Diffusion-SDPO: Safeguarded Direct Preference Optimization for Diffusion Models

Minghao Fu[1,2,3⋆], Guo-Hua Wang[3⋆⋆], Tianyu Cui[3], Qing-Guo Chen[3], Zhao Xu[3], Weihua Luo[3], and Kaifu Zhang[3]

[1] School of Artificial Intelligence, Nanjing University, Nanjing, China
[2] National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
[3] Alibaba Cloud Computing, Hangzhou, China
fumh@lamda.nju.edu.cn, wangguohua@alibaba-inc.com

Abstract. Text-to-image diffusion models deliver high-quality images, yet aligning them with human preferences remains challenging. We revisit diffusion-based Direct Preference Optimization (DPO) for these models and identify a critical pathology: enlarging the preference margin does not necessarily improve generation quality. In particular, the standard Diffusion-DPO objective can increase the reconstruction error of both winner and loser branches. Consequently, degradation of the less-preferred outputs can become sufficiently severe that the preferred branch is also adversely affected even as the margin grows. To address this, we introduce Diffusion-SDPO, a safeguarded update rule that preserves the winner by adaptively scaling the loser gradient according to its alignment with the winner gradient. A first-order analysis yields a closed-form scaling coefficient that guarantees the error of the preferred output is non-increasing at each optimization step. Our method is simple, model-agnostic, broadly compatible with existing DPO-style alignment frameworks and adds only marginal computational overhead. Across standard text-to-image benchmarks, including validations on the large-scale FLUX.1-dev model, Diffusion-SDPO delivers consistent gains over preference-learning baselines on automated preference, aesthetic, and prompt alignment metrics. Our code is available at https://github.com/AIDC-AI/Diffusion-SDPO.

Keywords: Text-to-Image Generation · Diffusion Models · Direct Preference Optimization

1 Introduction

Text-to-image diffusion models [5] have achieved remarkable success in generating diverse and high-quality images [10, 19]. However, aligning these powerful generative models with nuanced human preferences remains a critical challenge. Recent approaches have begun to incorporate human feedback [4] into

⋆ Work done during the internship at Alibaba Cloud Computing.
⋆⋆ G. Wang is the corresponding author.

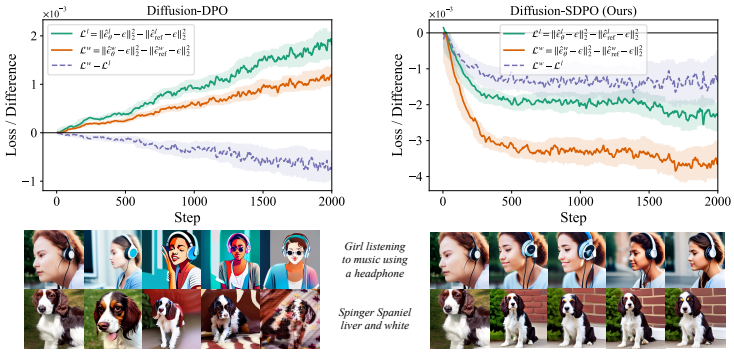

Fig. 1: Training dynamics of preference losses during DPO finetuning *without (left)* and *with (right)* our safe-λ mechanism on SD 1.5 [31]. Images beneath the plots illustrate samples generated at training steps $\{0, 500, 1000, 1500, 2000\}$.

diffusion model training, drawing inspiration from alignment techniques used in large language models. In particular, Direct Preference Optimization (DPO) [30] has emerged as a promising alternative to reinforcement learning for finetuning on human preferences. DPO directly optimizes the model on pairwise human comparisons (winner vs. loser outputs), and has been successfully adapted to text-to-image diffusion models in methods [13, 21, 22, 38, 48] to improve visual appeal and prompt alignment. Despite these advances, we find that existing DPO-based alignment of diffusion models still faces a fundamental limitation: simply maximizing the preference margin between "winner" and "loser" outputs does *not* necessarily translate to better absolute generation quality of the fine-tuned model.

In our empirical analysis, we find that standard Diffusion-DPO [38] exhibits unstable training dynamics, and the model's generative quality can deteriorate as training proceeds. As illustrated in the left part of Fig. 1, we find that both the winner's and loser's denoising losses tend to increase over time, even though the preference margin ($\mathcal{L}^w - \mathcal{L}^l$) becomes more negative in the intended direction. This indicates that the model is widening the relative preference gap by making the less-preferred outputs worse, rather than truly improving the preferred outputs. In other words, relative alignment comes at the expense of absolute quality. The lack of a safeguard on the winner's loss in existing DPO objectives leads to unstable training and potential collapse, corroborating observations in prior work [26, 34, 42] that overly aggressive preference optimization can harm generative performance. These findings motivate the need for a new approach to preference-based diffusion finetuning that can increase preference alignment while preserving or improving the quality of the preferred outputs.

To address this challenge, we propose Diffusion-SDPO[4] – a Safeguarded Direct Preference Optimization method for diffusion models. The key idea in Diffusion-SDPO is to introduce a simple yet effective winner-preserving update rule that controls the influence of the loser sample's gradient at each training step. In contrast to standard DPO [30,38] which updates the model by contrasting winner and loser equally, we derive an adaptive scaling factor for the loser's gradient based on the geometry of the winner and loser gradients. Intuitively, our method downweights the loser branch's contribution whenever its gradient is misaligned with the winner's gradient. Grounded in a first-order analysis, the safeguard computes a closed-form λ_{safe} from the inner product of the winner and loser gradients, guaranteeing that each step does not worsen the preferred output's reconstruction loss. In practice, Diffusion-SDPO seamlessly modifies the DPO objective with this adaptive loser scaling (see Fig. 1, right), which expands the preference margin while strictly controlling the absolute error of preferred outputs. Notably, our approach is model-agnostic and can be applied on top of various diffusion alignment frameworks [13, 21, 38, 48], acting as a plug-in optimizer that stabilizes training. Our contributions can be summarized as follows:

- We show that enlarging the winner–loser margin in diffusion preference optimization does not guarantee higher quality and can degrade preferred outputs, revealing a gap between relative alignment and absolute error control.
- Based on these analysis, we propose *Diffusion-SDPO*, a winner-preserving training scheme that adaptively scales the loser gradient by its geometric alignment with the winner gradient to first order. Our method is simple to implement and adds negligible overhead.
- Extensive experiments on SD 1.5 [31], SDXL [28] (both UNets [32]), and the industrial-scale Ovis-U1 [39], FLUX.1-dev [19] (both DiTs [27]) show that our method is architecture-agnostic. It delivers consistent improvements in preference metrics while preserving or enhancing aesthetic quality, stabilizing training, and avoiding collapse. These benefits persist across text-to-image, image editing and unified generation setups.

