CONCORDANCE-GUIDED ROLLOUT REWEIGHTING AND RESAMPLING FOR MULTI-REWARD IMAGE GENERATION POST-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-reward post-training of text-to-image models commonly reduces several rewards to a single scalar reward. However, the resulting scalarized advantage can disagree in sign with single-reward advantages, concealing conflicting supervision within a rollout group generated for the same prompt. To address this issue, we introduce Scalar Reward Concordance (SRC), a continuous score that quantifies the weakest active-reward support for each rollout's scalarized advantage under a fixed reward scalarization. Derived from within-rollout gradient alignment for the GRPO family, SRC uses only observed reward values. It identifies concordant positive and negative supervision, including rollouts with jointly high or jointly low rewards. Building on this measure, we develop two methods: SRC-Reweight redistributes training weights over existing rollouts without additional sampling, while SRC-Evolve draws additional on-policy rollouts and selects a fixed-size population using SRC. Both methods integrate with GRPO-Guard and DiffusionNFT while preserving the prescribed reward scalarization. We further introduce Joint Success Rate (JSR) to compare models in terms of their ability to satisfy multiple reward requirements simultaneously. Experiments covering five training rewards and three held-out evaluation rewards show improved aesthetics with prompt following broadly retained. SRC-Evolve also improves the joint success rate on the applicable training rewards relative to baselines with matched numbers of generated rollouts.

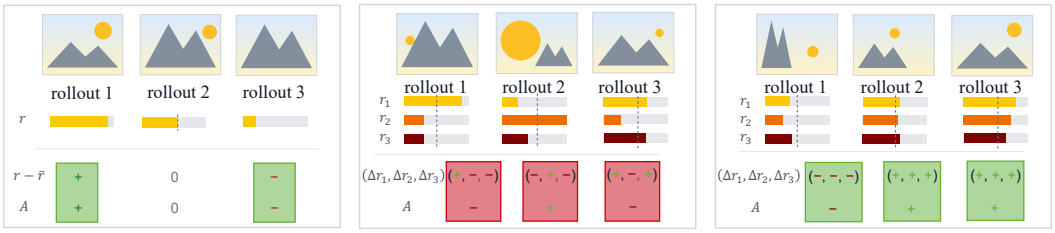
1 INTRODUCTION

Text-to-image models are expected to satisfy several requirements at once: an image should follow the prompt, render objects and text correctly, and remain visually appealing. Reward models capture different aspects of these requirements and may rank the same images differently (Xu et al., 2023; Kirstain et al., 2023; Zhang et al., 2024; Ghosh et al., 2023). Reward-based post-training adapts diffusion and flow models to such objectives (Black et al., 2024; Liu et al., 2025; Zheng et al., 2025). When several rewards are used, a common approach combines them through a weighted sum into a *scalar reward*. However, improving this aggregate does not necessarily adequately improve every reward. A high aggregate score can hide poor performance on an individual reward, so training signals based on that score can conceal conflicts among rewards.

Existing work addresses multi-reward conflict in several ways. In text-to-image post-training, CMO adjusts objective weights using reward correlations among candidates generated for the same prompt (Wi et al., 2026). MARBLE retains reward-specific advantages and gradients and coordinates them in gradient space (Zhao et al., 2026). These studies examine how multiple reward objectives can be weighted and optimized together. In this work, we take a rollout-level perspective on multi-reward post-training, where a *rollout* is a generated candidate together with its generation trajectory. We ask: **Can we improve multi-reward post-training by identifying and prioritizing rollouts whose scalar supervision agrees with all active rewards, while keeping the reward scalarization fixed?**

To answer this question, consider how training signals are formed within a *rollout group*, the set of rollouts generated for the same prompt. Standardizing scalar rewards within this group gives *scalarized advantages*, where positive values encourage a rollout, while negative values discourage it. Consider an image that scores above the group mean on PickScore but below it on CLIPScore.

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

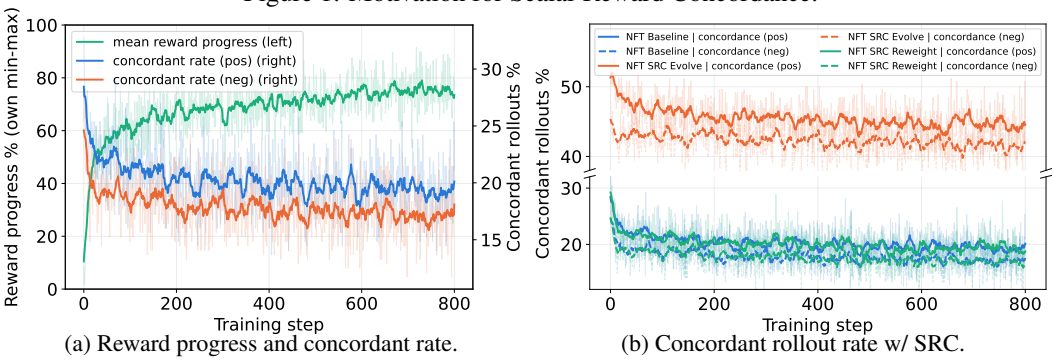


(a) **Single reward.** The single-reward advantage and scalarized advantage necessarily agree in sign.

(b) **Multiple rewards: disagreement.** One or more active single-reward advantages disagree in sign with the scalarized advantage A.

(c) **Multiple rewards: concordance.** Every active single-reward advantage agrees in sign with the scalarized advantage A.

Figure 1: Motivation for Scalar Reward Concordance.



(a) Reward progress and concordant rate.

(b) Concordant rollout rate w/ SRC.

Figure 2: **Concordant rollout rate during multi-reward post-training (DiffusionNFT).** (a) Reward progress rises steeply early, when concordant rollout fractions are higher. (b) SRC-Evolve has higher observed concordant fractions, while SRC-Reweight remains close to the baseline.

Its scalarized advantage can still be positive, so the scalar training signal encourages a rollout that falls below average on one of the rewards. Aggregation thus hides which rewards support the scalar training signal and which oppose it.

To expose this disagreement, we first standardize each reward separately within the group to obtain its *single-reward advantage*. With a single reward, this advantage and the scalarized advantage agree in sign (Figure 1a). With multiple rewards, they can disagree (Figure 1b). We call a rollout's supervision *concordant* when all its active single-reward advantages agree in sign with its scalarized advantage, where active rewards are those with positive weights in the scalar reward (Figure 1c). Concordance includes both positive supervision from rollouts above the group mean on every active reward and negative supervision from those below every mean. It therefore describes agreement among training signals, rather than absolute image quality.

To illustrate the potential of concordant supervision in practice, we track the fraction of concordant rollouts as post-training progresses in DiffusionNFT. At each training step, we measure the fraction of rollouts that participate in the training whose active single-reward advantages all agree in sign with their scalarized advantage. We count concordant rollouts with positive and negative scalarized advantages separately, dividing each count by the full rollout-group size. These statistics cover all three prompt sources described in Section 5.1. Reward progress averages the reward curves after each is min–max normalized using its own minimum and maximum over the training run. Early in training, the positive and negative concordant rollout fractions are higher, while the normalized reward curve rises steeply. Later, these fractions decline and the reward curve flattens (Figure 2a). This temporal pattern suggests two ways to use concordant rollouts: give those already in a sampled group more training weight, or draw and select additional on-policy rollouts to increase their fraction in the training group.

To prioritize rollouts under either intervention, we introduce *Scalar Reward Concordance* (SRC), a continuous score of the weakest active-reward support for a rollout's scalarized advantage. By taking the weakest support, SRC exposes disagreement that stronger support from other rewards could mask. A negative score identifies a rollout for which at least one active reward opposes the scalar training signal.

We then devise two methods based on SRC. **SRC-Reweight** assigns larger training weights to rollouts with higher SRC, emphasizing concordant supervision. **SRC-Evolve** draws additional on-

policy rollouts and selects a fixed-size population using scalar elitism and SRC. Figure 2b compares concordant rollout fractions during DiffusionNFT post-training. For SRC-Evolve, group statistics are recomputed on the final selected rollout group before measuring concordance. SRC-Evolve maintains higher fractions than the baseline for both positive and negative scalarized advantages. SRC-Reweight redistributes training mass over the sampled rollouts without directly changing their unweighted concordant fractions; its measured trajectories remain close to the baseline.

We evaluate both methods under DiffusionNFT and GRPO-Guard using individual rewards and *Joint Success Rate* (JSR), the probability that one image simultaneously meets all reward thresholds. Across both backbones, the results show improved aesthetics while prompt following is broadly retained. SRC-Evolve also improves the joint success rate on the applicable training rewards relative to baselines with matched numbers of generated rollouts.

Our contributions are threefold:

- We derive SRC, a continuous rollout-level score that quantifies the weakest active-reward support for a rollout's scalarized advantage using only observed reward values.
- We develop SRC-Reweight to redistribute training weights over sampled rollouts and SRC-Evolve to select a fixed-size population using additional on-policy sampling, scalar elitism, and SRC.
- We evaluate both methods under DiffusionNFT and GRPO-Guard through individual rewards, JSR, rollout-group diagnostics, and comparisons with matched rollout counts.

