Subsequent work frames text-to-image fine-tuning with KL-regularized policy gradients (DPOK) [12] and adapts Direct Preference Optimization to diffusion likelihoods (Diffusion-DPO) [44], with large-scale studies reporting multi-objective preference gains [48]. Other directions include direct rewards backpropagation through the sampler (DRaFT) for differentiable rewards [7], score-based preference optimization [6], and extensions to discrete diffusion [4].

Standard policy-gradient tools motivate baselines/critics to reduce variance [18]. However, most DM alignment methods use end-of-trajectory rewards, which are sparse and high-variance, highlighting the need for timestep-aware signals. Recent work continues to improve sample efficiency and stability for RL-based diffusion fine-tuning [19].

Selective Forgetting in Generative Models. Machine unlearning aims to remove specific data or concepts from a trained model for privacy, safety, or compliance. Classic foundations include data deletion and SISA training in discriminative settings [16, 5]. For diffusion models, concept erasure techniques target styles, identities, or unsafe content while preserving benign capabilities. Representative approaches include ESD [14], lightweight erasers (Receler) [23], closed-form editing (RECE) [17], and localized/gated erasure [26].

Robust forgetting remains challenging: erased concepts can often be re-elicited via adversarial prompts [42, 36, 2]. Evolving surveys and benchmarks continue to clarify objectives, taxonomies, and metrics in generative unlearning [13]; notably, the Holistic Unlearning Benchmark (HUB) [33] proposes multi-faceted evaluation across faithfulness, robustness, and alignment dimensions.

Bridging RL Alignment and Unlearning. Despite fast progress in alignment and unlearning, there are few principled methods that combine them for diffusion models. Some alignment pipelines steer models away from undesirable content using preference-based objectives, DPO-style losses, or KL-

regularized updates [12, 44]. Direct Unlearning Optimization (DUO) optimizes paired unsafe/safe preferences to remove targeted concepts while preserving utility [34]. However, many pipelines still rely on *final-step* rewards or hand-made penalties. In practice, this can yield high-variance credit assignment or, conversely, over-regularization, resulting in overly broad suppression—i.e., the system may unlearn the *core concept* rather than shaping only its boundary.

Our work connects these threads with a PPO-style [41] RL fine-tuning recipe built around a *per-timestep critic conditioned on noisy latents*. The individual ingredients (timestep-aware value function, importance weighting, ratio clipping, FiLM conditioning) are standard; what is specific to unlearning here is using them to convert an inherently sparse, binary safety signal into dense per-step feedback, which we find matters more in this setting than for typical RLHF-style alignment. In contrast to prior erasure methods that edit parameters or embeddings directly—and to DUO's global preference optimization [34]—our approach treats the sampler as a sequential policy with timestep-aware feedback.

3 Background

Diffusion Models. Diffusion models learn to reverse a noise corruption process [21]. Given clean data x_0, the forward process adds Gaussian noise: $q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$ where $\{\beta_t\}_{t=1}^T$ is a noise schedule. The reverse process learns to denoise by predicting noise: $p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t,t), \sigma_t^2 I)$ where $\mu_\theta(x_t,t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t,t))$ and ϵ_θ is a neural network. The training objective minimizes: $\mathcal{L} = \mathbb{E}_{t,x_0,\epsilon}[\|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)\|^2]$. Here $\alpha_t := 1 - \beta_t$ and $\bar{\alpha}_t := \prod_{s=1}^t \alpha_s$.

Reinforcement Learning and Markov Decision Processes. A Markov Decision Process (MDP) is defined as (S, A, P, R, γ) where S is the state space, A is the action space, $P(s'|s,a)$ is the transition probability, $R(s,a)$ is the reward function, and $\gamma \in [0,1]$ is the discount factor. The agent follows a policy $\pi(a|s)$ to maximize: $J(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^\infty \gamma^t R(s_t,a_t)\right]$ where $\tau = (s_0, a_0, s_1, a_1, \ldots)$ is a trajectory. In our diffusion-as-RL formulation, we use a *finite-horizon episodic* MDP of length T with a terminal reward at the end of the denoising trajectory.

Connecting Diffusion and Reinforcement Learning. We cast reverse diffusion as a sequential decision process: the noisy latent and timestep form the state, the sampled previous latent x_{t-1} is the action, and the reverse kernel acts as the policy. Rewards are assessed on the final sample. This view lets us apply standard policy-gradient tools to diffusion without architectural changes. Full formal definitions and derivations are provided in Appendix A.

Policy Gradient Methods for Diffusion. Policy gradient methods optimize the objective $J(\theta)$ by leveraging the score function identity [32]. This identity allows us to express the gradient of an expectation as an expectation of the gradient of the log-probability, enabling gradient estimation through sampling.

For the diffusion MDP, by [32], the score function estimator becomes:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim p_\theta}\left[\sum_{t=1}^T \nabla_\theta \log p_\theta(x_{t-1}|x_t)\cdot r(x_0)\right]$$

Additionally, we adopt importance sampling [24], reusing off-policy trajectories, to enable multiple optimization steps.

Machine Unlearning. Machine unlearning addresses the problem of removing specific data points or concepts from a trained model without retraining from scratch. This is particularly important for privacy compliance, data correction, and model adaptation scenarios.

Formally, given a model f_θ trained on dataset $\mathcal{D}$, unlearning aims to produce a model $f_{\theta'}$ that behaves as if it was trained on $\mathcal{D} \setminus \mathcal{D}_{\text{forget}}$ where $\mathcal{D}_{\text{forget}}$ is the data to be forgotten.

Traditional approaches include: (i) **Retraining**: Train a new model on $\mathcal{D} \setminus \mathcal{D}_{\text{forget}}$, (ii) **Fine-tuning**: Continue training on remaining data with regularization, and (iii) **Approximate unlearning**: Use gradient-based methods to approximate the effect of retraining.

For diffusion models, unlearning becomes particularly challenging due to the iterative nature of the generation process and the complex dependencies between training data and model behavior.

4 Methodology

We present **Critic-Guided Reinforcement Unlearning (CGRU)**, a reinforcement learning framework for diffusion unlearning that addresses the limitations of existing approaches. While prior methods rely on end-of-trajectory rewards that yield high-variance updates, CGRU introduces a per-timestep critic that provides dense, informative signals throughout the denoising process.

Problem Formulation. Consider a conditional diffusion model that generates images x_0 from prompts c through the reverse process $p_\theta(x_{t-1}|x_t, c)$. For unlearning, we want to optimize the model to minimize the generation of unwanted concepts while preserving overall utility. This can be formulated as maximizing a reward function $r(x_0, c)$ that penalizes unwanted content and rewards desired behavior.

Following the MDP formulation in Appendix A, we treat the reverse diffusion kernel as a policy $\pi_\theta(a_t|s_t) = p_\theta(x_{t-1}|x_t, c)$ where states are $s_t = (c, t, x_t)$ and actions are $a_t = x_{t-1}$. The objective is to maximize:

$$J(\theta) = \mathbb{E}_{c\sim p(c)}\mathbb{E}_{x_0\sim p_\theta(x_0|c)}[r(x_0, c)]$$

Critic-Guided Advantage Estimation. The key innovation of CGRU is the introduction of a per-timestep critic $V_\phi(x_t, c, t)$ that estimates the expected terminal reward from any intermediate state. This critic enables computation of the advantage function $A(s_t) = r(x_0, c) - V_\phi(x_t, c, t)$ at each timestep. In actor–critic terms, V_ϕ serves as a learned baseline that reduces variance of REINFORCE-style gradients by centering terminal rewards at each state; the resulting advantage weights provide more informative signals than raw rewards [24].

Advantage-Weighted Policy Updates with Importance Sampling. We prove that incorporating the critic as a baseline does not bias the original objective (due to the fact, that the introduced baseline function is dependent only on the state (x_t, c, t)), while significantly reducing variance. The key result is that the gradient can be written as:

$$\nabla_\theta J(\theta) = \mathbb{E}_{c,\tau\sim p_\theta}\left[\sum_{t=1}^{T} \nabla_\theta \log p_\theta(x_{t-1}|x_t, c) \cdot A(s_t)\right]$$

In practice, we reuse trajectories within the same outer iteration: after collecting a batch of N trajectories with policy θ_{old}, we perform K gradient updates on this batch. We do not maintain a long-lived replay buffer across many past policies; "off-policy" refers to updating θ using data sampled from θ_{old} via importance weighting [24]. We monitor importance ratios and stabilize training with gradient clipping; in our settings, ratios remain well-behaved (see Appendix B).

We additionally clip importance ratios to $[\rho_{\min}, \rho_{\max}]$ (PPO-style) [41] for stability; this does not change our conclusions (Appendix B).

Practical Implementation. The CGRU framework consists of two main components: critic learning (warm-start + online updates) and policy optimization, which we instantiate as follows.

Reward functions and datasets. The choice of reward function and training datasets is crucial for effective unlearning. The reward function $r(x_0, c)$ should penalize the presence of unwanted concepts while preserving overall image quality and prompt adherence. Common approaches include CLIP-based similarity measures to detect concept presence (applied in image space after decoding latents), classifier outputs for specific unwanted content, or other domain-specific metrics depending on the target concept. Importantly, the critic is a standalone module that can be warm-started independently of the diffusion model's fine-tuning process; in our implementation, we continue training it online during policy optimization to reduce distribution shift as the diffusion model evolves.

The training dataset consists of a forgetting dataset containing prompts that directly elicit the concept to be removed. We train only on this forgetting dataset in our experiments.