2 Related Work

Diffusion Models for Text-to-Image and Unified Generation. Diffusion models have become a leading paradigm for image synthesis, offering strong quality and diversity [5]. Denoising diffusion with a variational objective [11] and continuous-time score-based formulations with SDEs [16,36] underpin modern systems. Refinements such as EDM [17] and rectified flow or flow matching [23, 25] clarify objectives and improve robustness. Guidance-based conditioning [6,12] enhances

[4] Throughout the text, "Diffusion-SDPO" is used as a conceptual umbrella for our method and its guiding principles. When referring to concrete instantiations, we write "X+SDPO" to denote the integration of SDPO with a specific base DPO variant X (e.g., Diffusion-DPO, DSPO, DMPO), which clarifies the application setting and configuration.

controllability. For text-to-image generation, latent diffusion [31] enables efficient high-resolution synthesis and supports large systems like SD3 [7] and FLUX [19]. In parallel, unified generators handle text-to-image and image editing within a single model [39]. Our method applies to both families and is architecture-agnostic, working with UNet [32]-style and DiT [27]-style backbones.

Preference Optimization for Diffusion Models. Direct Preference Optimization [8, 30, 38] has been adapted to diffusion models to align generation with human comparisons while avoiding full reinforcement learning. A broad class of variants [13, 40] calibrates the preference margin or the relative branch influence to improve stability and protect the generation. Other approaches seek to guide the update directions and step magnitudes in LLMs [47] by employing subspace projections and modest objective clipping [2, 3, 15, 42, 44]. Related work such as DPOP [26] promotes positivity constraints to mitigate failure modes in preference optimization, and MaPPO [20] incorporates prior knowledge via a maximum-a-posteriori objective. Diffusion-specific methods further account for the multi-step nature of denoising by reweighting across timesteps or by adding entropy regularization, exemplified by Balanced-DPO [37], DSPO [48], and SEE-DPO [34]. In contrast, our Diffusion-SDPO introduces a per-step, geometry-aware safe scaling factor based on the inner product between winner and loser output-space gradients, which provides direct control over the winner loss at each step while continuing to expand the preference margin.

3 Preliminaries

Diffusion Models. Diffusion models [11,35] construct a Markov chain that gradually corrupts clean data with additive noise and then learn a parametric denoiser to invert this corruption. Let a variance schedule $\{\beta_t\}_{t=1}^{T}$ be given and define $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$. The forward process can be defined as:

$$q(x_t \mid x_{t-1}) \;=\; \mathcal{N}\big(x_t;\, \sqrt{\alpha_t}\, x_{t-1},\, (1-\alpha_t)\, \mathbf{I}\big), \tag{1}$$

which implies the following closed-form perturbation of a clean sample x_0:

$$x_t \;=\; \sqrt{\bar{\alpha}_t}\, x_0 \;+\; \sqrt{1-\bar{\alpha}_t}\, \epsilon, \qquad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{2}$$

Equivalently, the marginal distribution conditioned on x_0 is

$$q(x_t \mid x_0) \;=\; \mathcal{N}\big(x_t;\, \sqrt{\bar{\alpha}_t}\, x_0,\, (1-\bar{\alpha}_t)\, \mathbf{I}\big). \tag{3}$$

Learning proceeds by training a network ϵ_θ that receives the noised input x_t and the time index t to predict the injected noise. Using the reparameterization in Eq. 2, the standard objective minimizes mean squared error between the true noise and the prediction:

$$\mathcal{L}_{\text{diffusion}} \;=\; \mathbb{E}_{x_0,t,\epsilon}\, \big\| \,\epsilon_\theta\big(x_t, t\big) - \epsilon \big\|_2^2. \tag{4}$$

where $x_0 \sim p_{\text{data}}, t \sim \text{Uniform}\{1, \ldots, T\}$ and $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Minimizing $\mathcal{L}_{\text{diffusion}}$ yields a time-aware denoiser that can be applied in reverse order to iteratively remove noise and synthesize new samples from an initial Gaussian latent.

Diffusion Model Alignment via Preference. Given a prompt c and two images x_0^w (preferred, "winner") and x_0^l (less preferred, "loser"), preference alignment for diffusion models seeks parameters θ such that the model assigns higher likelihood to x_0^w than to x_0^l [8,38]. A diffusion sampler produces a trajectory $(x_T, \dots, x_0)$ and, at each time t, a reverse conditional $p_\theta(x_t \mid x_{t+1}, c)$ [11,25,35]. To instantiate DPO in this setting, we adopt the standard formulation wherein the stepwise preference score is the log-likelihood ratio with respect to a frozen reference model [38,48]:

$$r_t(x_t, c) \;=\; \beta \,\log \frac{p_\theta(x_t \mid x_{t+1}, c)}{p_{\text{ref}}(x_t \mid x_{t+1}, c)}\,. \tag{5}$$

The Diffusion–DPO [38] loss applies Bradley-Terry-style [1] logistic regression to the winner-loser pair at the same t:

$$\mathcal{L}_{\text{Diffusion-DPO}} \;=\; -\,\mathbb{E}\Big[\,\log\sigma\Big(r_t(x_t^w, c) \;-\; r_t(x_t^l, c)\Big)\Big], \tag{6}$$

and averages Eq. 6 over $t \in \{0, \dots, T-1\}$ (or samples a single t per pair for an unbiased stochastic estimator). Equivalently, substituting Eq. 5 into Eq. 6 gives the explicit form

$$\begin{aligned}\mathcal{L}_{\text{Diffusion-DPO}} = -\;\mathbb{E}\Big[\log\sigma\Big(&\beta\log\frac{p_\theta(x_t^w \mid x_{t+1}^w, c)}{p_{\text{ref}}(x_t^w \mid x_{t+1}^w, c)} \\ &- \beta\log\frac{p_\theta(x_t^l \mid x_{t+1}^l, c)}{p_{\text{ref}}(x_t^l \mid x_{t+1}^l, c)}\Big)\Big].\end{aligned} \tag{7}$$