2 Related Work

Reward-based post-training for visual generative models. DDPO and DPOK develop online reinforcement learning for diffusion models (Black et al., 2024; Fan et al., 2023). Differentiable alternatives backpropagate reward gradients through the sampling chain for image generation (Clark et al., 2024), while Diffusion-DPO learns from offline human or AI preference pairs (Wallace et al., 2024). Flow-GRPO enables online group-relative RL for flow-matching models (Liu et al., 2025); DiffusionNFT instead contrasts positive and negative generations in a forward-process matching objective (Zheng et al., 2025). We apply SRC-based rollout weighting and selection to GRPO-Guard (Wang et al., 2026) and DiffusionNFT while preserving each backbone's training signal construction.

Multi-reward objectives and optimization. Early scalarization offers a simple interface to multiple rewards, but its behavior depends on reward scale, variance, and conflict. At the objective level, APEX uses adaptive normalization and dynamic objective priorities, whereas CMO estimates correlations among reward dimensions across a rollout group and assigns larger weights to objectives that are difficult to satisfy jointly (Chen et al., 2026b; Wi et al., 2026). These methods adjust reward-dimension weights rather than rollout weights under a fixed scalar reward. Multi-GRPO constructs objective-wise advantages (Lyu et al., 2025), while LucidNFT normalizes objectives before fusion for real-world image super-resolution (Fei et al., 2026). Pareto-based methods guide selection or construct training targets: Parrot performs batch-wise Pareto selection (Lee et al., 2024), CaPO calibrates reward scores and builds preference pairs from Pareto frontiers (Lee et al., 2025), and Flow-Multi masks advantages using non-dominated filtering within Flow-GRPO (Lee & Choi, 2026). PG-OT transports a dominated distribution toward a prompt-specific Pareto frontier (Ba et al., 2026). At the gradient level, MARBLE retains reward-specific advantages and gradients, then solves a gradient-space coordination problem (Zhao et al., 2026). Our focus is whether each active single-reward advantage of a rollout agrees in sign with its scalarized advantage. Whereas Pareto dominance compares reward vectors between candidates, SRC measures this agreement relative to group statistics. Rollouts with jointly low rewards can also have positive SRC, so concordance does not imply high reward or non-dominance.

Preference conditioning and capability consolidation. ParetoSlider conditions a single post-trained policy on preference weights to expose a continuous family of trade-offs at inference time (Golan et al., 2026). To consolidate reward-specific capabilities, Rewarded Soups interpolates parameters of models fine-tuned for different rewards (Rame et al., 2023), while MapReduce LoRA trains reward-specific adapters and iteratively merges them into a shared model (Chen et al., 2026a). Flow-OPD and DiffusionOPD distill specialist capabilities into a shared student (Fang et al., 2026; Li et al., 2026b). Our interventions instead act on rollout groups during joint multi-reward post-training.

Rollout-level weighting and selection. PODS generates a larger set of language-model rollouts and updates on a reward-diverse subset selected to maximize scalar reward variance (Xu et al., 2025). In molecular generation, DrugR assigns sample-level weights using Pareto membership and distance to Pareto anchors within an on-policy rollout batch (Liu et al., 2026a). SRC instead provides a continuous score of each rollout's weakest active-reward support for its scalarized advantage. With the prescribed scalar reward fixed, SRC-Reweight retains all sampled rollouts and assigns normalized training weights using SRC; SRC-Evolve combines SRC with scalar elitism to select a fixed-size population after additional on-policy resampling.

3 Scalar Reward Concordance

We measure whether each active reward supplies rollout-wise supervision consistent with the scalar multi-reward update. We derive the measure for the GRPO family (Shao et al., 2024); a local extension to DiffusionNFT is given in Appendix A.3. Rewards are treated only as observed scalar scores: we assume neither differentiability nor any semantics, correlation, or objective-gradient geometry.

For a rollout group $G = \{(\tau_i, \mathbf{r}_i)\}_{i=1}^{K}$ from one prompt, ignoring clipping and an independent KL term, the reward-dependent GRPO loss and likelihood ratio are

$$\mathcal{L}_{\mathrm{GRPO}}(\theta) = -\frac{1}{K}\sum_{i=1}^{K}\rho_i(\theta)A_i, \qquad \rho_i(\theta) = \frac{\pi_\theta(\tau_i \mid p)}{\pi_{\mathrm{old}}(\tau_i \mid p)}. \tag{1}$$

Define the shared likelihood-ratio base direction as

$$h_i := \nabla_\theta \rho_i(\theta) = \rho_i(\theta)\nabla_\theta \log \pi_\theta(\tau_i \mid p), \tag{2}$$

A rollout with scalarized advantage A_i therefore has ascent-oriented gradient $A_i h_i$. Any token-, timestep-, or trajectory-level aggregation is absorbed into h_i.

For reward k, define its single-reward advantage z_{ik} by standardizing the reward within the rollout group:

$$\bar{r}_k = \frac{1}{K}\sum_{j=1}^{K} r_{jk}, \qquad \sigma_k = \left(\frac{1}{K}\sum_{j=1}^{K}(r_{jk} - \bar{r}_k)^2\right)^{1/2}, \qquad z_{ik} = \frac{r_{ik} - \bar{r}_k}{\sigma_k}. \tag{3}$$

We assume $\sigma_k > 0$ for each active reward $k \in \mathcal{I} := \{k : w_k > 0\}$. Training on reward k alone would use the single-reward advantage z_{ik} and rollout gradient

$$g_{ik}^{\mathrm{single}} := z_{ik} h_i. \tag{4}$$

For the nonnegative scalarization $R_i = \mathbf{w}^\top \mathbf{r}_i$, define the scalarized advantage A_i by standardizing the scalar reward within the same group:

$$\bar{R} = \frac{1}{K}\sum_{j=1}^{K} R_j, \qquad \sigma_R = \left(\frac{1}{K}\sum_{j=1}^{K}(R_j - \bar{R})^2\right)^{1/2}, \qquad A_i = \frac{R_i - \bar{R}}{\sigma_R}. \tag{5}$$

We assume $\sigma_R > 0$. The standardized variables satisfy $\sigma_R A_i = \sum_k w_k \sigma_k z_{ik}$. The corresponding multi-reward rollout gradient is

$$g_i^{\mathrm{multi}} := A_i h_i. \tag{6}$$

Their within-rollout inner product is therefore

$$\left\langle g_{ik}^{\mathrm{single}}, g_i^{\mathrm{multi}} \right\rangle = z_{ik} A_i \|h_i\|_2^2. \tag{7}$$

Retaining the declared scalarization importance w_k, we define the reward-wise raw contribution

$$c_{ik}^{\mathrm{raw}} := w_k z_{ik} A_i. \tag{8}$$

For $h_i \neq \mathbf{0}$, the sign of this raw contribution is exactly the sign of the inner product in Equation 7. Hence $c_{ik}^{\mathrm{raw}} > 0$ means that reward k and the multi-reward objective give rollout i the same update

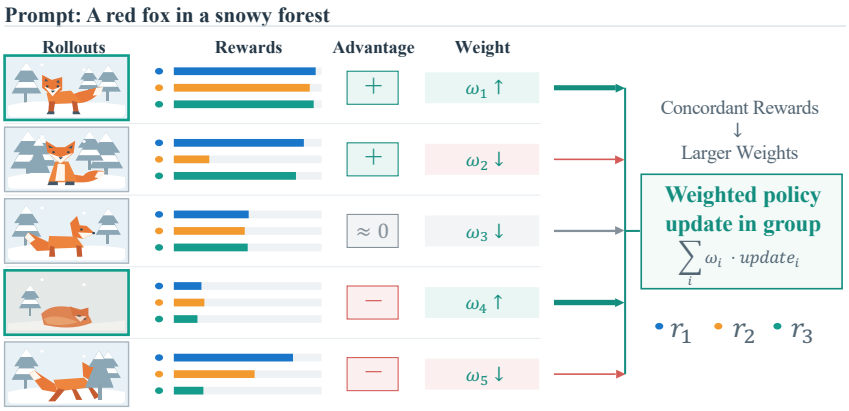


Figure 3: **SRC-Reweight overview.** Five rollouts sampled from the same prompt receive weights according to their reward concordance. Rollouts with jointly high or jointly low rewards receive larger weights, while retaining their positive or negative scalarized-advantage signs, respectively. Reward bars are illustrative.

sign, whereas $c_{ik}^{\text{raw}} < 0$ means that they disagree. This uses no gradient of the reward itself and makes no claim about cross-reward or cross-rollout parameter-space geometry. Its group-level reward-correlation identity is given in Appendix A.1.

To limit the influence of large $|A_i|$ while preserving local discrimination around $A_i = 0$, we saturate only the scalarized-advantage factor in c_{ik}^{raw}:

$$c_{ik}^{\text{sat}} := c_{ik}^{\text{raw}} \frac{1}{|A_i| + 1 + \epsilon} = w_k z_{ik} \frac{A_i}{|A_i| + 1 + \epsilon}, \tag{9}$$

and define SRC as the weakest active saturated contribution:

$$\text{SRC}(i; G) := \min_{k \in \mathcal{I}} c_{ik}^{\text{sat}}. \tag{10}$$

Here $\epsilon > 0$ is a dimensionless numerical safeguard. The exact factorization and limiting behavior are given in Appendix A.2.

Positive SRC means that every active single-reward advantage agrees in sign with the scalarized advantage; negative SRC means that at least one active single-reward advantage disagrees in sign. A value near zero indicates a small scalarized advantage or a small-magnitude single-reward advantage at the bottleneck. SRC is not an absolute quality measure: both a jointly high rollout, which supplies concordant positive supervision, and a jointly low rollout, which supplies concordant negative supervision, can score positively. It is also group-conditioned, so the same reward vector can receive a different SRC score in a different rollout group, even for the same prompt.