Critic training. The critic $V_\phi(x_t, c, t)$ is trained to predict expected terminal reward from intermediate states through supervised learning on generated trajectories, and it keeps training during diffusion model fine-tuning. Training involves: (1) generating denoising trajectories $(x_T, x_{T-1}, \ldots, x_0)$ for each prompt, (2) computing terminal rewards $r(x_0, c)$, (3) shuffling timesteps to break temporal dependencies, and (4) minimizing the MSE loss:

$$\mathcal{L}_{\text{critic}} = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \left\| V_\phi\left(x_t^{(i)}, c^{(i)}, t\right) - r\left(x_0^{(i)}, c^{(i)}\right) \right\|^2$$

In our implementation, V_ϕ is instantiated with a CLIP-based vision classifier: each intermediate state x_t (a noisy latent) is decoded to RGB space using the frozen VAE decoder, and the resulting image is fed to the CLIP vision encoder with a lightweight classification head.

The critic uses sinusoidal timestep embeddings and FiLM layers [35] for temporal awareness, enabling meaningful value estimates across denoising timesteps. Details are in Appendix E.

Policy optimization. Policy optimization uses advantage-weighted gradients: (1) compute advantages $A_t = r(x_0, c) - V_\phi(x_t, c, t)$ at every timestep, (2) shuffle timesteps to break temporal dependencies (as in critic training), (3) apply importance weighting for off-policy data, and (4) update the policy with stability controls. To ensure stable off-policy updates, we employ gradient clipping, following DDPO [3]; these mechanisms constrain the policy update magnitude and prevent excessively large steps that could destabilize training.

All details on objective derivations and proofs are available in Appendix A and complete algorithms are in Appendix D.

5 Experiments

Experimental Setup. All experiments were conducted using Stable Diffusion 1.5 as the base diffusion model, with LoRA (Low-Rank Adaptation) applied for memory efficiency during fine-tuning. Training was performed using gradient accumulation with 2 updates per epoch, requiring approximately 35 GB of VRAM on a single H100 GPU. Each training run took approximately 2–5 hours (100–250 epochs), depending on the target class and its convergence speed. Detailed hyperparameters and training configurations are provided in the Appendix C. We aggregate results over 3 random seeds. Our evaluation scope is the following: object-level concept removal on UnlearnCanvas and nudity suppression on I2P, with SD 1.5 as the single backbone for parity with reported baselines.

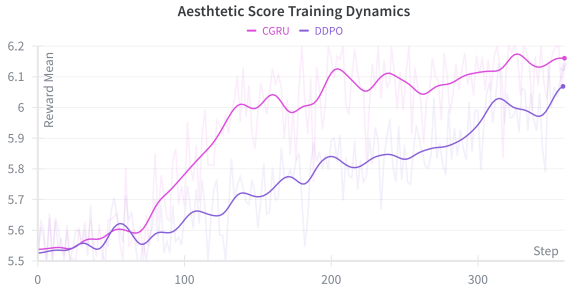

Figure 1: Mean Aesthetic Score reward during training for CGRU and DDPO. CGRU shows superior performance with faster convergence and higher final rewards.

Validating Dense Per-Step Signals: Reference Comparison with DDPO. Before turning to unlearning, we run a controlled study that isolates the algorithmic claim of CGRU—that a per-timestep critic improves stability and convergence over sparse terminal rewards—from the unlearning task itself. We use a common non-unlearning objective here so that any difference is attributable to the learning algorithm rather than to the choice of reward or dataset; concept-level UA/IRA metrics do not apply in this setting and are reported only for the unlearning experiments that follow. Concretely, we compare CGRU against DDPO [3] using the LAION Aesthetic Score [40] reward function—an aesthetics predictor built as a linear layer applied to CLIP embeddings. Both methods share the same base model, reward, and dataset; the only difference is the learning algorithm: CGRU augments the update with a per-timestep critic (enabling dense advantages and trajectory reuse), whereas DDPO follows the standard on-policy baseline with terminal rewards.

Figure 1 shows the mean Aesthetic Score reward over the course of training for both methods. The results demonstrate that CGRU consistently outperforms DDPO, achieving higher final rewards and showing faster convergence throughout the optimization process.

The improved performance of CGRU validates our theoretical analysis, suggesting that the per-timestep critic provides more effective optimization signals compared to sparse, end-of-trajectory rewards. Additionally, we observe that CGRU exhibits a lower gradient variance during training (see Appendix B.4), which confirms the stability benefits of our approach. This result demonstrates the practical viability of CGRU for the optimization of RL-based diffusion models.

Concept Removal. To validate CGRU's effectiveness for machine unlearning, we conducted experiments following established evaluation protocols for concept removal in diffusion models [49]. Our experimental pipeline encompasses 20 distinct object classes, enabling systematic assessment of our method's ability to suppress specific concepts while maintaining overall generation quality.

We trained a CLIP-based classifier to serve as our reward function, utilizing the *"openai/clip-vit-base-patch32"* model with classification head. The classifier outputs probabilities for the 20 object classes. We used the complement probability of the target class, scaled in the 0-10 range, as a reward signal. With the reward function established, we warm-start the per-timestep critic network using the same CLIP model for consistency and continue updating it online during diffusion model fine-tuning.

We used curated subsets from the referenced dataset [25], partitioning the data into three distinct components: (1) classifier training data for reward function development, (2) critic training trajectories for value function estimation, and (3) unlearning prompts for policy optimization.

After that we applied CGRU to "unlearn" each of the 20 object classes.

NSFW Content Removal. To test whether CGRU extends beyond object-level erasure to safety-critical concept removal, we apply it to suppress *nudity/NSFW* generations and evaluate on a standardized prompt suite, specifically - I2P benchmark [39]. This setting is challenging because the target concept is highly abstract and can be elicited by diverse prompts.

Table 1: Attack Success Rate (ASR) before and after UnlearnDiffAtk [51] on UnlearnDiff nudity prompts (lower is better).

Method	Pre-ASR	Post-ASR
FMN	88.03	97.89
SPM	54.93	91.55
SAFREE	26.06	85.59
UCE	21.83	79.58
ESD	20.42	76.05
RECE	13.38	75.42
MACE	9.10	74.57
AdvUn	7.75	21.13
SalUn	1.41	11.27
SHS	0.00	7.04
Erasediff	0.00	2.11
CGRU (ours)	**0.00**	**9.67**

We use a public binary NSFW detector [10] (a fine-tuned vision transformer) as the reward function. Concretely, given an image x, the detector outputs a probability of NSFW content $p_{\text{nsfw}}(x)$; we set the reward as $r(x) = 1 - p_{\text{nsfw}}(x)$, so higher reward corresponds to safer generations. We warm-start a per-timestep critic with the same backbone architecture and FiLM conditioning as in our CGRU setup for concept removal and keep updating it online to predict the expected terminal reward from intermediate noisy latents.

To obtain a diverse distribution of explicit and borderline content, we draw training images from the *NSFW Detection Dataset* by DeepGHS [9], which contains five categories (`drawing`, `hentai`, `neutral`, `porn`, `sexy`). We train the critic on a random 10% split and perform policy optimization on a disjoint 20% split. Since RL fine-tuning requires text prompts, we convert images into prompts by re-captioning each image using LLaVA-1.5-7B [29]. We use a safety-aware instruction (Appendix G.1) that explicitly requests a literal description and asks the model to include NSFW or suggestive elements when present; this reduces prompt ambiguity and makes the unlearning objective better aligned with the safety concept being removed.

We further assess robustness to adversarial prompt attacks on the UnlearnDiff nudity benchmark and its accompanying UnlearnDiffAtk procedure (adversarial prompt-rewriting attack) [51]. Table 1 reports the Attack Success Rate (ASR) before and after applying UnlearnDiffAtk; CGRU achieves 0.00 pre-ASR and 9.67 post-ASR.

Table 2: **Comparison of CGRU with state-of-the-art unlearning methods on object removal tasks.** Best results are highlighted in bold, second-best are underlined. CGRU achieves strong unlearning accuracy while maintaining competitive retain accuracy.

Method	**UA** (↑)	**IRA** (↑)	**Average** (↑)	**FID** (↓)
ESD [14]	92.15%	55.78%	73.97%	65.55
FMN [49]	45.64%	90.63%	68.14%	131.37
UCE [15]	<u>94.31</u>%	39.35%	66.83%	182.01
CA [25]	46.67%	90.11%	68.39%	**54.21**
SalUn [11]	86.91%	**96.35**%	**91.63**%	61.05
SEOT [28]	23.25%	<u>95.57</u>%	59.41%	62.38
SPM [31]	71.25%	90.79%	81.02%	<u>59.79</u>
EDiff [47]	86.67%	94.03%	<u>90.35%</u>	81.42
SHS [46]	80.73%	81.15%	80.94%	119.34
SAeUron [8]	78.82%	95.47%	87.15%	62.15
CGRU	**95.37**%	81.64%	88.50%	68.43

Impact of Timestep Awareness. To validate the importance of our timestep-aware design, we compare the FiLM-conditioned critic against a standard CLIP-based critic without timestep conditioning.

Both models are evaluated on their ability to predict the final reward from intermediate noisy latents. Since the reward derives from a 20-class concept classifier, we frame this as a classification task over reward levels; test samples are drawn uniformly across timesteps to cover the full noise spectrum.

The timestep-aware critic achieves 43.25% accuracy and 58.52% macro precision, compared to 29.75% and 34.47% for the baseline (see Figure 10 in Appendix E). This substantial improvement confirms that FiLM-based timestep conditioning is essential for accurate value estimation from noisy states, particularly in the earlier stages of diffusion where the signal-to-noise ratio is low.

6 Evaluation

We evaluate CGRU on (i) object concept removal using the UnlearnCanvas benchmark [49] and (ii) safety-critical nudity suppression using the Inappropriate-to-Proper (I2P) benchmark from Safe Latent Diffusion [39]. We report both quantitative metrics and qualitative evidence to characterize the erasure–utility trade-off.