Under common parameterizations, Eq. 5 reduces to simple residual comparisons. For DDPM-style Gaussians [11,35], writing $\hat{\epsilon}_\theta = \epsilon_\theta(x_{t+1}, c, t)$ for the predicted noise, $\hat{\epsilon}_{\text{ref}} = \epsilon_{\text{ref}}(x_{t+1}, c, t)$ for the reference noise and ϵ for the ground-truth noise that forms x_t, the log-ratio can be expressed as:

$$\log\frac{p_\theta(x_t \mid x_{t+1}, c)}{p_{\text{ref}}(x_t \mid x_{t+1}, c)} \propto -\tfrac{1}{2}\big\|\hat{\epsilon}_\theta - \epsilon\big\|_2^2 + \tfrac{1}{2}\big\|\hat{\epsilon}_{\text{ref}} - \epsilon\big\|_2^2 + \text{const}, \tag{8}$$

and an analogous expression holds for velocity or flow-matching parameterizations by replacing the noise residual with the corresponding target [25]. For notational brevity, we write the stepwise contrastive objective as $\mathcal{L}(x_{t+1}, c, t) = \frac{1}{2}\|\epsilon_\theta(x_{t+1}, c, t) - \epsilon\|_2^2 - \frac{1}{2}\|\epsilon_{\text{ref}}(x_{t+1}, c, t) - \epsilon\|_2^2$. Hence, the winner and loser margin loss are defined as $\mathcal{L}^w = \mathcal{L}(x_{t+1}^w, c, t)$ and $\mathcal{L}^l = \mathcal{L}(x_{t+1}^l, c, t)$, respectively. Substituting Eq. 8 into Eq. 7 gives the training loss

$$\hat{\mathcal{L}}_{\text{Diffusion-DPO}} = -\mathbb{E}_{t,\epsilon,c,x_0^w,x_0^l}\Big[\log\sigma\Big(-\beta(\mathcal{L}^w - \mathcal{L}^l)\Big)\Big], \tag{9}$$

where $\mathcal{D} = \{(c, x_0^w, x_0^l)\}$ denotes the DPO training dataset.

Limitations of Standard DPO. Substituting Eq. 8 into Eq. 7 yields an implementable objective whose inner term is the per-step error difference between winner and loser branches. Diffusion–DPO [38] thus encourages decreasing the

winner's prediction error while increasing the loser's at the same timestep. However, this objective does not guarantee a monotonic decrease of the winner loss. Empirically, over-penalizing the loser can also worsen the preferred sample. In the left part of Fig. 1, the margin $\mathcal{L}^w - \mathcal{L}^l$ becomes increasingly negative, yet both $\mathcal{L}^w$ and $\mathcal{L}^l$ increase, indicating degradation of absolute performance and potential instability or collapse. This exposes a gap between *relative* alignment (widening the margin) and *absolute* error control (preserving the preferred sample). The difficulty is that the winner and loser gradients are misaligned and vary across timesteps. We therefore introduce a simple stepwise update that, to first order, guarantees the preferred loss does not increase at each step while still promoting margin expansion.

4 Method: Diffusion-SDPO (Safe DPO)

We propose **Diffusion-SDPO**, a novel preference optimization scheme that adds a safety guard to the DPO update. The method adaptively scales the influence of the loser branch by a time-dependent factor λ_t so that the preferred sample's loss $\mathcal{L}^w$ does not increase after each parameter update. In practice, we follow the standard Diffusion–DPO pipeline: given a prompt c and a pair (x_0^w, x_0^l), we compute the per-sample losses $\mathcal{L}^w$ and $\mathcal{L}^l$ at the same diffusion time t, and then modify the backpropagated update by multiplying the loser-branch gradient by the safety factor to enforce a safe update condition. This directly addresses the limitation discussed above, because preventing any increase in the preferred loss ensures that preference-driven updates do not degrade the preferred output while still improving the preference margin.

4.1 Safe Update via First-Order Approximation

Our objective is to ensure that a gradient update driven by the preference loss (cf. Eq. 7) does not increase the winner's loss. For clarity of exposition, consider a linearized preference objective combining the two branches:[5]

$$\mathcal{L}^{\text{pref}}(\theta) = \mathcal{L}^w(\theta) - \lambda \cdot \mathcal{L}^l(\theta), \tag{10}$$

where $\lambda > 0$ is a scalar that adjusts the relative weight on the loser's loss. Setting $\lambda = 1$ recovers the intuitive gradient direction of standard DPO (decrease $\mathcal{L}^w$, increase $\mathcal{L}^l$), while $\lambda > 1$ would place even more emphasis on penalizing the loser. Our goal is to find an upper bound on λ that guarantees $\mathcal{L}^w$ will not increase for an infinitesimal gradient step on $\mathcal{L}^{\text{pref}}$.

Let $\nabla_\theta \mathcal{L}^w$ and $\nabla_\theta \mathcal{L}^l$ denote the gradients of the winner and loser losses, respectively. A gradient descent step of size η on Eq. 10 gives the parameter

[5] In practice, the actual Diffusion-DPO gradient (Eq. 6) includes a logistic scaling factor $\sigma(\cdot)$ that multiplies the winner and loser gradients equally, thus not altering the update direction. We therefore analyze the simpler weighted difference objective that captures the same first-order direction.

update:

$$\Delta\theta = -\eta \cdot \nabla_\theta \mathcal{L}^{\text{pref}} = -\eta\Big(\nabla_\theta \mathcal{L}^w - \lambda \nabla_\theta \mathcal{L}^l\Big). \tag{11}$$

The first-order change in the winner's loss can be approximated by a Taylor expansion:

$$\Delta\mathcal{L}^w \approx \nabla_\theta \mathcal{L}^{w\top} \Delta\theta = -\eta\Big(\|\nabla_\theta \mathcal{L}^w\|_2^2 - \lambda \nabla_\theta \mathcal{L}^{w\top} \nabla_\theta \mathcal{L}^l\Big). \tag{12}$$

To *prevent* increase in $\mathcal{L}^w$, we require $\Delta\mathcal{L}^w \leq 0$, i.e., $\nabla_\theta \mathcal{L}^{w\top} \Delta\theta \leq 0$. Ignoring the trivial positive factor η, the safety condition becomes:

$$\|\nabla_\theta \mathcal{L}^w\|_2^2 - \lambda \nabla_\theta \mathcal{L}^{w\top} \nabla_\theta \mathcal{L}^l \geq 0. \tag{13}$$

Solving for λ yields a bound on the allowable loser weight:

$$\lambda \leq \frac{\|\nabla_\theta \mathcal{L}^w\|_2^2}{\nabla_\theta \mathcal{L}^{w\top} \nabla_\theta \mathcal{L}^l}. \tag{14}$$

Notably, if the dot product $\nabla_\theta \mathcal{L}^{w\top} \nabla_\theta \mathcal{L}^l$ is negative or zero, then Eq. 13 is automatically satisfied for any $\lambda \geq 0$. In those cases, the update is intrinsically safe: the loser branch either helps reduce $\mathcal{L}^w$ or affects orthogonal parameter directions. The problematic scenario is when $\nabla_\theta \mathcal{L}^{w\top} \nabla_\theta \mathcal{L}^l > 0$, i.e., the loser's gradient has a component that would raise the winner's loss. Eq. 14 then yields a finite positive λ threshold. Any choice of λ above this threshold would violate the safety inequality, leading to $\Delta\mathcal{L}^w > 0$ to first order. Conversely, choosing λ at or below this threshold ensures $\Delta\mathcal{L}^w \approx 0$ or negative, guaranteeing that the winner's loss does not increase.