4 Concordance-Guided Rollout Reweighting and Resampling

Section 3 measures whether each active single-reward advantage of a rollout agrees in sign with its scalarized advantage. We use this score to redistribute training mass toward rollouts with larger SRC. We first develop SRC-Reweight for the sampled rollouts, and then SRC-Evolve for the setting where additional sampling budget is available.

4.1 SRC-Reweight: Reweighting Observed Rollouts

Given an on-policy group $G = \{(\tau_i, \mathbf{r}_i)\}_{i=1}^{K}$ from one prompt, SRC-Reweight preserves all observed rollouts but emphasizes those whose active single-reward advantages agree in sign with their scalarized advantage, as illustrated in Figure 3. We set $\widehat{s}_i = \text{SRC}(i; G)$ using Equation 10. We use the baseline's fixed reward scalarization.

Lower SRC indicates weaker agreement at the least concordant active reward; in particular, a negative score means that at least one single-reward advantage disagrees in sign with the rollout's scalarized advantage. To reduce the training mass assigned to such conflicting supervision, we use a monotone

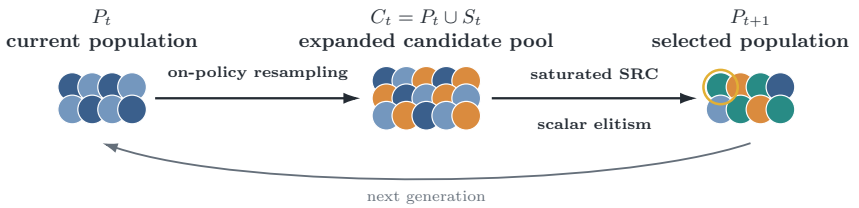


Figure 4: **SRC-Evolve dataflow.** On-policy resampling expands the current population into a candidate pool. Saturated SRC with scalar elitism selects the next fixed-size population, which seeds the following generation.

softmax map and assign

$$\omega_i = \frac{1-\lambda}{K} + \lambda \frac{\exp(\widehat{s}_i/\tau)}{\sum_{j=1}^{K} \exp(\widehat{s}_j/\tau)}, \qquad \tau > 0, \quad 0 \le \lambda < 1. \tag{11}$$

The uniform floor $(1-\lambda)/K$ ensures that every rollout retains nonzero mass, and $\lambda = 0$ recovers the uniform baseline. The softmax component can be derived by maximizing expected concordance while regularizing its departure from the original uniform group; see Appendix B.1.

For GRPO-Guard, we weight the scalarized advantage as $A_i^{\mathrm{SRC}} = K\omega_i A_i$, where K compensates for the baseline's mean reduction; an independent KL term remains uniformly aggregated. For DiffusionNFT, we use ω_i to aggregate the per-rollout policy losses. Further details are given in Appendix B.2. SRC-Reweight adds only $O(Kd)$ reward-space computation and requires no extra denoising or reward-model evaluation.

4.2 SRC-Evolve: Resampling-based rollout augmentation

SRC-Reweight reweights the contributions of sampled rollouts without increasing the sampling budget. When additional sampling budget is available, we can select the training group from a larger set of candidates. We propose SRC-Evolve, a population-based method for this selection. In each generation, it draws additional on-policy rollouts, merges them with the current population, and then keeps K candidates for the policy update.

For each prompt, SRC-Evolve initializes a population of K rollouts. At each generation, it samples M additional complete on-policy rollouts from the current policy and evaluates them with the original reward models. Let $C_t = P_t \cup S_t$ be the merged pool of the current population and the newly resampled rollouts. We compute the reward means and standard deviations from C_t and keep these statistics fixed throughout selection. Using these candidate-pool statistics, SRC-Evolve directly reuses the saturated SRC score from Equation 10 as its fitness, $f_i = \mathrm{SRC}(i; C_t)$. Because SRC is not an absolute quality measure and alone does not guarantee rollout quality, we use scalar elitism as a safeguard: we retain the highest-scalar-reward candidate $e^* = \arg\max_{i \in C_t} R_i$ and fill the remaining slots by fitness:

$$P_{t+1} = \{e^*\} \cup \mathrm{Top}_{K-1}\left(C_t \setminus \{e^*\}; f_i\right). \tag{12}$$

This safeguard preserves the highest observed scalar reward across generations. The selected population becomes the current population of the next generation. After T generations (typically 1 or 2), we recompute group reward statistics and the backbone's scalarized advantages or NFT normalizer on the final population P_T for the policy update. The complete procedure is given in Algorithm 1 in Appendix B.3, along with selector details and tie-breaking. To evaluate SRC-Evolve's sampling efficiency, Table 1 compares it with the corresponding baseline using the same number of generated rollouts as SRC-Evolve ($3K$ per PaP prompt, with K per GenEval or OCR prompt in both methods).

5 Experiments

5.1 Experimental setup

We implement and conduct all of our experiments based on the open-source Flow-Factory framework (Ping et al., 2026). We use Stable Diffusion 3.5 Medium (SD3.5-M) (Stability AI, 2024) as the base model and fine-tune LoRA parameters (Hu et al., 2022). We also evaluate Stable Diffusion XL (SD-XL) (Podell et al., 2024), Stable Diffusion 3.5 Large (SD3.5-L) (Stability AI, 2024), and FLUX.1-dev (FLUX.1-Dev) (Black Forest Labs, 2024) as comparison base models. We draw prompts from three sources covering generic preference, compositional accuracy, and rendered text:

Table 1: **Mean reward results on evaluation set**. Light gray marks training rewards; darker gray marks the rewards applied by SRC-Reweight and SRC-Evolve. **Bold** and underlined values indicate the best and second-best scores within each post-training backbone, including the 3× reference, respectively.

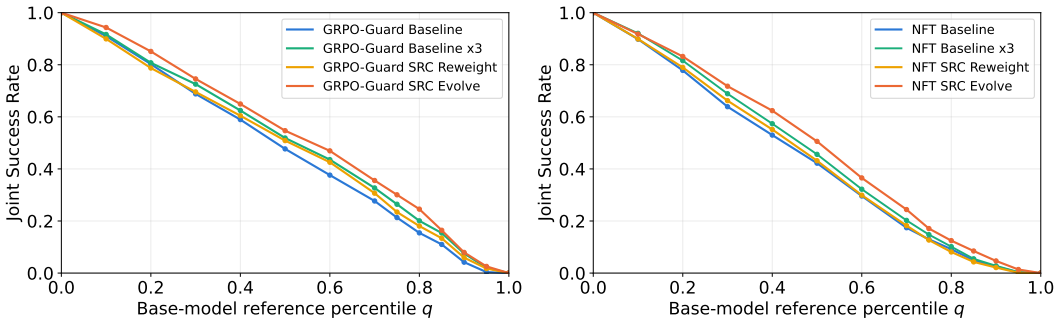
Evaluation Set	GenEval	OCR	PaP + OCR + GenEval			DrawBench			Overall
Rewards	GenEval	OCR	PickScore	CLIP	HPSv2	Aesth.	UniRwd	CycleRwd	
SD-XL	0.2712	0.1351	0.7736	0.2635	0.2261	5.0798	2.6516	0.7636	0.6093
SD3.5-L	0.6446	0.6739	0.8358	0.2990	0.2814	5.4092	3.1989	1.0590	0.9346
FLUX.1-Dev	0.6672	0.6108	0.8888	0.2965	0.2924	5.7055	3.3427	0.9074	0.9312
SD3.5-M w/ CFG	0.5841	0.5416	0.8566	0.2973	0.2606	5.3113	3.0684	1.0371	0.8832
w/ DiffusionNFT	0.8645	0.9223	0.8956	0.3011	0.3218	5.7277	3.3407	1.2308	1.0676
+ GDPO	0.8531	0.8929	0.8983	0.3021	0.3224	5.7794	3.3451	1.2254	1.0634
+ SRC-Reweight	0.8633	0.9257	0.8962	0.3002	0.3230	5.8317	3.3631	1.2431	1.0728
+ SRC-Evolve	0.8673	**0.9371**	**0.9005**	**0.3032**	**0.3297**	**6.0898**	**3.4489**	**1.2591**	**1.0907**
DiffusionNFT (3×)	**0.8714**	0.9328	0.8996	0.3028	0.3266	5.7946	3.3501	1.2564	1.0781
w/ GRPO-Guard	0.7820	0.9063	0.9000	0.3031	0.3241	5.8168	3.3262	1.2550	1.0585
+ SRC-Reweight	0.7809	0.9076	0.9025	0.3049	0.3328	5.8946	3.3450	1.2855	1.0688
+ SRC-Evolve	0.7884	0.9093	**0.9046**	**0.3057**	**0.3394**	**6.0555**	**3.3467**	1.3249	**1.0814**
GRPO-Guard (3×)	**0.7920**	**0.9142**	0.9024	0.3024	0.3334	5.9034	3.3344	**1.3442**	1.0766

Pick-a-Pic (PaP) (Kirstain et al., 2023), GenEval (Ghosh et al., 2023), and the OCR prompt set used in Flow-GRPO (Liu et al., 2025), respectively. We mix PaP, GenEval, and OCR prompts at equal ratio in each epoch of training. The training rewards are PickScore (Kirstain et al., 2023), HPSv2 (Wu et al., 2023), and CLIPScore (CLIP) (Hessel et al., 2021) for all prompts, together with GenEval for compositional prompts and normalized Levenshtein similarity computed from PP-OCRv6 transcriptions and denoted OCR in the tables (Zhang et al., 2026) for text-rendering prompts. The held-out evaluation rewards are AestheticScore (Aesth.) from the LAION improved aesthetic predictor (Schuhmann, 2022), UniReward (UniRwd) (Wang et al., 2025), and CycleReward (CycleRwd) (Bahng et al., 2025); none of the three is used for training or selection. The held-out evaluation set also includes DrawBench prompts (Saharia et al., 2022). The reward scaling details are described in Appendix C.2. More implementation details are provided in Appendix C.