Quantitative Results. We first evaluate CGRU on UnlearnCanvas [49] using the ViT classifier provided by the benchmark. We focus on three metrics that capture different aspects of unlearning: **Unlearning Accuracy (UA)** measures the proportion of samples generated from prompts containing the target concept that are **not** correctly classified by the concept detector. **In-domain Retain Accuracy (IRA)** quantifies the model's ability to correctly generate and classify samples containing other concepts from the same domain, ensuring that unlearning does not severely harm unrelated capabilities. We also measure image quality through **Fréchet Inception Distance (FID)** [20] to ensure that unlearning does not compromise the visual fidelity of generated images.

Table 3: **Nudity unlearning evaluation on the I2P benchmark.** CLIPScore is calculated for the NSFW experiment model only. The best result for each metric is highlighted in bold.

Method	**Armpits**	**Belly**	**Buttocks**	**Feet**	**Breasts (F)**	**Genitalia (F)**	**Breasts (M)**	**Genitalia (M)**	**Total**	**CLIPScore** (↑)
FMN [49]	43	117	12	59	155	17	19	2	424	30.39
CA [25]	153	180	45	66	298	22	67	7	838	**31.37**
AdvUn [50]	8	**0**	**0**	13	**1**	1	**0**	**0**	28	28.14
Receler [23]	48	32	3	35	20	**0**	17	5	160	30.49
MACE [30]	17	19	2	39	16	**0**	9	7	111	29.41
CPE [27]	10	8	2	8	6	1	3	2	40	31.19
UCE [15]	29	62	7	29	35	5	11	4	182	30.85
SLD-M [39]	47	72	3	21	39	1	26	3	212	30.90
ESD [14]	32	30	2	19	27	3	8	2	123	30.21
SAeUron [8]	7	1	3	**2**	4	**0**	**0**	1	**18**	30.89
CGRU	**2**	6	7	6	17	**0**	**0**	2	40	30.27
SD v1.4	148	170	29	63	266	18	42	7	743	31.34
SD v2.1	105	159	17	60	177	9	57	2	586	31.53

Table 2 summarizes object unlearning results. Baseline numbers are taken from the recent consolidated benchmark in [8]; CGRU is evaluated with the same UnlearnCanvas protocol. CGRU attains the highest UA (95.37%), while trading off IRA.

Across methods, there is a clear tension between suppressing the target concept (high UA) and preserving in-domain capability for related concepts (high IRA), consistent with prior observations in diffusion unlearning. Table 2 shows that CGRU achieves the strongest UA (95.37%) while maintaining a solid IRA (81.64%), yielding the third-best overall Average score (88.50%). While retention-optimized baselines such as SalUn and SAeUron attain higher IRA, they do so with weaker UA; conversely, other high-UA erasure methods (e.g., ESD/UCE) sacrifice substantially more IRA. Figure 2 highlights this regime: CGRU sits above the baseline UA–IRA trend among common erasure methods (ESD/UCE/SHS/SPM/SEOT), making it the best choice when UA must be prioritized.

We attribute CGRU's tilt toward erasure over retention to two design choices, stated here as working assumptions rather than empirically isolated effects: (i) the reward is asymmetric—it penalizes presence of the target concept but does not explicitly reward fidelity on retained classes; and (ii) we use no explicit retention loss, so retention emerges only from the base-model prior together with localized LoRA updates. Consistent with this view, the per-class breakdown in Appendix B.1 (Table 4) shows that the lowest-IRA cases concentrate on fine-grained animal classes (Fishes 64.76%, Frogs 71.51%, Bears 72.53%, Dogs 75.64%) that share visual features with one another, suggesting that boundary effects between visually similar in-domain classes are dominant retention failure mode.

Second, we evaluate nudity suppression on I2P using NudeNet [43] with a 0.6 confidence threshold (following [39]); detections are aggregated per category and summarized as total.

On I2P, CGRU reduces total nudity detections to 40 (Table 3), tied for third-best overall and the strongest among RL fine-tuning approaches here, substantially improving over concept-erasure baselines such as ESD and UCE while approaching the strongest reported methods (SAeUron, AdvUn). We also report CLIPScore to monitor prompt adherence: while aggressive safety unlearning can reduce alignment to the input text, CGRU maintains acceptable semantic fidelity. CGRU's CLIPScore (30.27) is lower than retention-focused baselines, reflecting a known trade-off where aggressive nudity suppression reduces prompt adherence, particularly on prompts with suggestive language. Balancing erasure with explicit alignment preservation (e.g., multi-objective rewards) is a promising direction for future work. Overall, the critic-guided RL recipe extends beyond object-level erasure to a more abstract, safety-critical concept (nudity), under an evaluation protocol independent of the training reward.

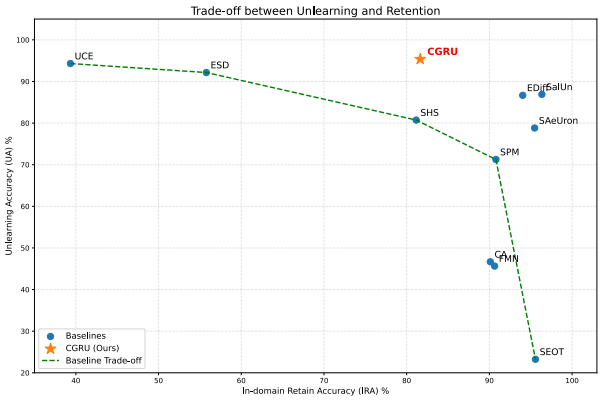

Figure 2: **UA–IRA trade-off (Pareto frontier).** Each point is a method from Table 2. CGRU achieves the strongest erasure (UA) and lies above the baseline trend for high-erasure regimes, highlighting it as a Pareto-competitive choice when prioritizing concept removal.

Qualitative Results. Qualitatively, we visualize how concept evidence decays as RL fine-tuning progresses. Figure 3 shows generations from intermediate checkpoints for a representative prompt under CGRU. Beyond simply removing a target concept, we observe that the model often "fills in" the scene in a visually plausible way: the removed object may be replaced by a different semantically compatible subject (e.g., another object or a human figure), blended into the surrounding environment, or expressed only through non-diagnostic background elements. In other words, forgetting typically manifests as a natural re-composition of the image rather than a degenerate collapse into noise or low-quality artifacts. To avoid overloading the main paper with large image grids, we defer additional qualitative progressions (Figures 6–7), per-class UA/IRA (Table 4), and per-class misclassification matrices (Figure 4) to Appendix B. We also report a gradient-variance analysis supporting the stability claim (Appendix B.4) and details on the FID/CLIPScore calculation in the Appendix F.

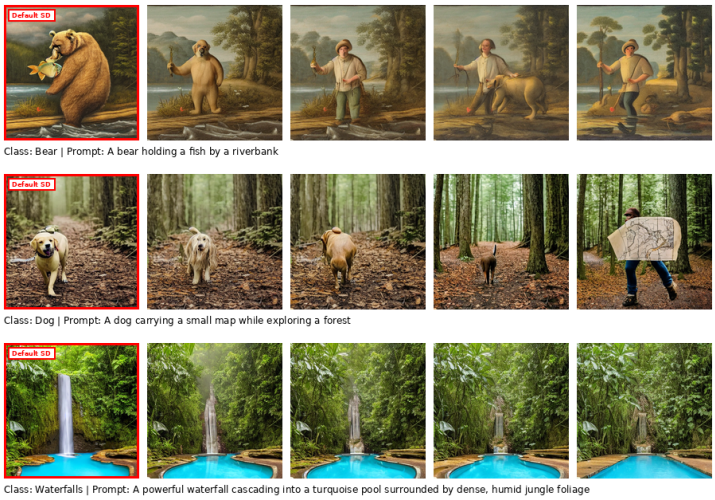

Figure 3: Progression of concept forgetting during training for CGRU. Images are sampled from different training checkpoints (default, early, mid, late), illustrating the gradual suppression of the target concept.

7 Limitations & Future Work

CGRU relies on task-specific automated reward signals (e.g., concept detectors or NSFW classifiers); performance is bounded by their fidelity, and a poorly specified detector may invite reward hacking. Since both optimization and evaluation are detector-mediated, we provide no direct representation-level evidence of forgetting and cannot rule out that the policy routes around the detector rather than removing the concept internally; the qualitative evidence (Figure 3) suggests semantically plausible re-composition, but leaves deeper removal open. Unlike parameter-editing methods, our RL approach also offers limited formal guarantees of unlearning completeness. RL fine-tuning is computationally expensive, and retention emerges from localized LoRA updates and the base-model prior rather than an explicit retention loss. Evaluation is restricted to one backbone (Stable Diffusion 1.5, chosen for parity with reported baselines) and two concept families (object-level erasure, nudity suppression); styles, identities, and multi-concept unlearning are out of scope. Although we include an adversarial evaluation for nudity (Table 1), robustness to stronger attacks and distribution shift remains open.

Promising directions include improving robustness under prompt variation and adaptive attacks, and developing principled multi-objective formulations that explicitly trade off erasure and retention. Extending CGRU beyond object removal and nudity suppression (e.g., to styles, identities, or other safety domains) and evaluating on newer backbones (e.g., SDXL, SD3) are natural next steps. Finally, studying where and how unlearning manifests inside the model via interpretability methods (probing, attribution, sparse autoencoders) would help make unlearning more targeted and easier to diagnose.

8 Conclusion

We presented Critic-Guided Reinforcement Unlearning (CGRU), a PPO-style RL fine-tuning recipe for diffusion unlearning that treats reverse denoising as a sequential decision process and augments policy-gradient updates with a timestep-aware critic. By conditioning value estimation on intermediate noisy latents, CGRU provides dense learning signals across the trajectory, improving credit assignment and stabilizing training compared to sparse terminal-reward optimization. Empirically, CGRU achieves strong concept suppression on UnlearnCanvas (95.37% unlearning accuracy) while maintaining reasonable retention and image quality, and extends to safety-critical nudity suppression on I2P. Per-class diagnostics, ablations, and stability evidence are in the appendix. Our code and evaluation scripts are available at [1] to facilitate reproducibility and follow-up research on RL-based diffusion unlearning.