4.2 Closed-Form Safeguard in Output Space

Directly evaluating the parameter-space bound in Eq. 14 is infeasible for a high-dimensional model, since it would require computing and storing the full gradients $\nabla_\theta \mathcal{L}^w$ and $\nabla_\theta \mathcal{L}^l$ just to take their dot product. However, we can derive a convenient proxy by considering gradients in the model's *output space*. Modern diffusion models predict a noise or image tensor as output, and the training loss (e.g., a denoising score-matching loss [11]) is defined on this output. Let o^w and o^l denote the model's output activations for the winner and loser branches respectively (for example, o could be the predicted noise residual at a certain diffusion step). Using the chain rule, we have $\nabla_\theta \mathcal{L}^w = J^{w\top} \nabla_o \mathcal{L}^w$ and $\nabla_\theta \mathcal{L}^l = J^{l\top} \nabla_o \mathcal{L}^l$, where J is the Jacobian $\partial o / \partial\theta$ and $\nabla_o \mathcal{L}$ is the gradient of the loss with respect to the model output. Let $g^w = \nabla_o \mathcal{L}^w$ and $g^l = \nabla_o \mathcal{L}^l$ denote the output-space gradients for the winner and the loser. Eq. 14 can then be written as:

$$\lambda \leq \frac{\|\nabla_\theta \mathcal{L}^w\|_2^2}{\nabla_\theta \mathcal{L}^{w\top} \nabla_\theta \mathcal{L}^l} = \frac{g^{w\top}\big(J^w J^{w\top}\big) g^w}{g^{w\top}\big(J^w J^{l\top}\big) g^l} \tag{15}$$

$$= \frac{\|g^w\|_2^2}{g^{w\top} g^l} \cdot \underbrace{\frac{g^{w\top}\big(J^w J^{w\top}\big) g^w}{\|g^w\|_2^2} \Big/ \frac{g^{w\top}\big(J^w J^{l\top}\big) g^l}{g^{w\top} g^l}}_{\rho}. \tag{16}$$

Algorithm 1: Training of Diffusion-SDPO.

Input: Dataset $\mathcal{D} = \{(c, x_0^w, x_0^l)\}$; model ϵ_θ; reference ϵ_{ref}; safety slack $\mu \in [0, 1]$; schedule length T; learning rate η.

while *not converged* **do**

1. Sample $t \sim \text{Uniform}\{0, \dots, T-1\}, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), (c, x_0^w, x_0^l) \sim \mathcal{D}$.
2. Get (x_{t+1}^w, x_{t+1}^l) from Eq. 2 and compute
$$\hat{\epsilon}_\theta^w = \epsilon_\theta(x_{t+1}^w, c, t), \qquad \hat{\epsilon}_\theta^l = \epsilon_\theta(x_{t+1}^l, c, t),$$
$$\hat{\epsilon}_{\text{ref}}^w = \epsilon_{\text{ref}}(x_{t+1}^w, c, t), \qquad \hat{\epsilon}_{\text{ref}}^l = \epsilon_{\text{ref}}(x_{t+1}^l, c, t).$$
3. Get per-branch residual objectives:
$$\mathcal{L}^w = \tfrac{1}{2}\|\hat{\epsilon}_\theta^w - \epsilon\|_2^2 - \tfrac{1}{2}\|\hat{\epsilon}_{\text{ref}}^w - \epsilon\|_2^2,$$
$$\mathcal{L}^l = \tfrac{1}{2}\|\hat{\epsilon}_\theta^l - \epsilon\|_2^2 - \tfrac{1}{2}\|\hat{\epsilon}_{\text{ref}}^l - \epsilon\|_2^2.$$
4. Compute λ_{safe} using Eq. 17: $\lambda_{\text{safe}} = (1-\mu)\|g^w\|_2^2 / g^{w\top} g^l$.
5. Scale only loser gradients: $\mathcal{L}_{\text{scaled}}^l = \mathcal{L}_{\text{detach}}^l + \lambda_{\text{safe}}\big(\mathcal{L}^l - \mathcal{L}_{\text{detach}}^l\big)^\dagger$.
6. Build loss using Eq. 9: $\mathcal{L}_{\text{DPO}} = -\log\sigma\big(-\beta(\mathcal{L}^w - \mathcal{L}_{\text{scaled}}^l)\big)$.
7. Update $\theta \leftarrow \theta - \eta\nabla_\theta\mathcal{L}_{\text{DPO}}$.

Output: Finetuned model ϵ_θ.

$\dagger$: $\mathcal{L}_{\text{detach}}^l$ is a copy of $\mathcal{L}^l$ without gradient flow.

The factor ρ in Eq. 16 encodes local Jacobian geometry. Estimating ρ during training requires parameter–space backpropagation and adds memory usage or wall–clock time (cf. Table 5). To keep the update lightweight, we do not track ρ explicitly. Instead, we absorb it into a scalar *safety slack* $\mu \in [0, 1]$ that contracts the proxy in a controlled way:

$$\lambda_{\text{safe}} = \frac{(1-\mu)\,\|g^w\|_2^2}{g^{w\top} g^l}. \tag{17}$$

This removes any dependence on parameter–space Jacobians and uses only the output–space gradients g^w and g^l that are already computed for the loss, so the additional cost is negligible. The slack μ serves as a robust guardrail under arbitrary local geometry. Larger μ yields a more conservative scaling of the loser branch; smaller μ recovers a more aggressive update. In practice, we find a fixed μ works well (see Sec. 5.3 for ablation on μ), and for an appropriate choice of μ, the output-space scheme yields λ_{safe} trajectories that closely match those obtained from parameter-space gradients (cf. Fig. 2).

During training, we clip λ_{safe} to $[0, 1]$ for stability (if $g^{w\top} g^l \leq 0$, we set $\lambda_{\text{safe}} = 1$). Whenever the loser's error vector has a positive correlation with the winner's error vector ($g^{w\top} g^l > 0$), λ_{safe} provides a finite limit to how strongly we can apply the loser's gradient without risking an increase in the winner's loss. For the logistic DPO objective, we implement this by scaling the backpropagated loser gradient with λ_{safe}. Algorithm 1 summarizes the procedure to integrate SDPO into Diffusion-DPO. For other methods, λ_{safe} is similarly used to scale the loser branch.