5.2 Mean Reward Results

We apply SRC-Reweight and SRC-Evolve to DiffusionNFT (Zheng et al., 2025) and GRPO-Guard (Wang et al., 2026), and additionally compare with an image-generation adaptation of GDPO (Liu et al., 2026b). All methods retain the baseline's fixed reward weights. We apply SRC only to PaP rollout groups; Appendix C.3 discusses this scope. The standard baselines and SRC-Reweight generate K rollouts per prompt. SRC-Evolve and the 3× references generate $3K$ rollouts per PaP prompt and K per GenEval or OCR prompt; 3× refers to the PaP sampling budget. The 3× references use all $3K$ PaP rollouts as a single training group, whereas SRC-Evolve trains on the K selected rollouts. We measure the five training rewards and three held-out evaluation rewards on fresh generations from the held-out evaluation set, averaging first within each prompt and then equally across applicable prompts. Overall is $[\prod_{k=1}^{8}(\bar{r}_k + 0.01)]^{1/8} - 0.01$, the shifted geometric mean of the eight reported reward means. Against a fixed baseline, it summarizes proportional changes in the shifted reward means rather than absolute score differences.

Table 1 shows gains across both backbones. SRC-Reweight improves six of eight rewards under DiffusionNFT and seven under GRPO-Guard, raising Overall from 1.0676 to 1.0728 and from 1.0585 to 1.0688, respectively. SRC-Evolve improves all eight rewards over each standard baseline and reaches Overall scores of 1.0907 and 1.0814. Against the corresponding 3× references, it improves seven rewards under DiffusionNFT and five under GRPO-Guard, with higher Overall in both cases. Under DiffusionNFT, SRC-Evolve also exceeds GDPO on all eight rewards and Overall.



(a) GRPO-Guard. (b) DiffusionNFT.

Figure 5: **Joint Success Rate curves.** A higher curve at a given base-model reference percentile q indicates a greater probability that one image meets every applicable training-reward threshold.

5.3 Joint Success Rate

The individual reward means in Table 1 do not reveal whether the same image satisfies multiple reward requirements simultaneously. We therefore compare models by the probability of jointly meeting shared quality thresholds.

Let $\mathcal{J}$ be the declared set of rewards applicable to a prompt source, and let $\mathbf{t} = (t_k)_{k\in\mathcal{J}}$ specify the thresholds for each reward. Following the definition of the multivariate survival function in Nelsen (2006), we define Joint Success Rate (JSR) for model π_θ as **the probability that all rewards simultaneously meet their specified quality thresholds**, shared across the compared models:

$$\mathrm{JSR}_\theta(\mathbf{t}) = \Pr_{p,\, x\sim\pi_\theta(\cdot|p)}\left[r_k(x,p) \geq t_k \quad \text{for all } k \in \mathcal{J}\right], \tag{13}$$

where p is drawn uniformly from the applicable held-out prompts. An image succeeds only if it meets all thresholds. We compare models at shared thresholds and estimate JSR from generated images.

Under a conditional multivariate Gaussian approximation, JSR at fixed thresholds depends on reward means and covariance (Appendix A.4). SRC favors rollouts whose active single-reward advantages agree in sign with the scalarized advantage, which may reduce conflicting training signals and help explain higher joint success. The reported JSR is estimated directly and does not require Gaussian rewards.

Since rewards have different scales, we choose their thresholds through a common reference percentile and sweep that percentile to obtain a curve. For each backbone and prompt source, let F_k^{base} be the empirical CDF of raw reward k from fresh images generated by the frozen SD3.5-M base model before post-training, giving each prompt equal weight. We fix these reference CDFs and use them for the standard baseline, its $3\times$ reference, SRC-Reweight, and SRC-Evolve, so that all post-training methods are compared against the same initial model's reward distribution. For $q \in [0,1]$, set $t_k(q) = (F_k^{\mathrm{base}})^{-1}(q)$, where $(F_k^{\mathrm{base}})^{-1}(q) = \inf\{u \in \mathbb{R} : F_k^{\mathrm{base}}(u) \geq q\}$ is the empirical quantile. Substituting this shared threshold vector into Equation 13 gives $J_\theta(q) = \mathrm{JSR}_\theta(\mathbf{t}(q))$, namely:

$$J_\theta(q) = \Pr_{p,\, x\sim\pi_\theta(\cdot|p)}\left[r_k(x,p) \geq (F_k^{\mathrm{base}})^{-1}(q) \quad \text{for all } k \in \mathcal{J}\right], \qquad q \in [0,1], \tag{14}$$

For example, at $q = 0.8$, an image must reach the base model's 80th-percentile threshold in every included reward. Discrete rewards can produce tied thresholds and plateaus in the curve.

We freeze the final checkpoints and generate fresh images for each prompt in the held-out evaluation set. We estimate $J_\theta(q)$ by averaging the success indicator over images within each prompt and then equally over all prompts from PaP, GenEval, and OCR. The three sources contain equal numbers of held-out prompts, so each source contributes one third of the aggregate. For each prompt, the joint event uses only the applicable subset of the five training rewards and the reference CDFs for its source.

Joint success curves. Figure 5 plots $J_\theta(q)$ across the full percentile range for the standard baseline, its $3\times$ reference, SRC-Reweight, and SRC-Evolve. The $3\times$ reference matches SRC-Evolve in generated rollout count, including candidates discarded during selection. For both GRPO-Guard and DiffusionNFT, SRC-Evolve has the highest joint success over most interior percentiles shown, including relative to the corresponding $3\times$ reference. The $3\times$ reference also exceeds the standard baseline over most interior percentiles. SRC-Reweight stays close to the standard baseline, with crossings between their curves depending on the threshold.

Table 2: **SRC-Evolve component ablations.** Overall scores by selection scheme. **Bold** and underlined values mark the best and second-best scores for each backbone across the four schemes.

· *w/o scalar elitism*			· *w/ scalar elitism*		
Selection Scheme	DiffusionNFT	GRPO-Guard	Selection Scheme	DiffusionNFT	GRPO-Guard
Scalar reward R_i	1.0815	1.0746	$\min_{k\in\mathcal{I}} c_{ik}^{\text{raw}}$	1.0836	1.0721
SRC	<u>1.0897</u>	<u>1.0782</u>	SRC	**1.0907**	**1.0814**

5.4 Further analysis

SRC-Evolve component ablations. Table 2 compares four schemes under the same prompt, seed, and resampling settings. The left half retains the top K candidates by scalar reward R_i or SRC without scalar elitism. The right half first retains the scalar elite $\arg\max_{i\in C_t} R_i$, then fills the remaining $K-1$ positions by $\min_{k\in\mathcal{I}} c_{ik}^{\text{raw}}$ or the saturated SRC score. We report Overall on fresh generations from the held-out evaluation set. The full SRC-Evolve selector has the highest Overall on both backbones. SRC selection without scalar elitism ranks second. With scalar elitism, saturated SRC exceeds selection by raw contributions on both backbones.

SRC-Reweight weight concentration. We report normalized effective sample size (ESS) on reweighted PaP groups to check whether SRC-Reweight concentrates training weight on a small number of rollouts. Appendix D.2 defines the diagnostic and reports the full results. Figure 7 shows that ESS/K stays close to 1 across logged training steps for both backbones, with means of 0.9966 for DiffusionNFT and 0.9942 for GRPO-Guard.

Combining SRC-Reweight with SRC-Evolve. Table 5 compares SRC-Evolve with SRC-Evolve + SRC-Reweight. Adding reweighting raises four of the eight reported rewards under DiffusionNFT and two under GRPO-Guard, but slightly lowers Overall.

Evaluations on larger models. We further examine whether the SRC methods transfer from SD3.5-M to the larger, undistilled FLUX.2 [klein] 9B Base model (Klein9B) (Black Forest Labs, 2026). We use Flow-GRPO as its post-training backbone. Table 6 in Appendix D.3 reports the corresponding post-training results. Relative to Flow-GRPO, SRC-Reweight and SRC-Evolve raise Qwen-Image-Bench (Li et al., 2026a) Overall from 50.98 to 51.14 and 51.35, respectively. Both also improve OCR and DrawBench PickScore and aesthetic scores, while GenEval and Creative Generation decrease. These results suggest that SRC can also be useful in this larger-model setting, with mixed changes across measures.