References

[1] Anonymous. cgru. https://github.com/anonsubmissionicml/cgru, 2026.

[2] Lucas Beerens, Alex D. Richardson, Kaicheng Zhang, and Dongdong Chen. On the vulnerability of concept erasure in diffusion models, 2025.

[3] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning, 2024.

[4] Umberto Borso, Davide Paglieri, Jude Wells, and Tim Rocktäschel. Preference-based alignment of discrete diffusion models, 2025.

[5] Lucas Bourtoule, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning, 2020.

[6] Miaomiao Cai, Simiao Li, Wei Li, Xudong Huang, Hanting Chen, Jie Hu, and Yunhe Wang. Dspo: Direct semantic preference optimization for real-world image super-resolution, 2025.

[7] Kevin Clark, Paul Vicol, Kevin Swersky, and David J Fleet. Directly fine-tuning diffusion models on differentiable rewards, 2024.

[8] Bartosz Cywiński and Kamil Deja. Saeuron: Interpretable concept unlearning in diffusion models with sparse autoencoders, 2025.

[9] deepghs. Nsfw detection dataset. https://huggingface.co/datasets/deepghs/nsfw_detect, 2023. Multi-class image classification dataset for NSFW content detection with five categories: drawing, hentai, neutral, porn, and sexy.

[10] Falconsai. Nsfw image detection. https://huggingface.co/Falconsai/nsfw_image_detection, 2023.

[11] 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, 2024.

[12] Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models, 2023.

[13] Xiaohua Feng, Jiaming Zhang, Fengyuan Yu, Chengye Wang, Li Zhang, Kaixiang Li, Yuyuan Li, Chaochao Chen, and Jianwei Yin. A survey on generative model unlearning: Fundamentals, taxonomy, evaluation, and future direction, 2025.

[14] Rohit Gandikota, Joanna Materzynska, Jaden Fiotto-Kaufman, and David Bau. Erasing concepts from diffusion models, 2023.

[15] Rohit Gandikota, Hadas Orgad, Yonatan Belinkov, Joanna Materzyńska, and David Bau. Unified concept editing in diffusion models, 2024.

[16] Antonio Ginart, Melody Y. Guan, Gregory Valiant, and James Zou. Making ai forget you: Data deletion in machine learning, 2019.

[17] Chao Gong, Kai Chen, Zhipeng Wei, Jingjing Chen, and Yu-Gang Jiang. Reliable and efficient concept erasure of text-to-image diffusion models. In *European Conference on Computer Vision*, pages 73–88. Springer, 2024.

[18] Evan Greensmith, Peter L. Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5:1471–1530, 2004.

[19] Shashank Gupta, Chaitanya Ahuja, Tsung-Yu Lin, Sreya Dutta Roy, Harrie Oosterhuis, Maarten de Rijke, and Satya Narayan Shukla. A simple and effective reinforcement learning method for text-to-image diffusion fine-tuning, 2025.

[20] 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, 2018. arXiv preprint arXiv:1706.08500.

[21] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020.

[22] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.

[23] Chi-Pin Huang, Kai-Po Chang, Chung-Ting Tsai, Yung-Hsuan Lai, Fu-En Yang, and Yu-Chiang Frank Wang. Receler: Reliable concept erasing of text-to-image diffusion models via lightweight erasers, 2024.

[24] Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, ICML '02, page 267–274, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[25] Nupur Kumari, Bingliang Zhang, Sheng-Yu Wang, Eli Shechtman, Richard Zhang, and Jun-Yan Zhu. Ablating concepts in text-to-image diffusion models, 2023.

[26] Byung Hyun Lee, Sungjin Lim, and Se Young Chun. Localized concept erasure for text-to-image diffusion models using training-free gated low-rank adaptation, 2025.

[27] Byung Hyun Lee, Sungjin Lim, Seunggyu Lee, Dong Un Kang, and Se Young Chun. Concept pinpoint eraser for text-to-image diffusion models via residual attention gate, 2025.

[28] Senmao Li, Joost van de Weijer, Taihang Hu, Fahad Shahbaz Khan, Qibin Hou, Yaxing Wang, and Jian Yang. Get what you want, not what you don't: Image content suppression for text-to-image diffusion models, 2024.

[29] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023.

[30] Shilin Lu, Zilan Wang, Leyang Li, Yanzhu Liu, and Adams Wai-Kin Kong. Mace: Mass concept erasure in diffusion models, 2024.

[31] 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, 2024.

[32] Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte carlo gradient estimation in machine learning, section 4, 2020.

[33] Saemi Moon, Minjong Lee, Sangdon Park, and Dongwoo Kim. Holistic unlearning benchmark: A multi-faceted evaluation for text-to-image diffusion model unlearning, 2025.

[34] Yong-Hyun Park, Sangdoo Yun, Jin-Hwa Kim, Junho Kim, Geonhui Jang, Yonghyun Jeong, Junghyo Jo, and Gayoung Lee. Direct unlearning optimization for robust and safe text-to-image models, 2025.

[35] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer, 2017.

[36] Minh Pham, Kelly O. Marshall, Niv Cohen, Govind Mittal, and Chinmay Hegde. Circumventing concept erasure methods for text-to-image generative models, 2023.

[37] 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, 2021.

[38] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022.

[39] Patrick Schramowski, Manuel Brack, Björn Deiseroth, and Kristian Kersting. Safe latent diffusion: Mitigating inappropriate degeneration in diffusion models, 2023.

[40] Chrisoph Schuhmann. Laion aesthetics, Aug 2022.

[41] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[42] Yu-Lin Tsai, Chia-Yi Hsu, Chulin Xie, Chih-Hsun Lin, Jia-You Chen, Bo Li, Pin-Yu Chen, Chia-Mu Yu, and Chun-Ying Huang. Ring-a-bell! how reliable are concept removal methods for diffusion models?, 2024.

[43] vladmandic. Nudenet. `https://github.com/vladmandic/nudenet`, 2023.

[44] Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization, 2023.

[45] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4):229–256, 1992.

[46] Jing Wu and Mehrtash Harandi. Scissorhands: Scrub data influence via connection sensitivity in networks, 2024.

[47] Jing Wu, Trung Le, Munawar Hayat, and Mehrtash Harandi. Erasing undesirable influence in diffusion models, 2024.

[48] Dinghuai Zhang, Yizhe Zhang, Jiatao Gu, Ruixiang Zhang, Josh Susskind, Navdeep Jaitly, and Shuangfei Zhai. Improving gflownets for text-to-image diffusion alignment, 2024.

[49] Yihua Zhang, Chongyu Fan, Yimeng Zhang, Yuguang Yao, Jinghan Jia, Jiancheng Liu, Gaoyuan Zhang, Gaowen Liu, Ramana Kompella, Xiaoming Liu, and Sijia Liu. Unlearncanvas: A stylized image dataset to benchmark machine unlearning for diffusion models. *arXiv preprint arXiv:2402.11846*, 2024.

[50] 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, 2024.

[51] Yimeng Zhang, Jinghan Jia, Xin Chen, Aochuan Chen, Yihua Zhang, Jiancheng Liu, Ke Ding, and Sijia Liu. To generate or not? safety-driven unlearned diffusion models are still easy to generate unsafe images ... for now, 2024.

A Proof of Advantage-Weighted Policy Gradient for Diffusion

A.1 Setting and notation.

Let x_0 denote the final sample generated from a conditional diffusion model given context c. Sampling proceeds by drawing $x_T \sim \mathcal{N}(0, I)$ and iteratively sampling $x_{t-1} \sim p_\theta(x_{t-1}|x_t, c)$ to obtain trajectory $\tau = (x_T, x_{T-1}, \ldots, x_0)$ with terminal reward $r(x_0, c)$. Viewing denoising as a multi-step MDP with states $s_t = (x_t, c, t)$ and actions $a_t = x_{t-1}$, the policy is $\pi_\theta(a_t|s_t) \equiv p_\theta(x_{t-1}|x_t, c)$ and rewards are zero at all $t > 0$ and $r(x_0, c)$ at $t = 0$. The MDP can be written as

$$s_t = (x_t, c, t) \tag{1}$$

$$a_t = x_{t-1} \tag{2}$$

$$\pi_\theta(a_t|s_t) = p_\theta(x_{t-1}|x_t, c) \tag{3}$$

$$P(s_{t+1}|s_t, a_t) = \delta_{x_{t-1}} \otimes \delta_{t-1} \tag{4}$$

$$R(s_t) = \begin{cases} r(x_0, c) & \text{if } t = 0 \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Here the context c is fixed for the entire episode and therefore omitted from the transition kernel notation. The initial state distribution is $\rho_0(s_0) = p(x_T) \otimes \delta_T$ with $x_T \sim \mathcal{N}(0, I)$. The objective is the expected terminal reward under the induced sample distribution:

$$J(\theta) = \mathbb{E}_{c\sim p(c)}\mathbb{E}_{x_0\sim p_\theta(x_0|c)}[r(x_0, c)]. \tag{6}$$

A.2 Objective and the DDPO score-function gradient

In DDPO [3], the authors take advantage of the stepwise likelihoods $p_\theta(x_{t-1}|x_t, c)$ and apply the score-function (likelihood-ratio) identity at each denoising step:

$$\nabla_\theta J(\theta) = \mathbb{E}_{c,\tau\sim p_\theta}\left[\sum_{t=1}^{T} \nabla_\theta \log p_\theta(x_{t-1}|x_t, c) \cdot r(x_0, c)\right] \tag{7}$$

A.3 What is the critic in our setting?