Table 1: Reward score comparison on the HPS V2 with SD 1.5. Rows labeled "+ SDPO" report the performance obtained by applying our SDPO to the corresponding base method in the preceding row. †: results from our implementation due to the lack of official code. Best results are in **bold**. Owing to space constraints, the full table is provided in the appendix.

Method	PickScore(↑)	HPS(↑)	Aes.(↑)	CLIP(↑)	IR(↑)
SD 1.5	0.2088	0.2697	5.4933	0.3480	-0.0469
SFT	0.2168	0.2838	5.7851	0.3591	0.6619
Diff.-KTO	0.2164	0.2766	5.6288	0.3420	0.5593
MaPO†	0.2124	0.2760	5.6890	0.3528	0.3308
DPOP†	0.2144	0.2780	5.7071	0.3563	0.3735
Diff.-DPO	0.2131	0.2743	5.6639	0.3552	0.1705
+ **SDPO**	0.2174	0.2827	**5.8744**	0.3600	0.6211
DSPO	0.2168	0.2837	5.8346	0.3598	0.6483
+ **SDPO**	0.2172	0.2847	5.8474	0.3586	0.6578
DMPO†	0.2131	0.2766	5.6538	0.3551	0.3171
+ **SDPO**	**0.2182**	**0.2848**	5.8574	**0.3612**	**0.7061**

5 Experiments

5.1 Experimental Setting

Datasets and Models. Following [21, 48], we finetune Stable Diffusion 1.5 (SD 1.5) and SDXL on preference pairs from the Pick-a-Pic V2 (Pick V2) [18] training set. For evaluation, we use test prompts from Pick V2, HPS V2 [41], and PartiPrompts [46]. Beyond these SD-based backbones, we further study the scalability of our method with respect to both model capacity and preference data volume by extending experiments to two larger DiT-style generators. First, we consider Ovis-U1 [39], a 3.6B unified DiT [27] model that supports both text-to-image synthesis and image editing. Second, to validate generality on a substantially larger model, we apply our approach to FLUX.1-dev [19], a 12B rectified-flow DiT [27] model for text-to-image generation. For this setting, we build an in-house preference dataset of 186K high-quality DPO examples spanning diverse domains.

Training Details and Baselines. We integrate SDPO into Diffusion-DPO [38], DSPO [48], and DMPO [21] implementations and keep their official hyperparameters. All models are finetuned for 2000 steps with a global batch size of 2048. The learning rate is 1×10^{-8} for SD 1.5 and 1×10^{-9} for SDXL. For the safeguard coefficient μ, on SD 1.5 we set 0.9 for Diffusion-DPO+SDPO and DMPO+SDPO, 0.2 for DSPO+SDPO. On SDXL, μ is fixed as 0.6 for all variants. We compare against several baselines: the original pretrained SD 1.5 and SDXL, supervised finetuning (SFT), Diffusion-KTO [22], MaPO [13], DPOP [26], and original Diffusion-DPO, DSPO, DMPO. For FLUX.1-dev, we train at 1024×1024 resolution with a global batch size of 512 and a learning

Table 2: Average win rate comparison (%) over the HPS V2 using SD 1.5. Each row reports *Model 1* vs. *Model 2* on identical prompts. The upper block summarizes SDPO augmentation results (base + SDPO vs. base), and the lower block compares each model against SD 1.5. Values $> 50\%$ indicate that *Model 1* generally outperforms *Model 2*.

Model 1	Model 2	PickScore	HPS	Aes.	CLIP	IR	Mean
SDPO augmentation effect (base+SDPO vs base)							
Diff.-DPO+ SDPO	Diff.-DPO	66.12	78.62	**70.62**	52.50	**71.62**	67.90
DSPO + SDPO	DSPO	52.62	53.00	53.25	48.50	52.38	51.95
DMPO + SDPO	DMPO	**73.50**	**79.25**	67.12	**53.12**	71.50	**68.90**
Versus SD 1.5							
Diff.-DPO	SD 1.5	76.38	69.50	66.88	57.50	63.50	66.75
Diff.-DPO+ SDPO	SD 1.5	80.75	86.25	**83.38**	56.88	79.25	77.30
DSPO	SD 1.5	78.38	86.00	78.75	**58.88**	79.75	76.35
DSPO + SDPO	SD 1.5	81.75	**89.75**	79.12	56.75	80.12	77.50
DMPO	SD 1.5	68.50	74.50	68.38	53.00	69.88	66.85
DMPO + SDPO	SD 1.5	**82.25**	88.12	79.25	58.38	**81.75**	**77.95**

Table 3: Comparison of Ovis-U1 [39] variants on preference, structured alignment, and image editing benchmarks. Higher is better for all metrics.

Model	Preference Eval		Structured Alignment		Image Editing	
	CLIP	HPS V2	GenEval	DPG-Bench	ImgEdit	GEdit-EN
Ovis-U1	0.3188	0.2986	0.89	83.72	4.00	6.42
Ovis-U1 + DPO	0.3192	0.2997	0.88	83.78	4.01	6.43
Ovis-U1 + SDPO	**0.3201**	**0.3082**	**0.91**	**84.84**	**4.11**	**6.60**

rate of 1×10^{-5}. For SDPO, we use a slack of $\mu = 0.99$, which empirically yields effective safe scaling coefficients $\lambda_{\text{safe}} \approx 0.1$ at typical training steps.

Evaluation. We evaluate models on automatic preference metrics, including PickScore [18], HPS V2 [41], LAION Aesthetic Classifier [33], CLIP [29] and ImageReward [43] (IR) scores. Sampling uses a guidance scale of 7.5 and 50 denoising steps. For Ovis-U1 and FLUX.1-dev, we further probe structured text–image alignment using GenEval [9] and DPG-Bench [14]. Since Ovis-U1 also supports editing, we additionally benchmark image-editing quality on ImgEdit [45] and GEdit-EN [24].

5.2 Main Results

Table 1 shows that adding SDPO to Diffusion-DPO, DSPO, and DMPO consistently improves automatic reward metrics under SD 1.5, with DMPO+SDPO typically achieving the best overall scores. Win-rate results on SD 1.5 (Table 2) further confirm that each base method benefits from SDPO and that SDPO variants also outperform the SD 1.5 baseline, indicating stronger preference align-

Table 4: Full reward and structured alignment results for FLUX.1-dev. The row marked with § corresponds to Diff.-DPO trained with a 0.01× learning rate. Higher is better for all metrics.