Qualitative results. Appendix D.4 presents two three-column comparisons, with each row using the same prompt across methods. Figure 8 shows SD3.5-M outputs from DiffusionNFT and its SRC-Reweight and SRC-Evolve variants on DrawBench, GenEval, and OCR prompts. These examples provide illustrative comparisons of visual appearance and prompt following. Figure 9 compares Flow-GRPO and its SRC variants on Klein9B using DrawBench, Qwen-Image-Bench, DPGBench (Hu et al., 2024), and OCR prompts. The examples show differences in composition, fine detail, and text rendering across the three methods.

6 Conclusion

In this paper, we studied a mismatch in multi-reward image generation post-training: a scalarized advantage can encourage or discourage a rollout even when one or more active single-reward advantages have the opposite sign. We introduced Scalar Reward Concordance (SRC), which measures the weakest active-reward support for each rollout's scalarized advantage. We then used SRC to redistribute the training weights of existing rollouts in SRC-Reweight and to select a fixed-size population from additional on-policy rollouts in SRC-Evolve. Across DiffusionNFT and GRPO-Guard, the results show improved aesthetics while prompt following is broadly retained. SRC-Evolve also improves the joint success rate on the applicable training rewards.

REFERENCES

Ying Ba, Tianyu Zhang, Mohan Zhou, Yalong Bai, Wenyi Mo, Guiwei Zhang, Bing Su, and Ji-Rong Wen. Pareto-guided optimal transport for multi-reward alignment. *arXiv preprint arXiv:2605.13155*, 2026.

Hyojin Bahng, Caroline Chan, Fredo Durand, and Phillip Isola. Cycle consistency as reward: Learning image-text alignment without human preferences. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22934–22946, October 2025. URL https://arxiv.org/abs/2506.02095.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations*, 2024.

Black Forest Labs. Announcing Black Forest Labs: FLUX.1, August 2024. URL https://bfl.ai/blog/24-08-01-bfl.

Black Forest Labs. FLUX.2 [klein]: Towards interactive visual intelligence, January 2026. URL https://bfl.ai/blog/flux2-klein-towards-interactive-visual-intelligence. Released January 15, 2026.

Chieh-Yun Chen, Zhonghao Wang, Qi Chen, Zhifan Ye, Min Shi, Yue Zhao, Yinan Zhao, Hui Qu, Wei-An Lin, Yiru Shen, et al. Mapreduce lora: Advancing the pareto front in multi-preference optimization for generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 34375–34384, 2026a.

Dongliang Chen, Xinlin Zhuang, Junjie Xu, Luojian Xie, Zehui Wang, Jiaxi Zhuang, Haolin Yang, Liang Dou, Xiao He, Xingjiao Wu, and Ying Qian. Apex: Learning adaptive priorities for multi-objective alignment in vision-language generation. *arXiv preprint arXiv:2601.06574*, 2026b.

Kevin Clark, Paul Vicol, Kevin Swersky, and David J. Fleet. Directly fine-tuning diffusion models on differentiable rewards. In *International Conference on Learning Representations*, 2024.

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. *Advances in Neural Information Processing Systems*, 36, 2023.

Zhen Fang, Wenxuan Huang, Yu Zeng, Yiming Zhao, Shuang Chen, Kaituo Feng, Yunlong Lin, Lin Chen, Zehui Chen, Shaosheng Cao, et al. Flow-opd: On-policy distillation for flow matching models. *arXiv preprint arXiv:2605.08063*, 2026.

Song Fei, Tian Ye, Sixiang Chen, Zhaohu Xing, Jianyu Lai, and Lei Zhu. Lucidnft: Lr-anchored multi-reward preference optimization for generative real-world super-resolution. *arXiv preprint arXiv:2603.05947*, 2026.

Alan Genz and Frank Bretz. *Computation of Multivariate Normal and t Probabilities*, volume 195 of *Lecture Notes in Statistics*. Springer, 2009. doi: 10.1007/978-3-642-01689-9.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36, 2023.

Shelly Golan, Michael Finkelson, Ariel Bereslavsky, Yotam Nitzan, and Or Patashnik. Paretoslider: Diffusion models post-training for continuous reward control. *arXiv preprint arXiv:2604.20816*, 2026.

Yuncheng Guo, Junyan Ye, Chenjue Zhang, Hengrui Kang, Haohuan Fu, Conghui He, and Weijia Li. OmniAID: Decoupling semantics and artifacts for universal ai-generated image detection in the wild. *arXiv preprint arXiv:2511.08423*, 2025. URL https://arxiv.org/abs/2511.08423.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. CLIPScore: A reference-free evaluation metric for image captioning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 7514–7528. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.emnlp-main.595. URL https://aclanthology.org/2021.emnlp-main.595/.

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. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. ELLA: Equip diffusion models with LLM for enhanced semantic alignment. *arXiv preprint arXiv:2403.05135*, 2024. URL https://arxiv.org/abs/2403.05135.

Yuval Kirstain, Adam Polyak, Uriel Singer, Shahbuland Matiana, Joe Penna, and Omer Levy. Pick-a-pic: An open dataset of user preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36:36652–36663, 2023.

Jaegun Lee and Janghoon Choi. Flow-multi: A flow-matching multi-reward framework for text-to-image generation. *Sensors*, 26(4):1120, 2026.

Kyungmin Lee, Xiahong Li, Qifei Wang, Junfeng He, Junjie Ke, Ming-Hsuan Yang, Irfan Essa, Jinwoo Shin, Feng Yang, and Yinxiao Li. Calibrated multi-preference optimization for aligning diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18465–18475, 2025.

Seung Hyun Lee, Yinxiao Li, Junjie Ke, Innfarn Yoo, Han Zhang, Jiahui Yu, Qifei Wang, Fei Deng, Glenn Entis, Junfeng He, et al. Parrot: Pareto-optimal multi-reward reinforcement learning framework for text-to-image generation. In *European Conference on Computer Vision*, pp. 462–478. Springer, 2024.

Niantong Li, Guangzheng Hu, Weixu Qiao, et al. Qwen-Image-Bench: From generation to creation in text-to-image evaluation. *arXiv preprint arXiv:2605.28091*, 2026a. URL https://arxiv.org/abs/2605.28091.

Quanhao Li, Junqiu Yu, Kaixun Jiang, Yujie Wei, Zhen Xing, Pandeng Li, Ruihang Chu, Shiwei Zhang, Yu Liu, and Zuxuan Wu. Diffusionopd: A unified perspective of on-policy distillation in diffusion models. *arXiv preprint arXiv:2605.15055*, 2026b.

Kaiqing Lin, Zhiyuan Yan, Ruoxin Chen, Junyan Ye, Ke-Yue Zhang, Yue Zhou, Peng Jin, Bin Li, Taiping Yao, and Shouhong Ding. Seeing before reasoning: A unified framework for generalizable and explainable fake image detection. *arXiv preprint arXiv:2509.25502*, 2025. URL https://arxiv.org/abs/2509.25502.

Haoran Liu, Zheni Zeng, Yukun Yan, Yuxuan Chen, and Yunduo Xiao. Drugr: Optimizing molecular drugs through LLM-based explicit reasoning. *arXiv preprint arXiv:2602.08213*, 2026a. URL https://arxiv.org/abs/2602.08213.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-GRPO: Training flow matching models via online RL. *Advances in neural information processing systems*, 38:40783–40818, 2025.

Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Peter Belcak, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, Yejin Choi, Jan Kautz, and Pavlo Molchanov. GDPO: Group reward-decoupled normalization policy optimization for multi-reward RL optimization. *arXiv preprint arXiv:2601.05242*, 2026b. URL https://arxiv.org/abs/2601.05242.

Qiang Lyu, Zicong Chen, Chongxiao Wang, Haolin Shi, Shibo Gao, Ran Piao, Youwei Zeng, Jianlou Si, Fei Ding, Jing Li, et al. Multi-grpo: Multi-group advantage estimation for text-to-image generation with tree-based trajectories and multiple rewards. *arXiv preprint arXiv:2512.00743*, 2025.

Roger B. Nelsen. *An Introduction to Copulas*. Springer Series in Statistics. Springer, New York, 2 edition, 2006. doi: 10.1007/0-387-28678-0.

Bowen Ping, Chengyou Jia, Minnan Luo, Hangwei Qian, and Ivor Tsang. Flow-Factory: A unified framework for reinforcement learning in Flow-Matching models. *arXiv preprint arXiv:2602.12529*, 2026. URL https://arxiv.org/abs/2602.12529.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2307.01952.

Alexandre Rame, Guillaume Couairon, Corentin Dancette, Jean-Baptiste Gaya, Mustafa Shukor, Laure Soulier, and Matthieu Cord. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. *Advances in Neural Information Processing Systems*, 36:71095–71134, 2023.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://arxiv.org/abs/2205.11487.

Christoph Schuhmann. CLIP+MLP aesthetic score predictor. GitHub repository, 2022. URL https://github.com/christophschuhmann/improved-aesthetic-predictor.

Zhihong Shao et al. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300.

Stability AI. Introducing Stable Diffusion 3.5, October 2024. URL https://stability.ai/news-updates/introducing-stable-diffusion-3-5.

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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8228–8238, 2024.

Jing Wang, Jiajun Liang, Jie Liu, Henglin Liu, Gongye Liu, Jun Zheng, Wanyuan Pang, Ao Ma, Zhenyu Xie, Xintao Wang, Meng Wang, Pengfei Wan, and Xiaodan Liang. GRPO-Guard: Mitigating implicit over-optimization in flow matching via regulated clipping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5988–5998, 2026.