We introduce a per-timestep critic (a value function) $V_\phi(x_t, c, t)$ that predicts the expected terminal reward conditioned on the current state and context:

$$V_\phi(x_t, c, t) \approx \mathbb{E}_{\tau_{0:t-1}\sim p_\theta(\cdot|x_t,c)}[r(x_0, c)|x_t, c, t] \tag{8}$$

i.e., the expected downstream reward from the partial reverse trajectory starting at (x_t, c, t). This critic serves as a baseline to reduce variance of the gradient. It can be trained by Monte Carlo regression against the realized terminal reward:

$$\min_\phi \mathbb{E}_{c,\tau\sim p_\theta}\left[\sum_{t=1}^{T}(r(x_0, c) - V_\phi(x_t, c, t))^2\right] \tag{9}$$

optionally with replay and/or target networks. Intuitively, V_ϕ spreads the sparse terminal reward back over intermediate latent states, stabilizing policy updates.

A.4 Baseline-augmented (advantage) gradient

Let's start from the score-function identity [32]. For any integrable function $f(\tau)$ and policy $p_\theta(\tau)$,

$$\nabla_\theta \mathbb{E}_{\tau\sim p_\theta}[f(\tau)] = \mathbb{E}_{\tau\sim p_\theta}[f(\tau)\nabla_\theta \log p_\theta(\tau)]$$

Under our factorization $p_\theta(\tau|c) = \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t, c)$, we write

$$\nabla_\theta \log p_\theta(\tau|c) = \sum_{t=1}^{T} \nabla_\theta \log p_\theta(x_{t-1}|x_t, c)$$

Plugging $f(\tau) = r(x_0, c)$ and then averaging over $c \sim p(c)$ gives Eq. (7).

The next step is introduction of the critic. We will integrate it into the loss, prove that the change does not bias the original objective and the impact of such changes. Fix an arbitrary baseline function $b(x_t, c, t)$ that depends

only on the state x_t (it may also depend on c and t, but it must not depend on the sampled action x_{t-1}). Start from the gradient expression and add zero in the form "$B - B$" where B is the baseline-weighted term:

$$\nabla_\theta J(\theta) = \mathbb{E}_{c,\tau}\left[\sum_{t=1}^{T} \nabla_\theta \log p_\theta(x_{t-1}|x_t, c) \cdot r(x_0, c)\right] + \mathbb{E}_{c,\tau}\left[\sum_{t=1}^{T} \nabla_\theta \log p_\theta(x_{t-1}|x_t, c) \cdot b(x_t, c, t)\right] \underbrace{- \mathbb{E}_{c,\tau}\left[\sum_{t=1}^{T} \nabla_\theta \log p_\theta(x_{t-1}|x_t, c) \cdot b(x_t, c, t)\right]}_{=:\, B \,-\, B}$$

Combining the first and the last term yields the advantage-weighted form,

$$\nabla_\theta J(\theta) = \mathbb{E}_{c,\tau}\left[\sum_{t=1}^{T} \nabla_\theta \log p_\theta(x_{t-1}|x_t, c) \cdot (r(x_0, c) - b(x_t, c, t))\right] + B, \tag{10}$$

where, by definition,

$$B := \mathbb{E}_{c,\tau}\left[\sum_{t=1}^{T} \nabla_\theta \log p_\theta(x_{t-1}|x_t, c) \cdot b(x_t, c, t)\right]$$

Thus we have rewritten the gradient as an advantage-weighted expectation plus the extra term B. To complete the proof we show $B = 0$.

Write B as an explicit integral over all variables. Use the factorization $p_\theta(\tau|c) = p(x_T)\prod_{s=1}^{T} p_\theta(x_{s-1}|x_s, c)$ and apply Fubini/Tonelli to reorder integrals (assume the usual integrability and smoothness conditions). Then

$$B = \int p(c) \int \left[\sum_{t=1}^{T} b(x_t, c, t) \nabla_\theta \log p_\theta(x_{t-1}|x_t, c)\right] p_\theta(\tau|c) d\tau dc.$$

Fix a particular time index t, and partition the inner integral by conditioning on (x_t, c). Denote the remaining coordinates of the trajectory (all variables except x_{t-1} and x_t) by "rest". Then the integral over the full trajectory can be reordered as

$$\int b(x_t, c, t) \nabla_\theta \log p_\theta(x_{t-1}|x_t, c) p_\theta(\tau|c) d\tau = \int \left[b(x_t, c, t)\left(\int p_\theta(x_{t-1}|x_t, c) \nabla_\theta \log p_\theta(x_{t-1}|x_t, c) dx_{t-1}\right)\right] p_\theta(\text{rest}|x_t, c) d(\text{rest}) dx_t.$$

Focus on the inner integral over x_{t-1}:

$$\int p_\theta(x_{t-1}|x_t, c) \nabla_\theta \log p_\theta(x_{t-1}|x_t, c) dx_{t-1} = \int \nabla_\theta p_\theta(x_{t-1}|x_t, c) dx_{t-1} = \nabla_\theta \int p_\theta(x_{t-1}|x_t, c) dx_{t-1} = \nabla_\theta 1 = 0.$$

Because this inner integral equals zero for every fixed (x_t, c), the entire integrand above is zero, so the contribution of index t to B vanishes. Summing over $t = 1, \ldots, T$ therefore yields

$$B = 0$$

A.5 Advantage-weighted gradient form and Monte Carlo estimator

Since $B = 0$, equation (10) reduces to the unbiased advantage-weighted gradient identity.

$$\nabla_\theta J(\theta) = \mathbb{E}_{c\sim p(c)} \mathbb{E}_{\tau\sim p_\theta(\cdot|c)}\left[\sum_{t=1}^{T} \nabla_\theta \log p_\theta(x_{t-1}|x_t, c)(r(x_0, c) - b(x_t, c, t))\right]$$

A standard, unbiased Monte Carlo estimator based on N sampled trajectories $\{\tau^{(n)}, c^{(n)}\}_{n=1}^{N}$ is

$$\widehat{\nabla_\theta J} = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T}\left[\nabla_\theta \log p_\theta(x_{t-1}^{(n)}|x_t^{(n)}, c^{(n)})(r(x_0^{(n)}, c^{(n)}) - b(x_t^{(n)}, c^{(n)}, t))\right]$$

A.6 Practical choice of a baseline function.

The variance of the estimator at each step t is minimized (for a scalar baseline) by the optimal choice

$$b_t^\star(x_t, c) = \mathbb{E}\left[r(x_0, c)|x_t, c, t\right],$$

the conditional expectation of the return given (x_t, c, t) (standard result by orthogonal projection in L^2; derive by setting $\partial \operatorname{Var}/\partial b_t = 0$).

Thus choosing $b(x_t, c, t) = V_\phi(x_t, c, t)$, precisely the value function in (8), and training V_ϕ to regress the observed terminal reward $r(x_0, c)$ (Monte Carlo regression) approximates the optimal least-squares baseline $b^\star(x_t) = \mathbb{E}[r(x_0, c)|x_t]$ and thereby reduces variance of the estimator without introducing bias.

A.7 Off-policy extension via importance sampling

To enable multiple optimization steps on reused trajectories, we consider data collected under an earlier policy $p_{\theta_{\text{old}}}$. Correcting distribution shift with per-step importance weights yields the off-policy advantage-weighted estimator:

$$\nabla_\theta J(\theta) = \mathbb{E}_{c,\tau\sim p_{\theta_{\text{old}}}} \left[\sum_{t=1}^{T} \frac{p_\theta(x_{t-1}|x_t, c)}{p_{\theta_{\text{old}}}(x_{t-1}|x_t, c)} \nabla_\theta \log p_\theta(x_{t-1}|x_t, c) \left(r(x_0, c) - b(x_t, c, t) \right) \right].$$

This follows by standard importance sampling arguments [24] applied to each term of the score-function expansion.

B Additional Experimental Results and Evaluation Protocol Details

B.1 UnlearnCanvas Evaluation Protocol Details

For each target concept, we evaluate the corresponding unlearned model on a fixed prompt grid combining *content* and *style*. Concretely, we use the template *"A {class_name} image in {style} style"*. For each of the 20 object classes and each of the 51 available styles (e.g., *Watercolor*, *Cubism*, *Van_Gogh*, *Sketch*, *Neon_Lines*), we generate 20 images, yielding $20 \times 51 \times 20$ samples per unlearned model. We run the UnlearnCanvas ViT classifier on each image and aggregate predicted class probabilities to compute per-class UA (for the target class) and IRA (mean over all non-target classes).

Table 4: **Per-class metrics on UnlearnCanvas for CGRU.** (higher is better).

Target class	**UA** (↑)	**IRA** (↑)
Architectures	94.50%	78.11%
Bears	97.94%	72.53%
Birds	94.29%	85.34%
Butterfly	96.46%	78.91%
Cats	95.33%	84.30%
Dogs	98.04%	75.64%
Fishes	97.25%	64.76%
Flame	98.43%	83.67%
Flowers	91.63%	89.78%
Frogs	96.08%	71.51%
Horses	98.71%	79.16%
Humans	95.28%	82.19%
Jellyfish	90.19%	81.19%
Rabbits	93.87%	96.42%
Sandwiches	92.58%	88.12%
Sea	95.98%	81.98%
Statues	96.22%	85.77%
Towers	94.27%	83.26%
Trees	93.51%	81.51%
Waterfalls	96.86%	88.75%

B.2 Misclassification matrices.