Method	PickScore	HPS	Aes.	CLIP	IR	GenEval	DPG
FLUX.1-dev	0.2252	0.2889	6.0735	0.3498	0.9916	0.66	85.75
Diff.-DPO	0.1998	0.2467	5.2857	0.2573	-0.7788	0.23	58.97
Diff.-DPO§	0.2253	0.2881	6.0701	0.3501	0.9938	0.66	85.78
Diff.-DPO + SDPO	**0.2314**	**0.2991**	**6.1910**	**0.3566**	**1.1108**	**0.71**	**86.36**

ment without sacrificing quality. On SDXL, the gains are moderate yet consistent (see Appendix), suggesting reliable scaling to larger UNet [32] backbones.

Table 5: Ablation results for winner-preserving rules on SD 1.5 (prompts: HPS V2). We report PickScore/HPS V2, one-step training Time (s) and peak GPU memory (GB) on a single NVIDIA A100 with batch size 16 and 128 gradient accumulation steps in BF16. ‡: fixed λ_{safe} in SDPO. †: λ_{safe} computed with parameter-space gradients.

Method	PickScore (↑)	HPS (↑)	Time (↓)	GPU Mem. (↓)
MaPO	0.2124	0.2760	162	53.8
DPOP	0.2144	0.2780	187	58.5
Diff.-DPO	0.2131	0.2743	183	57.0
Diff.-DPO+SDPO‡	0.2158	0.2803	183	57.0
Diff.-DPO+SDPO†	0.2176	0.2828	314	63.3
Diff.-DPO+SDPO	0.2174	0.2827	184	57.1

On the unified Ovis-U1 model (Table 3), SDPO yields clear improvements in preference metrics and editing scores, demonstrating effectiveness on a DiT backbone that supports both text-to-image generation and image editing. While naive Diffusion-DPO can enlarge preference margins at the expense of fidelity, our safeguarded integrations preserve details and improve prompt adherence across diverse prompts. Visual comparisons (cf. Appendix) align with these quantitative trends, indicating that SDPO stabilizes optimization and enhances perceptual quality.

Furthermore, results on FLUX.1-dev (Table 4) reveals a clear three-regime behavior. At the default learning rate (1×10^{-5}), naive Diffusion-DPO catastrophically degrades the model: all five reward metrics drop, and both GenEval and DPG-Bench scores collapse, suggesting that aggressively enlarging the preference margin without safeguarding the winner branch can push parameters far outside the pretrained basin. When reducing the learning rate by 0.01 (Diff.-DPO§ with 1×10^{-7}), the collapse disappears and metrics revert to near the base FLUX.1-dev level, indicating an overly conservative regime with little effective learning. In contrast, Diff.-DPO + SDPO improves over the base model on *all* reported metrics while keeping the original high learning rate: PickScore, HPS V2, Aesthetics, CLIP, and ImageReward all increase, while GenEval and

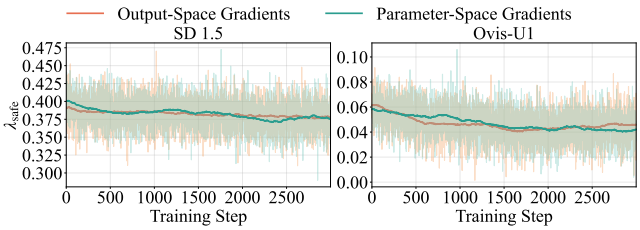

Fig. 2: Training dynamics of λ_{safe} on SD 1.5 (left) and Ovis-U1 (right) with two computation schemes (using output-space gradients *vs.* parameter-space gradients). The trajectories closely match throughout training, and the output-space variant requires substantially less computation while maintaining comparable aesthetic rewards (see Table 5).

DPG-Bench rise in tandem. Taken together, these improvements suggest that SDPO provides stable preference optimization on high-capacity rectified-flow backbones and, crucially, translates those gains into both better perceptual quality and stronger structured text-image alignment.

5.3 Ablation Study

Modular Ablation & Computational Cost. Table 5 compares winner preserving strategies when embedded into MaPO, DPOP, and Diffusion-DPO. MaPO applies a fixed winner weight and removes the reference model, which weakens calibration of absolute error though it requires less computation and GPU memory. DPOP protects the winner through thresholded update filtering, but this rule was designed for autoregressive language models and does not fully match diffusion training dynamics. Our SDPO preserves the winner by rescaling the loser update with a safeguard coefficient λ_{safe} selected in the output space according to directional alignment. A fixed λ_{safe}—that is, holding λ_{safe} constant throughout training (see Table 5, row 4)—already improves over MaPO and DPOP on PickScore and HPS, whereas allowing λ_{safe} to adapt during training yields further gains. These improvements support the hypothesis that output–space selection of λ_{safe} stabilizes the winner while maintaining pressure to enlarge the preference margin.

Fig. 2 compares two mechanisms for computing the dynamic λ_{safe}: using output-space gradients and using parameter-space gradients. The output-space trajectory closely matches the parameter-space trajectory in both level and trend, indicating similar effectiveness. Because it reuses signals from the standard backward pass and only performs lightweight reductions, the output-space variant adds essentially no runtime or memory overhead relative to the Diffusion-DPO baseline. In contrast, as shown in Table 5, computing λ_{safe} from parameter-space gradients requires extra vector-Jacobian products and temporary buffers for per-parameter inner products, increasing training time by about 72% and peak memory by about 11%.

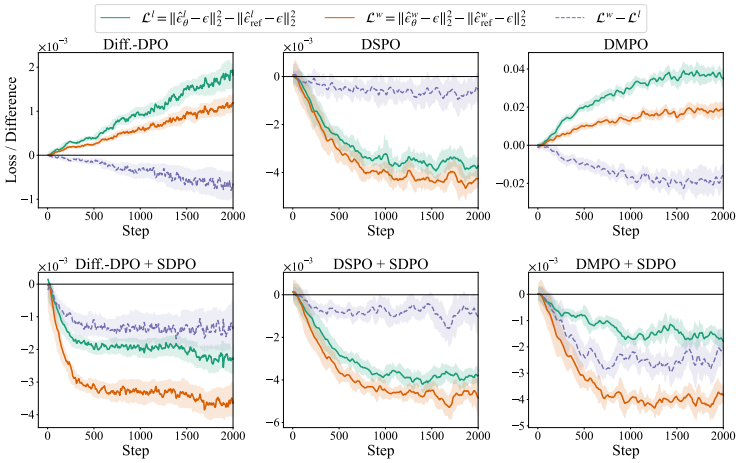

Fig. 3: Training dynamics across three objectives with and without SDPO on SD 1.5.