Yibin Wang, Yuhang Zang, Hao Li, Cheng Jin, and Jiaqi Wang. Unified reward model for multimodal understanding and generation. *arXiv preprint arXiv:2503.05236*, 2025. URL https://arxiv.org/abs/2503.05236.

Jungmyung Wi, Hyunsoo Kim, and Donghyun Kim. Correlation-weighted multi-reward optimization for compositional generation. *arXiv preprint arXiv:2603.18528*, 2026. URL https://arxiv.org/abs/2603.18528.

Xiaoshi Wu, Yiming Hao, Keqiang Sun, Yixiong Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. *arXiv preprint arXiv:2306.09341*, 2023. URL https://arxiv.org/abs/2306.09341.

Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imagereward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36, 2023.

Yixuan Even Xu, Yash Savani, Fei Fang, and Zico Kolter. Not all rollouts are useful: Down-sampling rollouts in LLM reinforcement learning. *arXiv preprint arXiv:2504.13818*, 2025. URL https://arxiv.org/abs/2504.13818.

Sixian Zhang, Bohan Wang, Junqiang Wu, Yan Li, Tingting Gao, Di Zhang, and Zhongyuan Wang. Learning multi-dimensional human preference for text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8018–8027, 2024.

Yubo Zhang, Xueqing Wang, Manhui Lin, Yue Zhang, Penglongyi Deng, Ting Sun, Tingquan Gao, Zelun Zhang, Jiaxuan Liu, Changda Zhou, Hongen Liu, Suyin Liang, Cheng Cui, Yi Liu, Dianhai Yu, and Yanjun Ma. PP-OCRv6: From 1.5m to 34.5m parameters, surpassing billion-scale VLMs on OCR tasks. *arXiv preprint arXiv:2606.13108*, 2026. URL https://arxiv.org/abs/2606.13108.

Canyu Zhao, Hao Chen, Yunze Tong, Yu Qiao, Jiacheng Li, and Chunhua Shen. Marble: Multi-aspect reward balance for diffusion rl. *arXiv preprint arXiv:2605.06507*, 2026.

Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. Diffusionnft: Online diffusion reinforcement with forward process. *arXiv preprint arXiv:2509.16117*, 2025.

AI USE STATEMENT

Generative AI tools were used to implement code, search for relevant literature, create figures and tables, and edit the manuscript for clarity and readability. The authors independently designed the experiments, interpreted the results, and developed the theoretical analysis and derivations. The authors reviewed the AI-assisted materials and take responsibility for the final code, citations, visuals, text, and scientific claims in this paper.

A ADDITIONAL THEORETICAL DETAILS

This section supplements the SRC construction in Section 3 with its reward-correlation identity, saturation rationale, and local extension to DiffusionNFT. It also derives the analytical expression for JSR under a conditional Gaussian assumption in Section 5.3.

A.1 RELATION TO GROUP-LEVEL REWARD CORRELATIONS

We expand the reward-correlation connection stated after Equation 8 in Section 3, showing how raw contributions aggregate into a group-level reward-space quantity.

Restricting vectors and matrices in this subsection to the active coordinates, let $\mathbf{z}_i = (z_{ik} : k \in \mathcal{I})$, $D = \mathrm{diag}(\sigma_k : k \in \mathcal{I})$, $W = \mathrm{diag}(w_k : k \in \mathcal{I})$, and $\widetilde{\mathbf{w}} = D\mathbf{w}/\sigma_R$. The identity $\sigma_R A_i = \sum_k w_k \sigma_k z_{ik}$ implies $A_i = \mathbf{z}_i^\top \widetilde{\mathbf{w}}$. Hence the raw contribution profile in Equation 8 is

$$\mathbf{c}_i^{\mathrm{raw}} = A_i W \mathbf{z}_i = W \mathbf{z}_i \mathbf{z}_i^\top \widetilde{\mathbf{w}}. \tag{15}$$

Because every coordinate of $\mathbf{z}_i$ has zero group mean and unit group variance, the matrix

$$C_G = \frac{1}{K} \sum_{i=1}^{K} \mathbf{z}_i \mathbf{z}_i^\top \tag{16}$$

is the empirical reward correlation matrix of the group. Averaging the raw profiles over rollouts gives

$$\frac{1}{K} \sum_{i=1}^{K} \mathbf{c}_i^{\mathrm{raw}} = W \left(\frac{1}{K} \sum_{i=1}^{K} \mathbf{z}_i \mathbf{z}_i^\top \right) \widetilde{\mathbf{w}} \tag{17}$$

$$= W C_G \widetilde{\mathbf{w}}. \tag{18}$$

Thus, the group average of the kth raw contribution is the declared weight w_k times reward k's group correlation with the scalarized advantage. Equation 15 identifies each rollout's additive contribution to this reward-space quantity.

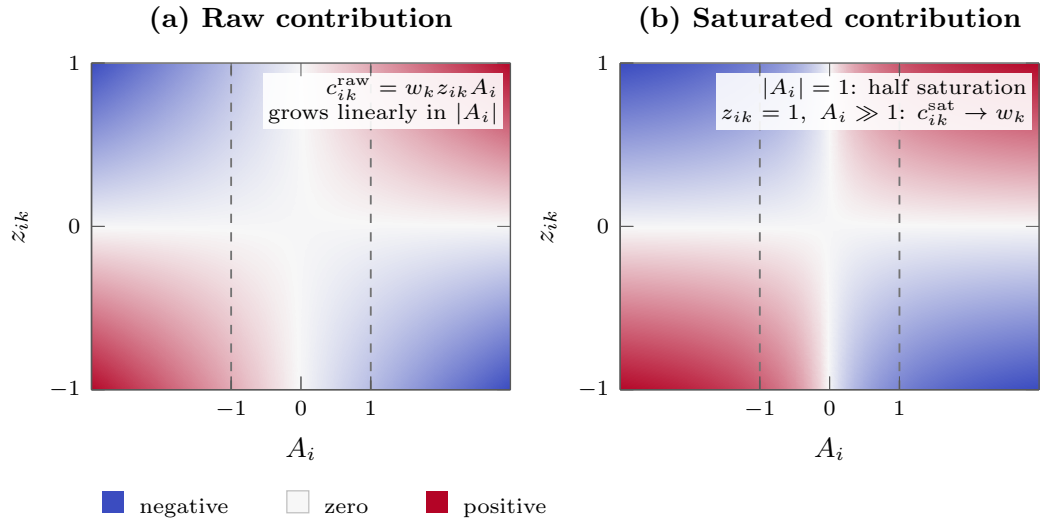


Figure 6: **Raw versus saturated coordinate contributions.** Color indicates contribution sign and relative magnitude. Dashed lines mark $|A_i| = 1$, where the saturation gate is approximately one half.

A.2 Saturation Rationale

This subsection explains the saturation introduced in Equation 9 of Section 3. The factorization below clarifies how it limits large scalarized advantages while preserving contribution signs and sensitivity near zero.

Define the weakest signed reward margin

$$m_i = \min_{k \in \mathcal{I}} w_k z_{ik} \operatorname{sign}(A_i), \qquad \operatorname{sign}(0) = 0. \tag{19}$$

The weakest raw coordinate factorizes as

$$\min_{k \in \mathcal{I}} c_{ik}^{\text{raw}} = |A_i| m_i. \tag{20}$$

Because A_i is standardized, $|A_i| = 1$ means that the scalar reward lies one group standard deviation from its mean. The factor $|A_i|$ provides local discrimination around the scalar group mean, but can amplify the contributions of large-$|A_i|$ rollouts. For a fixed group size K, the normalization in Equation 5 implies $|A_i| \le \sqrt{K-1}$. Using m_i alone would prevent this amplification but turn an arbitrarily small nonzero scalarized advantage into a full-strength sign decision. We therefore saturate the scalarized-advantage factor smoothly, using one standard deviation as the transition scale.

The saturated profile used by SRC is

$$\mathbf{c}_i^{\text{sat}} := \mathbf{c}_i^{\text{raw}} \frac{1}{|A_i| + 1 + \epsilon} = W \mathbf{z}_i \operatorname{sign}(A_i) \frac{|A_i|}{|A_i| + 1 + \epsilon}. \tag{21}$$

Consequently,

$$\text{SRC}(i; G) = m_i \frac{|A_i|}{|A_i| + 1 + \epsilon}. \tag{22}$$

The score approaches zero when $|A_i| \ll 1$ and approaches m_i when $|A_i| \gg 1$. Saturation preserves every coordinate sign but compresses magnitude; unlike $\mathbf{c}_i^{\text{raw}}$, the saturated profile is not an additive decomposition of the group-level reward-correlation quantity. Figure 6 illustrates how saturation compresses contribution magnitudes while preserving their signs. The schematic varies only A_i and z_{ik} for $w_k > 0$ and negligible ϵ.

A.3 Extension to DiffusionNFT

We extend the within-rollout sign analysis in Section 3, specifically Equation 7, to DiffusionNFT. The derivation identifies the local conditions under which raw contributions describe agreement with

its reward-driven update component. We first analyze the unweighted squared-error policy objective; below we relate this calculation to the prediction parameterization and adaptive weighting used in the DiffusionNFT implementation (Zheng et al., 2025).