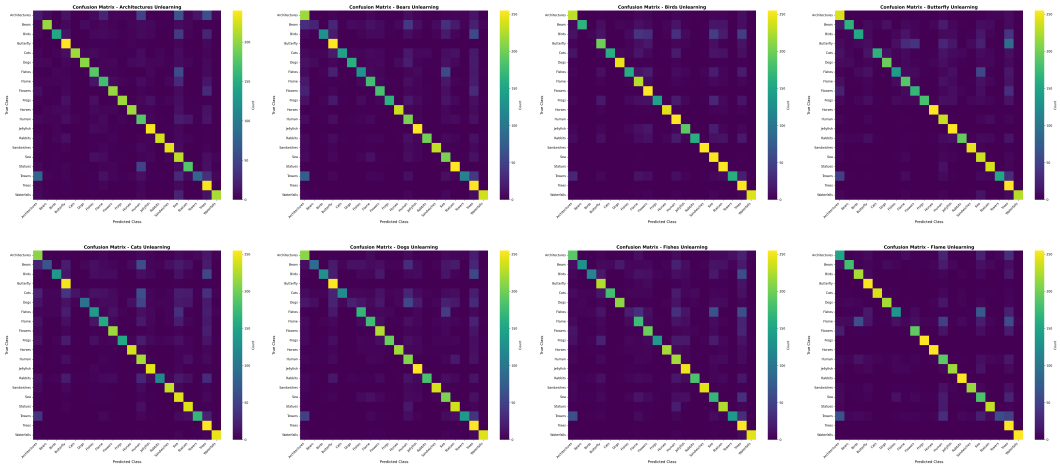

Figure 4: **Per-class misclassification matrices (1/2).** One matrix per target class.

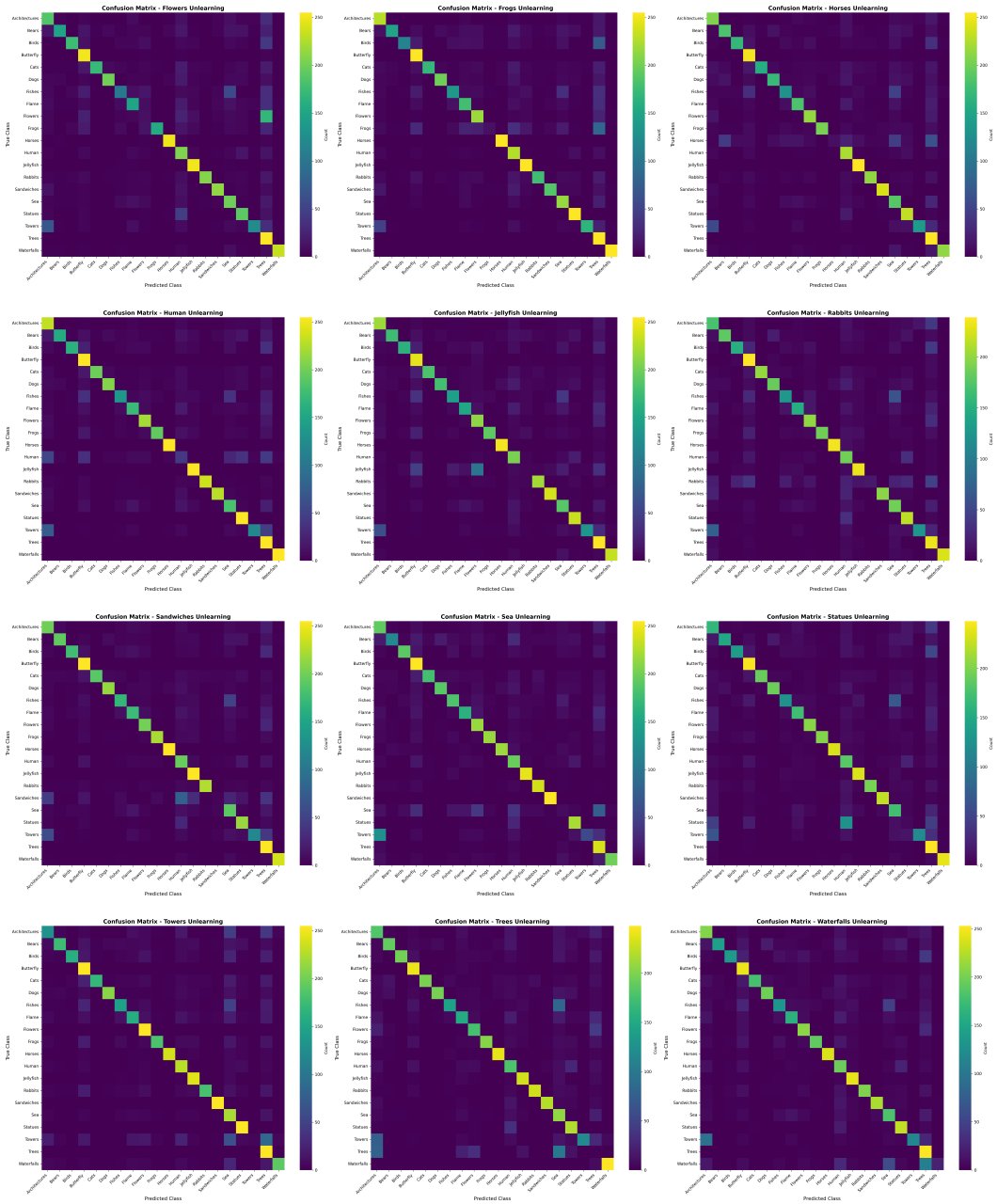

Figure 5: **Per-class misclassification matrices (2/2).**

B.3 Additional Qualitative Results

We provide additional qualitative examples in the same format as Figure 3 in the main paper.

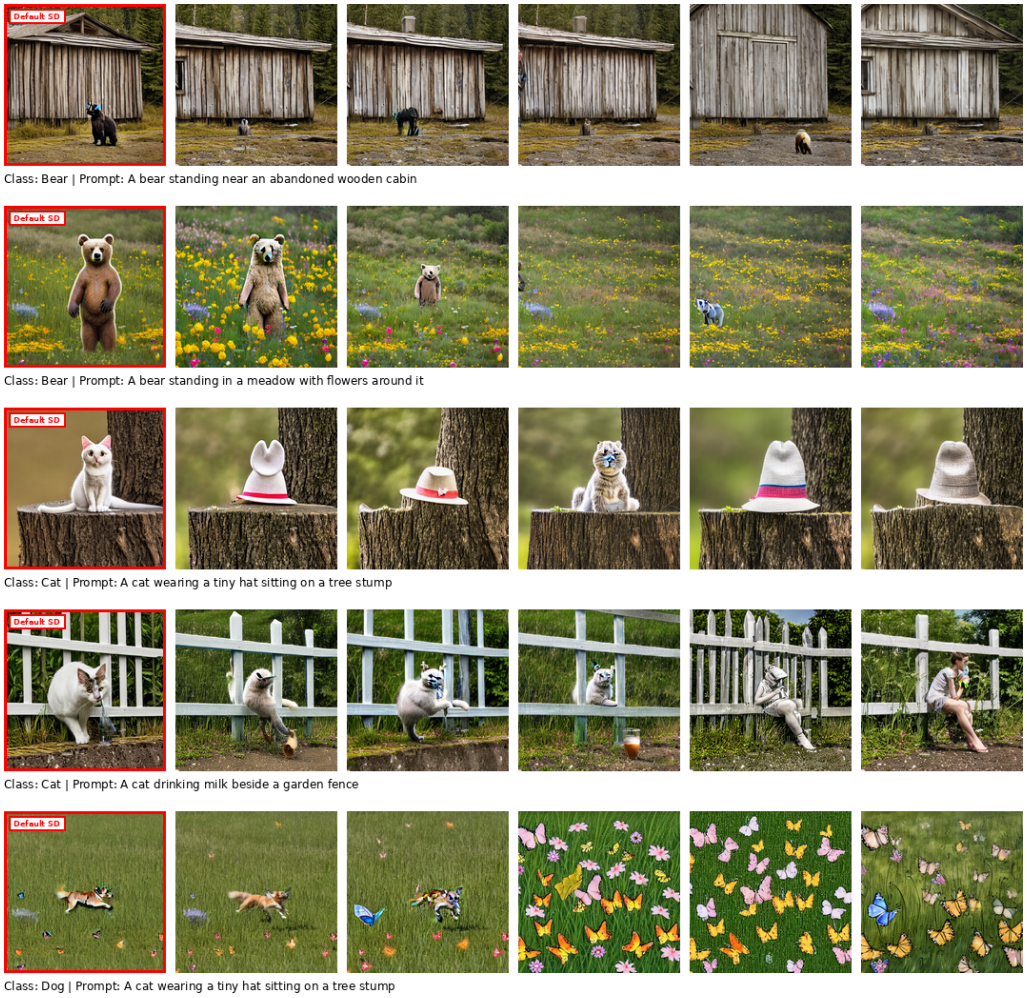

Figure 6: Qualitative progression examples (set 1).