Note that the two curves in Fig. 2 use different values of μ, which indicates that by adjusting μ, the output-space scheme can approximate the behavior of parameter-space estimation. Although computing λ_{safe} in the output space is an approximation, our empirical results show that proper tuning of μ effectively compensates for this mismatch, allowing the scaled gradients to closely match the parameter-space formulation while retaining the same training efficiency.

Why does SDPO generalize across DPO variants? Fig. 3 contrasts the training dynamics of Diff.-DPO, DSPO, and DMPO with or without SDPO. Without SDPO, $\mathcal{L}^w - \mathcal{L}^l$ decreases as expected, whereas $\mathcal{L}^w$ remains nondecreasing and drifts upward in Diff.-DPO and DMPO, indicating unstable optimization. With SDPO, $\mathcal{L}^w$ drops early and remains low, $\mathcal{L}^l$ declines smoothly without overshoot, and $\mathcal{L}^w - \mathcal{L}^l$ decreases steadily to a plateau. For DSPO, which already regularizes branch imbalance via its score preference objective and progressively increases the weight on the winner branch, adding SDPO causes no degradation and typically yields slightly improved reward results (cf. Table 1).

We observe a shared qualitative profile across the three SDPO-augmented settings: after basic rescaling, trajectories from different objectives largely overlap. $\mathcal{L}^w$ follows a monotone, fast-then-slow descent, $\mathcal{L}^l$ descends smoothly, and their gap grows in a stable manner across timesteps. This empirical regularity suggests that SDPO successfully corrects harmful update directions and magnitudes by acting on gradient geometry, thereby normalizing training dynamics.

Sensitivity of Hyperparameters. We plot a figure to analyze the sensitivity to the safeguard slack μ in the Appendix. Overall, μ exhibits broad, gently convex optima rather than requiring sharp tuning: both HPS V2 and PickScore remain close to their best values across wide intervals, suggesting that SDPO is governed primarily by gradient–alignment geometry in output space rather

than the specific loss form. On SDXL, all SDPO–augmented objectives maintain near–optimal performance over a wide range centered around $\mu \approx 0.6$, consistent with a smoother, higher-capacity optimization landscape.

On SD 1.5, the more aggressive baselines (Diffusion-DPO and DMPO) benefit from a larger slack μ, which further contracts the loser contribution whenever its direction conflicts with the winner. In contrast, DSPO already progressively assigns greater weight to the winner branch during training, so a smaller slack is sufficient. Practically, μ can be chosen via an early-phase heuristic: increase μ if the winner loss $\mathcal{L}^w$ drifts upward or oscillates; decrease μ in small steps if the preference margin stalls despite a stable $\mathcal{L}^w$. Following this rule of thumb, we adopt a single default on SDXL ($\mu = 0.6$ for all objectives) and two defaults on SD 1.5 ($\mu = 0.9$ for Diffusion–DPO and DMPO; $\mu = 0.2$ for DSPO).

SDPO preserves gains with longer training. We further examine whether the safeguarding mechanism remains effective under extended training (Fig. 4). The baseline exhibits a non-monotonic trajectory: both HPS V2 and PickScore increase early, then plateau or decline as training proceeds. In contrast, augmenting Diffusion-DPO with SDPO yields continued gains that stabilize at a higher level, widening the gap over the baseline as steps increase. Qualitative examples in Fig. 1 corroborate this trend: baseline generations develop saturation and texture artifacts under prolonged training, whereas SDPO maintains crisp structure and consistent style. Overall, SDPO sustains improvements in preference metrics without sacrificing perceptual quality during longer training. This follows from its winner-preserving constraint, which limits the influence of loser-branch gradients and ensures that the winner loss does not increase.

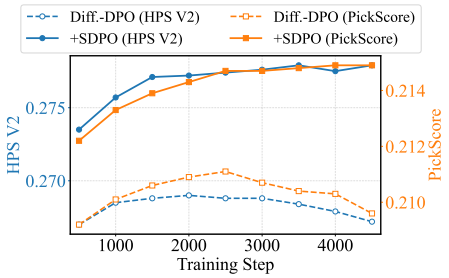

Fig. 4: Comparison of Diffusion–DPO with and without SDPO under longer training on SD 1.5 (Test prompts: Pick V2).

6 Conclusions and Limitations

In this paper, we presented Diffusion-SDPO, a safeguarded preference optimization scheme that stabilizes DPO-style diffusion finetuning by preserving the preferred branch while improving preference matching. The method scales the loser gradient by its alignment with the winner and guarantees, to first order, that the winner's reconstruction loss does not increase. Across SD 1.5, SDXL (UNet), and Ovis-U1, FLUX.1-dev (DiT), our method yields consistent improvements on automated preference, aesthetic, and prompt-alignment metrics with negligible computational overhead, while remaining model-agnostic, straightforward to implement, and applicable to multiple DPO variants.

However, the safeguard is derived from a first-order approximation, and its validity weakens when the loss landscape exhibits strong curvature. Moreover, the coefficient ρ (cf. Eq. 16) used to estimate gradient alignment can be noisy or biased, which leads to imperfect scaling of loser gradients. Nevertheless, our empirical results indicate that with a suitably chosen μ, this estimation is accurate enough in practice and the safeguard behaves as intended. Future work includes developing second-order or trust-region safeguards to maintain robust winner preservation under diverse training dynamics.