The same reward-wise sign relation also arises in the reward-driven component of DiffusionNFT, but not by treating DiffusionNFT as a policy-gradient method. DiffusionNFT converts the centered scalar reward into the signed optimality signal

$$2p_i - 1 = \mathrm{clip}\left(\frac{\sigma_R A_i}{Z_G}, -1, 1\right), \qquad Z_G > 0, \tag{23}$$

which mixes positive and negative forward-process fitting losses. The sampled training tuple, observed rewards, group statistics, Z_G, and hence p_i are held fixed when differentiating with respect to θ. Let $\beta > 0$ denote its guidance strength, v_i^{old} the frozen sampling-policy prediction, and v_i^{tar} the forward-process target. For the trainable velocity predictor $v_{\theta,i}$, define its Jacobian and the ascent-oriented base direction as

$$J_i := \frac{\partial v_{\theta,i}}{\partial \theta}, \qquad h_i^{\mathrm{NFT}} := -J_i^\top (v_i^{\mathrm{old}} - v_i^{\mathrm{tar}}). \tag{24}$$

Proposition 1 (Local DiffusionNFT reward-component identity)**.** *At a fresh policy reference $v_{\theta,i} = v_i^{\mathrm{old}}$ and before optimality clipping activates, i.e., $|\sigma_R A_i| < Z_G$, retain the positive standard-deviation assumptions of Section 3 and define, for $k \in \mathcal{I}$,*

$$g_i^{\mathrm{NFT,rew}} = 2\beta \frac{\sigma_R A_i}{Z_G} h_i^{\mathrm{NFT}}, \qquad g_{ik}^{\mathrm{NFT}} = 2\beta \frac{w_k \sigma_k z_{ik}}{Z_G} h_i^{\mathrm{NFT}}. \tag{25}$$

Then

$$\left\langle g_{ik}^{\mathrm{NFT}}, g_i^{\mathrm{NFT,rew}} \right\rangle = \frac{4\beta^2}{Z_G^2} \sigma_k \sigma_R c_{ik}^{\mathrm{raw}} \|h_i^{\mathrm{NFT}}\|_2^2. \tag{26}$$

If $h_i^{\mathrm{NFT}} \neq \mathbf{0}$, this inner product has the same sign as c_{ik}^{raw} and c_{ik}^{sat}.

Proof. For one forward-process training tuple, DiffusionNFT forms the implicit positive and negative predictors

$$v_{\theta,i}^+ = (1-\beta) v_i^{\mathrm{old}} + \beta v_{\theta,i}, \qquad v_{\theta,i}^- = (1+\beta) v_i^{\mathrm{old}} - \beta v_{\theta,i}. \tag{27}$$

For the unweighted squared-error policy objective, the per-rollout loss is

$$\ell_i^{\mathrm{NFT}} = p_i \|v_{\theta,i}^+ - v_i^{\mathrm{tar}}\|_2^2 + (1-p_i)\|v_{\theta,i}^- - v_i^{\mathrm{tar}}\|_2^2. \tag{28}$$

Treat p_i, v_i^{old}, and v_i^{tar} as stop-gradient quantities and write

$$e_i^{\mathrm{old}} = v_i^{\mathrm{old}} - v_i^{\mathrm{tar}}, \qquad \delta v_i = v_{\theta,i} - v_i^{\mathrm{old}}. \tag{29}$$

The two residuals are

$$v_{\theta,i}^+ - v_i^{\mathrm{tar}} = e_i^{\mathrm{old}} + \beta \delta v_i, \qquad v_{\theta,i}^- - v_i^{\mathrm{tar}} = e_i^{\mathrm{old}} - \beta \delta v_i. \tag{30}$$

Differentiating equation 28 therefore gives

$$\nabla_\theta \ell_i^{\mathrm{NFT}} = 2\beta^2 J_i^\top \delta v_i + 2\beta(2p_i - 1) J_i^\top e_i^{\mathrm{old}}. \tag{31}$$

The first term is reward-independent, whereas the second is the reward-driven contrast term. At a fresh reference, $\delta v_i = 0$. Before clipping, $2p_i - 1 = \sigma_R A_i / Z_G$, so the reward-driven update direction under loss minimization is

$$-\nabla_\theta \ell_i^{\mathrm{NFT}} = 2\beta \frac{\sigma_R A_i}{Z_G} h_i^{\mathrm{NFT}} = g_i^{\mathrm{NFT,rew}}. \tag{32}$$

Because $\sigma_R A_i = \sum_k w_k \sigma_k z_{ik}$, this direction decomposes as

$$g_i^{\mathrm{NFT,rew}} = \sum_{k \in \mathcal{I}} 2\beta \frac{w_k \sigma_k z_{ik}}{Z_G} h_i^{\mathrm{NFT}} = \sum_{k \in \mathcal{I}} g_{ik}^{\mathrm{NFT}}. \tag{33}$$

Taking the within-rollout inner product yields

$$\left\langle g_{ik}^{\mathrm{NFT}}, g_i^{\mathrm{NFT,rew}} \right\rangle = \frac{4\beta^2}{Z_G^2} w_k \sigma_k z_{ik} \sigma_R A_i \|h_i^{\mathrm{NFT}}\|_2^2 \tag{34}$$

$$= \frac{4\beta^2}{Z_G^2} \sigma_k \sigma_R c_{ik}^{\mathrm{raw}} \|h_i^{\mathrm{NFT}}\|_2^2, \tag{35}$$

which proves the proposition. □

Relation to the implemented policy loss. The SD3 implementation computes squared errors in the predicted clean latent, with separate stop-gradient adaptive normalizers for the two branches and additional positive loss scaling. For a fixed forward-process tuple, converting a velocity residual to a clean-latent residual multiplies it by the negative noise-time coefficient. Its squared error therefore differs by a common nonnegative scalar factor. At $v_{\theta,i} = v_i^{\mathrm{old}}$, the two branch predictions coincide, so their adaptive normalizers also coincide. Consequently, at this reference the implemented policy gradient is a common positive scalar multiple of the gradient above for a nonzero noise-time coefficient. The same factor multiplies every reward component and its square multiplies the inner product, preserving the sign identity. The exact coefficient in Equation 26 is for the unweighted velocity objective as written; it is not the numerical coefficient of the implemented loss. A zero noise-time coefficient or a zero base direction yields a zero update and no strict sign conclusion.

Scope of the conclusion. The quantities g_{ik}^{NFT} are additive reward components using the same scalar-reward normalizer Z_G and base direction. They are not the complete gradients of separate single-reward training runs, which may use different normalizers and clipping states. Away from the fresh reference, the unweighted objective also has the reward-independent term in Equation 31, and the implemented branch normalizers need not coincide. The identity therefore does not describe the complete update throughout training or any additional reference-model regularization. It establishes a within-rollout sign interpretation under the stated conditions, not an ordering of actual reward improvements by SRC or a guarantee that all rewards improve.

A.4 JSR AND THE JOINT REWARD DISTRIBUTION

JSR in Section 5.3 is a joint exceedance probability defined for arbitrary reward distributions (Nelsen, 2006). Here we derive its analytical expression under a conditional multivariate Gaussian assumption. The reported JSR curves are estimated directly from generated images and do not require this assumption.

Conditional Gaussian model. Fix a prompt p and restrict the reward vector to the declared set $\mathcal{J}$, with $d = |\mathcal{J}|$. Suppose that

$$\mathbf{r}(x,p) \sim \mathcal{N}(\boldsymbol{\mu}_{\theta,p}, \Sigma_{\theta,p}), \qquad x \sim \pi_\theta(\cdot \mid p), \tag{36}$$

where $\boldsymbol{\mu}_{\theta,p}$ and $\Sigma_{\theta,p}$ are the conditional mean vector and covariance matrix. Write

$$D_{\theta,p} = \mathrm{diag}(\sigma_{\theta,p,k} : k \in \mathcal{J}), \qquad \Sigma_{\theta,p} = D_{\theta,p} C_{\theta,p} D_{\theta,p}, \tag{37}$$

where $\sigma_{\theta,p,k}$ is the marginal standard deviation and $C_{\theta,p}$ is the conditional reward correlation matrix. Assume that $\Sigma_{\theta,p}$ is positive definite. The standardized vector $\mathbf{u} = D_{\theta,p}^{-1}(\mathbf{r} - \boldsymbol{\mu}_{\theta,p})$ then follows $\mathcal{N}(\mathbf{0}, C_{\theta,p})$.

Exact Gaussian expression. Let $\Phi_d(\mathbf{a}; C) = \Pr_{\mathbf{u}\sim\mathcal{N}(\mathbf{0},C)}(\mathbf{u} \le \mathbf{a})$ denote the standard multivariate normal CDF, with inequalities interpreted coordinate-wise. For positive definite C,

$$\Phi_d(\mathbf{a}; C) = \frac{1}{(2\pi)^{d/2}|C|^{1/2}} \int_{-\infty}^{a_1} \cdots \int_{-\infty}^{a_d} \exp\left(-\frac{1}{2}\mathbf{u}^\top C^{-1}\mathbf{u}\right) du_d \cdots du_1. \tag{38}$$

For a shared reward-threshold vector $\mathbf{t}$, standardization and the symmetry $-\mathbf{u} \overset{d}{=} \mathbf{u}$ give

$$\begin{aligned} &\Pr_{x\sim\pi_\theta(\cdot|p)}[\mathbf{r}(x,p) \ge \mathbf{t}] \\ &= \Pr\left[\mathbf{u} \ge D_{\theta,p}^{-1}(\mathbf{t} - \boldsymbol{\mu}_{\theta,p})\right] \\ &= \Pr\left[-\mathbf{u} \le D_{\theta,p}^{-1}(\boldsymbol{\mu}_{\theta,p} - \mathbf{t})\right] \\ &= \Phi_d\left(D_{\theta,p}^{-1}(\boldsymbol{\mu}_{\theta,p} - \mathbf{t}); C_{\theta,p}\right). \end{aligned} \tag{39}$$

Consequently, averaging over the same uniformly sampled held-out prompts as in Equation 13 yields

$$\mathrm{JSR}_\theta(\mathbf{t}) = \mathbb{E}_p\left[\Phi_d\left(D_{\theta,p}^{-1}(\boldsymbol{\mu}_{\theta,p} - \mathbf{t}); C_{\theta,p}\right)\right]. \tag{40}$$

This is an exact analytical expression in terms of the multivariate normal CDF under the stated Gaussian assumption. For a general correlation matrix, it does not reduce to a product of univariate CDFs or generally have an elementary closed form; numerical evaluation of multivariate normal probabilities is treated by Genz & Bretz (2009).