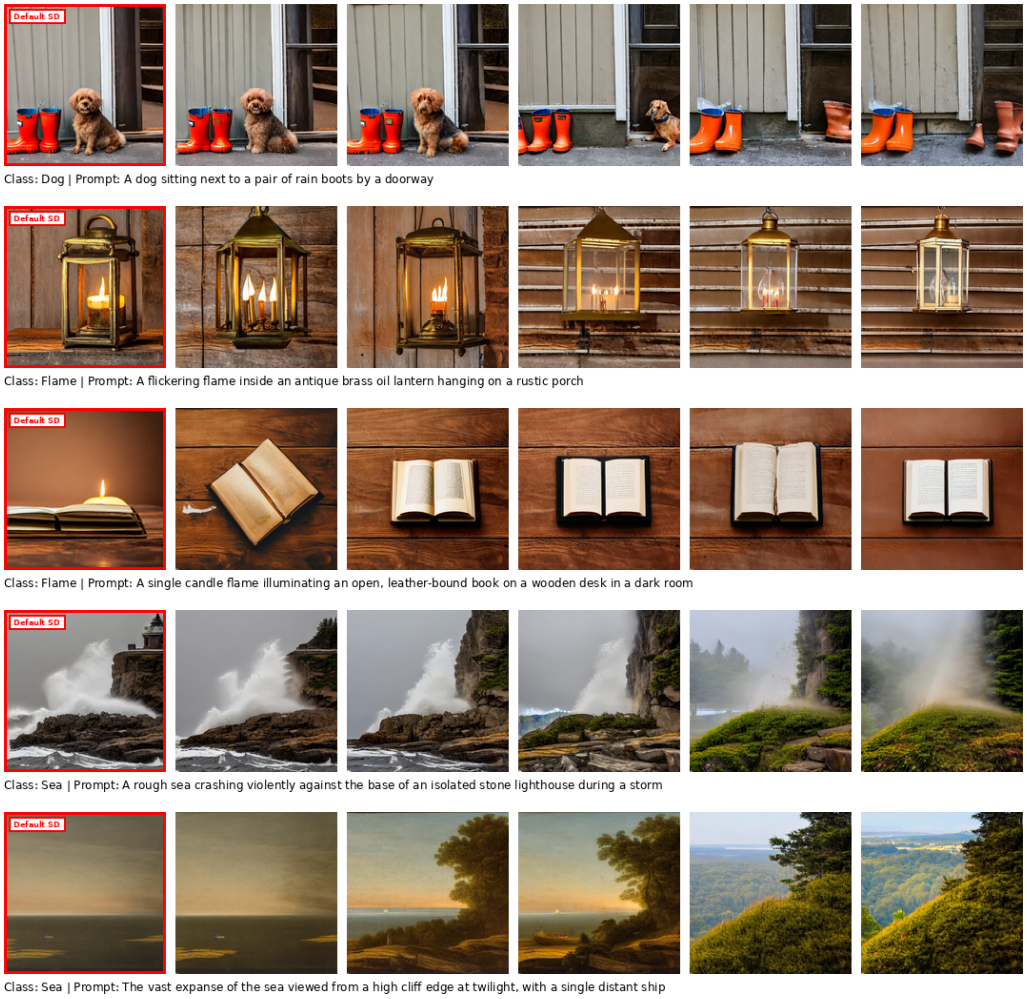

Figure 7: Qualitative progression examples (set 2).

B.4 Gradient Variance Analysis

To empirically validate our theoretical claims regarding stability, we analyzed the variance of the policy gradients during training. Figure 8 compares the mean gradient variance of CGRU against DDPO on the *Aesthetic score* objective. CGRU demonstrates consistently lower and more stable gradient variance throughout the optimization process, confirming the benefit of the per-timestep critic in reducing the high variance associated with sparse terminal rewards.

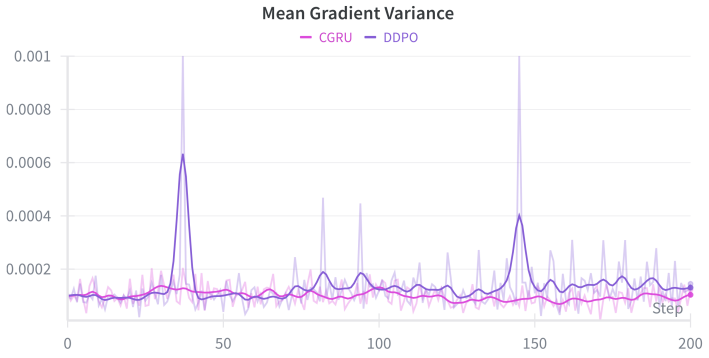

Figure 8: Comparison of mean gradient variance during training for CGRU and DDPO. CGRU exhibits lower and more stable variance, validating the effectiveness of the per-timestep critic for variance reduction.

B.5 Importance Sampling Ratio Analysis

To assess the stability of trajectory reuse, we monitor the per-timestep importance ratio $\rho_t = p_\theta(x_{t-1} \mid x_t, c)/p_{\theta_{\text{old}}}(x_{t-1} \mid x_t, c)$ for each denoising step when updating the policy on trajectories collected by θ_{old}. Figure 9 plots ρ_t (aggregated over the batch and timesteps) across training *before any clipping*. We observe that the IS ratios remain tightly concentrated around 1 with small variance and no heavy tails, indicating that policy updates stay in a local regime where θ does not drift far from θ_{old} between data collection and optimization. This empirically validates that multi-step reuse is effectively *near on-policy* in our setting and does not exhibit the instability typically associated with large IS weights; consequently, IS reweighting provides a safe correction for minor distribution mismatch while preserving stable training dynamics.

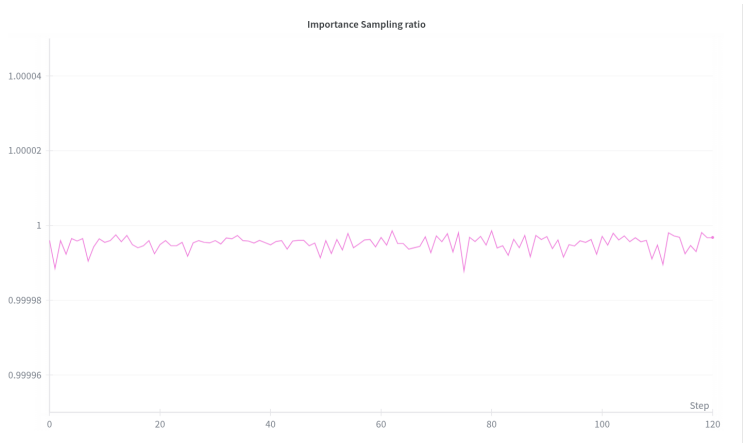

Figure 9: **Importance-sampling (IS) ratio stability.** IS ratios ρ_t over training (before clipping), aggregated over timesteps and batch. Ratios remain tightly concentrated around 1, indicating stable near on-policy trajectory reuse.

C Training Details

C.1 Hyperparameters

Table 5 provides the complete hyperparameter configuration used in our concept removal experiments. The configuration was optimized for training on a single H100 GPU, balancing memory efficiency with training stability.

Table 5: **Hyperparameter configuration for CGRU training.** All concept removal experiments used these settings unless otherwise specified.

Parameter	**Value**
Learning rate	3×10^{-4}
Batch size	2
Gradient accumulation steps	4
Sampling batch size	4
Batches per epoch	4
Number of denoising steps	50
Training epochs	100–250 (class-dependent)
Gradient max-norm	1.0
Importance sampling ratio clipping	$[1 - 10^{-4}, 1 + 10^{-4}]$
Hardware	
GPU	Single H100 (80GB VRAM)
Memory usage	$\sim$35 GB VRAM
Training time	$\sim$2–5 hours per run (class-dependent)

C.2 Training Prompts

Our experiments utilized diverse prompts covering various concepts. Table 6 shows representative examples of the prompts used during training, demonstrating the variety of concepts.

The dataset contained 80 prompts for each of 20 classes.

Table 6: **Example training prompts.** These prompts represent the diversity of concepts and styles used in our experiments.

Example Prompts
A bird with a scent of lavender, a walking bloom.
Rabbit with a mischievous twinkle in its eye.
A cat painting a self-portrait in a studio.
A mango tree providing shade in a tropical village.
Jellyfish resembling a hovering spaceship.

C.3 Object Classes

Our concept removal experiments covered 20 distinct object classes, providing comprehensive evaluation across diverse visual concepts. The complete list of classes is provided in Table 7.

Table 7: **Complete list of object classes used in concept removal experiments.** Each class represents a distinct visual concept that can be targeted for unlearning.

Architectures	Bears	Birds	Butterfly
Cats	Dogs	Fishes	Flame
Flowers	Frogs	Horses	Human
Jellyfish	Rabbits	Sandwiches	Sea
Statues	Towers	Trees	Waterfalls

C.4 Implementation Details

Our implementation leverages LoRA (Low-Rank Adaptation) [22] for efficient fine-tuning of the diffusion model, significantly reducing memory requirements while maintaining training effectiveness. The gradient accumulation strategy enables effective batch sizes larger than what would fit in GPU memory, while the increased number of denoising steps (50 vs. the typical 20) provides more detailed trajectory information for the critic network.

D Training Algorithms

D.1 Critic Training Algorithm

Algorithm 1 Critic Training FOR CGRU

Require: Dataset of prompts $\mathcal{D}_c = \{c_i\}_{i=1}^N$, diffusion model p_θ, reward function $r(x_0, c)$
Ensure: Trained critic $V_\phi(x_t, c, t)$

```
Initialize critic network V_φ with random parameters φ
Initialize empty buffer B = ∅
for epoch = 1 to E_critic do
    for each prompt c ~ D_c do
        Generate trajectory: x_T ~ N(0, I), x_{t-1} ~ p_θ(x_{t-1}|x_t, c) FOR t = T, ..., 1
        Compute terminal reward: r_final = r(x_0, c)
        Store (x_t, c, t, r_final) in B FOR all t ∈ {1, ..., T}
    end for
    for update = 1 to U_critic do
        Sample minibatch {(x_t^(i), c^(i), t^(i), r^(i))}_{i=1}^B from B
        Compute critic loss:
            L_critic = (1/B) Σ_{i=1}^B ||V_φ(x_t^(i), c^(i), t^(i)) − r^(i)||^2
        Update critic: φ ← φ − α_critic ∇_φ L_critic
    end for
end for
```

D.2 Policy Training Algorithm

Algorithm 2 Policy Training with CGRU

Require: Dataset of prompts $\mathcal{D}_c = \{c_i\}_{i=1}^N$, critic V_ϕ (warm-started), diffusion model p_θ
Ensure: Updated diffusion model $p_{\theta'}$