References

1. Bradley, R.A., Terry, M.E.: Rank analysis of incomplete block designs: I. the method of paired comparisons. Biometrika **39**(3/4), 324–345 (1952)
2. Cho, J.H., Oh, J., Kim, M., Lee, B.J.: Rethinking DPO: The role of rejected responses in preference misalignment. arXiv:2506.12725 (2025)
3. Chowdhury, S.R., Kini, A., Natarajan, N.: Provably robust DPO: aligning language models with noisy feedback. In: ICML. pp. 42258 – 42274 (2024)
4. Christiano, P.F., Leike, J., Brown, T.B., Martic, M., Legg, S., Amodei, D.: Deep reinforcement learning from human preferences. In: NeurIPS. pp. 4302–4310 (2017)
5. Croitoru, F.A., Hondru, V., Ionescu, R.T., Shah, M.: Diffusion models in vision: A survey. IEEE TPAMI **45**(9), 10850–10869 (2023)
6. Dhariwal, P., Nichol, A.: Diffusion models beat GANs on image synthesis. In: NeurIPS. pp. 8780–8794 (2021)
7. Esser, P., Kulal, S., Blattmann, A., et al.: Scaling rectified flow Transformers for high-resolution image synthesis. In: ICML. pp. 12606–12633 (2024)
8. Gambashidze, A., Kulikov, A., Sosnin, Y., Makarov, I.: Aligning diffusion models with noise-conditioned perception. arXiv:2406.17636 (2024)
9. Ghosh, D., Hajishirzi, H., Schmidt, L.: GENEVAL: An object-focused framework for evaluating text-to-image alignment. In: NeurIPS (2023)
10. Google: Introducing gemini 2.5 flash image (aka nano-banana). `https://developers.googleblog.com/en/introducing-gemini-2-5-flash-image/` (2025), accessed: 27 June 2026
11. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: NeurIPS. pp. 6840 – 6851 (2020)
12. Ho, J., Salimans, T.: Classifier-free diffusion guidance. arXiv:2207.12598 (2022)
13. Hong, J., Paul, S., Lee, N., Rasul, K., Thorne, J., Jeong, J.: Margin-aware preference optimization for aligning diffusion models without reference. In: ICLR Workshop (2025)
14. Hu, X., Wang, R., Fang, Y., Fu, B., Cheng, P., Yu, G.: ELLA: Equip diffusion models with LLM for enhanced semantic alignment. arXiv:2403.05135 (2024)
15. Huang, A., Zhan, W., Xie, T., Lee, J.D., Sun, W., Krishnamurthy, A., Foster, D.J.: Correcting the mythos of KL-regularization: Direct alignment without overparameterization via chi-squared preference optimization. In: ICLR. pp. 1–16 (2025)
16. Jo, J., Lee, S., Hwang, S.J.: Score-based generative modeling of graphs via the system of stochastic differential equations. In: ICML. pp. 10362–10383 (2022)
17. Karras, T., Aittala, M., Laine, S., Aila, T.: Elucidating the design space of diffusion-based generative models. In: NeurIPS. pp. 26565 – 26577 (2022)

18. Kirstain, Y., Polyak, A., Singer, U., Matiana, S., Penna, J., Levy, O.: Pick-a-Pic: An open dataset of user preferences for text-to-image generation. In: NeurIPS. pp. 36652 – 36663 (2023)
19. Labs, B.F.: FLUX.1-dev. `https://huggingface.co/black-forest-labs/FLUX.1-dev` (2024), accessed: 27 June 2026
20. Lan, G., Zhang, S., Wang, T., et al.: MaPPO: Maximum a posteriori preference optimization with prior knowledge. arXiv:2507.21183 (2025)
21. Li, B., Xu, M., Dang, M., Ermon, S.: Divergence minimization preference optimization for diffusion model alignment. arXiv:2507.07510 (2025)
22. Li, S., Kallidromitis, K., Gokul, A., Kato, Y., Kozuka, K.: Aligning diffusion models by optimizing human utility. In: NeurIPS. pp. 24897 – 24925 (2025)
23. Lipman, Y., Chen, R.T., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. In: ICLR. pp. 1–13 (2023)
24. Liu, S., Han, Y., Xing, P., et al.: Step1X-Edit: A practical framework for general image editing. arXiv:2504.17761 (2025)
25. Liu, X., Gong, C., Liu, Q.: Flow straight and fast: Learning to generate and transfer data with rectified flow. In: ICLR. pp. 1–15 (2023)
26. Pal, A., Karkhanis, D., Dooley, S., Roberts, M., Naidu, S., White, C.: Smaug: Fixing failure modes of preference optimisation with dpo-positive. arXiv:2402.13228 (2024)
27. Peebles, W., Xie, S.: Scalable diffusion models with Transformers. In: ICCV. pp. 4172–4182 (2023)
28. Podell, D., English, Z., Lacey, K., et al.: SDXL: Improving latent diffusion models for high-resolution image synthesis. In: ICLR. pp. 1–13 (2024)
29. Radford, A., Kim, J.W., Hallacy, C., et al.: Learning transferable visual models from natural language supervision. In: ICML. pp. 8748–8763 (2021)
30. Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C.D., Finn, C.: Direct preference optimization: your language model is secretly a reward model. In: NeurIPS. pp. 53728–53741 (2023)
31. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: CVPR. pp. 10674–10685 (2022)
32. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional networks for biomedical image segmentation. In: MICCAI. pp. 234–241 (2015)
33. Schuhmann, C., Vencu, R., Beaumont, R., et al.: LAION-Aesthetics: Predicting the aesthetic quality of images. `https://laion.ai/blog/laion-aesthetics/` (2022), accessed: 27 June 2026
34. Shekhar, S., Singh, S., Zhang, T.: SEE-DPO: Self entropy enhanced direct preference optimization. TMLR (2025)
35. Sohl-Dickstein, J., Weiss, E.A., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. In: ICML. p. 2256–2265 (2015)
36. Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. In: ICLR. pp. 1–12 (2022)
37. Tamboli, D., Chakraborty, S., Malusare, A., Banerjee, B., Bedi, A.S., Aggarwal, V.: BalancedDPO: Adaptive multi-metric alignment. arXiv:2503.12575 (2025)
38. Wallace, B., Dang, M., Rafailov, R., et al.: Diffusion model alignment using direct preference optimization. In: CVPR. pp. 8228–8238 (2024)
39. Wang, G.H., Zhao, S., Zhang, X., et al.: Ovis-U1 technical report. arXiv:2506.23044 (2025)

40. Wang, R., Sun, Q., Song, C., Wu, J., Chen, T., Zeng, Z., Li, Y.: Linear preference optimization: Decoupled gradient control via absolute regularization. arXiv:2508.14947 (2025)
41. Wu, X., Hao, Y., Sun, K., Chen, Y., Zhu, F., Zhao, R., Li, H.: Human Preference Score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. arXiv:2306.09341 (2023)
42. Xiao, T., Yuan, Y., Zhu, H., Li, M., Honavar, V.G.: Cal-DPO: Calibrated direct preference optimization for language model alignment. In: NeurIPS. pp. 114289–114320 (2024)
43. Xu, J., Liu, X., Wu, Y., et al.: ImageReward: Learning and evaluating human preferences for text-to-image generation. In: NeurIPS. pp. 15903–15935 (2024)
44. Yang, C., Jia, R., Gu, N., et al.: Orthogonal finetuning for direct preference optimization. arXiv:2409.14836 (2024)
45. Ye, Y., He, X., Li, Z., et al.: ImgEdit: A unified image editing dataset and benchmark. arXiv:2505.20275 (2025)
46. Yu, J., Xu, Y., Koh, J.Y., et al.: Scaling autoregressive models for content-rich text-to-image generation. arXiv:2206.10789 (2022)
47. Zhao, W.X., Zhou, K., Li, J., et al.: A survey of large language models. arXiv:2303.18223 (2023)
48. Zhu, H., Xiao, T., Honavar, V.G.: DSPO: Direct score preference optimization for diffusion model alignment. In: ICLR. pp. 1–13 (2025)