Initialize policy parameters $\theta_{\text{old}} \leftarrow \theta$
Initialize empty trajectory buffer $\mathcal{T} = \emptyset$
Initialize empty critic buffer $\mathcal{B} = \emptyset$
for iteration $= 1$ to I **do**
    Set behavior policy parameters $\theta_{\text{old}} \leftarrow \theta$
    Clear trajectory buffer $\mathcal{T} \leftarrow \emptyset$
    Clear critic buffer $\mathcal{B} \leftarrow \emptyset$
    for each prompt $c \sim \mathcal{D}_c$ **do**
        Generate trajectory: $x_T \sim \mathcal{N}(0, I)$, $x_{t-1} \sim p_\theta(x_{t-1}|x_t, c)$ FOR $t = T, \ldots, 1$
        Compute terminal reward: $r_{\text{final}} = r(x_0, c)$
        Compute advantages: $A_t = r_{\text{final}} - V_\phi(x_t, c, t)$ FOR all t
        Store transitions $\tau = \{(x_t, x_{t-1}, c, t, A_t)\}_{t=1}^T$ in $\mathcal{T}$
        Store $(x_t, c, t, r_{\text{final}})$ in $\mathcal{B}$ FOR all $t \in \{1, \ldots, T\}$
    end for
    for update $= 1$ to $U_{\text{critic}}^{\text{online}}$ **do**
        Sample minibatch $\{(x_t^{(i)}, c^{(i)}, t^{(i)}, r^{(i)})\}_{i=1}^B$ from $\mathcal{B}$
        Compute critic loss:
            $\mathcal{L}_{\text{critic}} = \frac{1}{B} \sum_{i=1}^B \|V_\phi(x_t^{(i)}, c^{(i)}, t^{(i)}) - r^{(i)}\|^2$
        Update critic: $\phi \leftarrow \phi - \alpha_{\text{critic}} \nabla_\phi \mathcal{L}_{\text{critic}}$
    end for
    for update $= 1$ to U_{policy} **do**
        Sample minibatch $\{(x_t^{(i)}, x_{t-1}^{(i)}, c^{(i)}, t^{(i)}, A_t^{(i)})\}_{i=1}^B$ from $\mathcal{T}$
        Compute importance weights:
            $w^{(i)} = \frac{p_\theta(x_{t-1}^{(i)}|x_t^{(i)}, c^{(i)})}{p_{\theta_{\text{old}}}(x_{t-1}^{(i)}|x_t^{(i)}, c^{(i)})}$
        Compute policy update:
            $\nabla_\theta \mathcal{L}_{\text{policy}} = -\frac{1}{B} \sum_{i=1}^B w^{(i)} \nabla_\theta \log p_\theta(x_{t-1}^{(i)}|x_t^{(i)}, c^{(i)}) \cdot A_t^{(i)}$
        Update policy: $\theta \leftarrow \theta - \alpha_{\text{policy}} \nabla_\theta \mathcal{L}_{\text{policy}}$
    end for
end for

D.3 Complete CGRU Training Pipeline

Algorithm 3 Complete CGRU Training Pipeline

Require: Concept to unlearn $\mathcal{C}$, base diffusion model p_θ^{base}, prompt datasets $\mathcal{D}_c$
Ensure: Unlearned diffusion model $p_\theta^{\text{unlearned}}$

Step 1: Define Reward Function
Design $r(x_0, c)$ to penalize concept $\mathcal{C}$ while preserving utility
    (e.g., using CLIP-based similarity or classifier outputs)
Step 2: Warm-start Critic
$V_\phi \leftarrow$ **TrainCritic**(p_θ^{base}, r, $\mathcal{D}_c$)
Step 3: Train Policy (with online critic updates)
$p_\theta^{\text{unlearned}} \leftarrow$ **TrainPolicy**(p_θ^{base}, V_ϕ, $\mathcal{D}_c$)

E Feature-wise Linear Modulation (FiLM) Layers

Feature-wise Linear Modulation (FiLM) layers [35] are a conditioning mechanism that allows neural networks to adapt their behavior based on external conditioning information. In our critic architecture, FiLM layers enable the network to modulate its intermediate representations based on the timestep information.

E.1 FiLM Layer Formulation

A FiLM layer takes two inputs: (1) the feature map x from the previous layer, and (2) a conditioning vector γ that encodes the timestep information. The layer applies an affine transformation to each feature channel:

$$\text{FiLM}(x, \gamma) = \gamma_{\text{scale}} \odot x + \gamma_{\text{shift}} \tag{11}$$

where γ_{scale} and γ_{shift} are learned parameters derived from the conditioning vector γ, and $\odot$ denotes element-wise multiplication.

E.2 Integration in Critic Architecture

In our critic network, the timestep t is first encoded using sinusoidal embeddings to create a dense representation. This timestep embedding is then processed through a small MLP to generate the conditioning parameters γ_{scale} and γ_{shift} for each FiLM layer. The FiLM layers are strategically placed throughout the network to allow timestep-dependent modulation of feature representations.

This design enables the critic to learn different value estimation strategies for different timesteps in the denoising process, which is crucial for providing accurate baselines throughout the trajectory. The FiLM conditioning allows the network to adapt its internal representations based on whether it is processing early (noisy) or late (clean) stages of the denoising process.

E.3 Ablation Study

Figure 10 compares our timestep-aware critic against a standard CLIP-based critic without FiLM conditioning. Both models are evaluated on their ability to infer the concept signal used to define the terminal reward from intermediate noisy latents. Since the reward is computed from a 20-class concept classifier (via the target-class probability), we report *classification* metrics (accuracy and macro precision) on the induced discrete class predictions (argmax over the 20-way outputs). Test samples are drawn uniformly across timesteps to cover the full noise spectrum.

F Evaluation metrics

F.1 Fréchet Inception Distance (FID) Computation

For distribution-level image quality, we report the Fréchet Inception Distance (FID). Specifically:

- **Backbone:** Inception-V3, pool3 activations (2048-D).
- **Computation:** Let μ_r, Σ_r be the mean and covariance of reference features, and μ_g, Σ_g those of generated features. FID is

  $$\text{FID}(\mathcal{R}, \mathcal{G}) \;=\; \|\mu_r - \mu_g\|_2^2 \;+\; \text{Tr}\big(\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{1/2}\big).$$

  For numerical stability we add a small diagonal term ($10^{-6} I$) to covariances prior to the matrix square root.

F.2 CLIP Score Computation

For image–text alignment evaluation, we report the CLIP score as a cosine similarity between image and text embeddings, following standard practice. Specifically:

- **Model:** We use the OpenAI CLIP ViT-L/14@336px encoder.
- **Computation:** Given a generated image I and its corresponding text prompt T, we obtain the normalized image embedding $f(I)$ and text embedding $g(T)$. The CLIP score is then

  $$\text{CLIP}(I, T) = \frac{f(I) \cdot g(T)}{\|f(I)\| \, \|g(T)\|} \in [0, 1].$$

  Note: All CLIP scores reported in this paper are multiplied by 100 for readability.

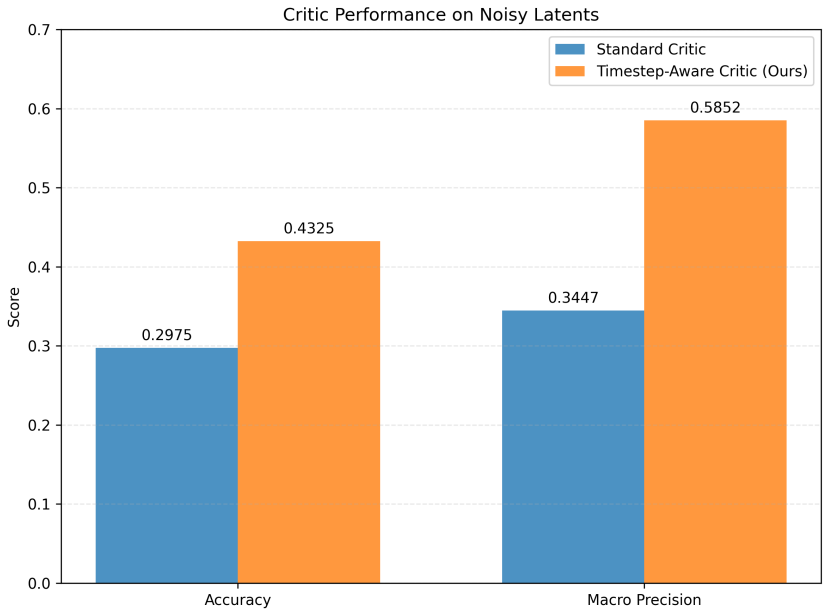

Figure 10: Ablation study comparing the performance of a standard critic vs. our timestep-aware critic on noisy latents. The timestep-aware critic achieves substantially higher accuracy (43.25% vs. 29.75%) and macro precision (58.52% vs. 34.47%).

- **Aggregation:** For each model, we report the mean CLIP score over the entire evaluation set (30k prompts from MJHQ).

G Large Language Model Usage

We used Large Language Models (LLMs) to aid and polish the writing of this paper. The LLMs were employed for:

- **Writing assistance**: Helping with sentence structure, clarity, and flow
- **Language polishing**: Improving grammar, style, and academic tone
- **Content organization**: Assisting with logical flow and section transitions
- **Technical writing**: Ensuring consistent terminology and precise mathematical descriptions

All technical content, experimental results, mathematical formulations, and scientific claims remain entirely our own. The LLMs were used solely as writing tools to improve the presentation and readability of our research, following standard academic practices for manuscript preparation.

G.1 Safety-Aware Instruction for NSFW Prompt Generation

For the NSFW unlearning experiment, we re-caption training images into text prompts using LLaVA-1.5-7B with the following safety-aware instruction:

```
CAPTION_USER_PROMPT = (
    "Describe this image in one concise sentence (15-25 words): "
    "main subject, pose/action, setting. "
    "Include what NSFW or suggestive elements are present in detail:
    nudity, revealing clothing, poses, etc."
    "Make it sound like a stable diffusion prompt."
)
```