B Additional Method Details

This section expands the methods in Section 4, providing the SRC-Reweight derivation, the full SRC-Evolve procedure, and the details needed to integrate both methods with the training backbones.

B.1 Derivation of SRC-Reweight

We derive the softmax weighting rule introduced in Section 4.1, Equation 11, to explain its regularization interpretation and the distinct roles of temperature and interpolation.

Let $u_i = 1/K$ be the original uniform mass. Before adding the explicit uniform floor in Equation 11, we obtain the softmax component by solving

$$\mathbf{q}^* = \arg\max_{\mathbf{q}\in\Delta_K} \left\{ \mathbf{q}^\top \widehat{\mathbf{s}} - \tau\, \mathrm{KL}(\mathbf{q}\|\mathbf{u}) \right\}. \tag{41}$$

The first term maximizes expected SRC, while the KL penalty discourages a large departure from the original rollout distribution. To derive the solution, introduce a multiplier ν for $\sum_i q_i = 1$. The stationarity condition is

$$\widehat{s}_i - \tau \left(\log \frac{q_i}{u_i} + 1 \right) + \nu = 0, \tag{42}$$

which implies $q_i^* \propto u_i \exp(\widehat{s}_i/\tau)$. Since $u_i = 1/K$, normalization gives the softmax component in Equation 11. Temperature τ controls relative discrimination among SRC scores, whereas λ controls the total interpolation away from uniform training. Finally, $\boldsymbol{\omega} = (1-\lambda)\mathbf{u} + \lambda \mathbf{q}^*$ gives Equation 11.

B.2 Applying SRC-Reweight to GRPO-Guard and DiffusionNFT

We expand the backbone integration described in Section 4.1 by specifying where the rollout weights ω_i enter each training objective. Both integrations retain the original group statistics when constructing training signals; the weights change how rollouts contribute to the loss.

GRPO-Guard. We replace the scalarized advantage by $A_i^{\mathrm{SRC}} = K\omega_i A_i$ in the reward-dependent policy term. The factor K compensates for the baseline's mean reduction, so that

$$-\frac{1}{K}\sum_{i=1}^{K} \rho_i(\theta) A_i^{\mathrm{SRC}} = -\sum_{i=1}^{K} \omega_i \rho_i(\theta) A_i. \tag{43}$$

The displayed equality uses the unclipped loss in Equation 1; with clipping, the same nonnegative factor $K\omega_i$ multiplies each rollout's clipped policy term. It does not enter the original advantage normalization. An independent reference-model KL penalty remains uniformly aggregated.

DiffusionNFT. We first construct the positive and negative branches using the original uniformly computed group mean reward and NFT normalizer. We then aggregate the complete per-rollout policy loss as $\sum_{i=1}^{K} \omega_i \ell_i^{\mathrm{NFT}}$, retaining the implementation's prediction parameterization, adaptive branch normalizers, and loss scaling. Equation 28 gives its unweighted squared-error form for the local analysis. This weights both the reward-independent and reward-driven components of the per-rollout loss; it does not replace the optimality probability p_i or redefine either branch.

B.3 SRC-Evolve

This subsection details the population evolution and environmental selection in Section 4.2. It specifies the resampling and selection procedure, explains the selection rule's lower-bound interpretation, and states the guarantee provided by scalar elitism.

Algorithm 1 SRC-Evolve

Require: Prompt p; initial population P_0 of K on-policy rollouts; number of generations T; resampling budget M

for $t = 0, \dots, T-1$ **do**
  Sample M complete on-policy rollouts S_t from the current policy conditioned on p
  Evaluate the rewards of every rollout in S_t
  $C_t \leftarrow P_t \cup S_t$
  Compute $f_i = \mathrm{SRC}(i; C_t)$ for every $i \in C_t$ using the reward means and standard deviations of C_t, held fixed throughout selection
  $e^* \leftarrow \arg\max_{i \in C_t} R_i$
  $P_{t+1} \leftarrow \{e^*\} \cup \mathrm{Top}_{K-1}(C_t \setminus \{e^*\}; f_i)$
end for
return P_T

SRC-Evolve is run independently for each prompt. An individual is a complete on-policy rollout with reward vector $\mathbf{r}_i$ and scalar reward $R_i = \mathbf{w}^\top \mathbf{r}_i$. The initial population P_0 consists of K on-policy rollouts sampled from the current model. Environmental selection restores the same cardinality after every generation, so $|P_t| = K$ for $t = 0, \dots, T$.

At each generation, we sample M additional on-policy rollouts, where M is a positive integer. We define the resampling ratio as M/K, using the same population size K as in the main method. The budget M directly controls the additional denoising and reward-evaluation cost per generation. Each rollout in S_t is sampled end-to-end from the current policy under the same prompt and inference configuration as the initial population. After reward evaluation, the candidate pool is $C_t = P_t \cup S_t$. Its reward means and fitness values are computed once and frozen for the ensuing environmental selection. Each candidate receives the saturated score in Equation 10, with $G = C_t$.

The selection rule admits a lower-bound interpretation. For a size-K subset $S \subseteq C_t$, let $\mathbf{c}_i^{\mathrm{sat}}(C_t)$ denote the saturated profile in Equation 21, computed using the reward means and standard deviations of C_t, held fixed for every subset S. Since $f_i = \mathrm{SRC}(i; C_t)$, we have

$$c_{ik}^{\mathrm{sat}}(C_t) \geq \min_{l \in \mathcal{I}} c_{il}^{\mathrm{sat}}(C_t) = f_i \tag{44}$$

for every active reward k. Therefore,

$$\min_{k \in \mathcal{I}} \frac{1}{K} \sum_{i \in S} c_{ik}^{\mathrm{sat}}(C_t) \geq \frac{1}{K} \sum_{i \in S} f_i. \tag{45}$$

Thus, conditional on retaining e^*, Equation 12 exactly maximizes a separable lower bound on the weakest aggregate saturated coordinate computed with these fixed candidate-pool statistics. Keeping the candidate-pool statistics fixed also means that the selected group can have different means and even different per-rollout contribution signs after recentering; these quantities must be recomputed for diagnostics. Since $P_t \subseteq C_t$ and $e^* \in P_{t+1}$, scalar elitism guarantees only that the best observed scalar reward is retained: $\max_{i \in P_{t+1}} R_i = \max_{i \in C_t} R_i \geq \max_{i \in P_t} R_i$. It does not guarantee improvement in population-mean reward, every reward coordinate, the Pareto frontier, or fresh-rollout quality. In implementation, non-elite candidates are sorted lexicographically by f_i, scalar reward R_i, and a stable identifier. The scalar elite is inserted first, the highest-ranked $K-1$ remaining candidates fill the population, and this complete survivor set becomes P_{t+1}.

If $t < T-1$, the selected population $P_{t+1} \subseteq P_t \cup S_t$ becomes the current population of the next generation. We use a fixed generation count T as the termination condition and return P_T as the training group for the downstream policy update.

C Additional Experimental and Implementation Details

This section supplements the setup in Section 5.1 with numerical settings needed to reproduce the comparisons.

Table 3: Experimental configuration grouped by component. Values spanning both backbone columns are shared.

Setting	DiffusionNFT	GRPO-Guard
Model		
LoRA rank	32	
LoRA scaling α	64	
Training		
Training epochs	800	1600
Training steps per epoch	1	2
Unique prompts per epoch	48	96
Prompt mixing ratio (PaP : GenEval : OCR)	1 : 1 : 1	
Sampling		
Image resolution	512×512	
Rollout-group size K	16	24
Training denoising steps	14	10
Evaluation denoising steps	40	40
CFG scale	1.0	4.5
Evaluation images per prompt (main results)	16 (for DrawBench), 4 (for other 3 sets)	
SRC-Reweight		
Interpolation λ	0.5	
Temperature τ	1	
SRC-Evolve		
Resampling ratio M/K	1	
Number of generations T	2	

Table 4: Fixed scalar reward weights for the SD3.5-M experiments.

Prompt source	PickScore	CLIP	HPSv2	GenEval	OCR
PaP	0.5	1	0.5	–	–
GenEval	0.2	0.2	0.2	1	–
OCR	0.2	0.2	0.2	–	1

C.1 Experimental configuration

We specify the numerical settings for the model, objectives, datasets, and rewards introduced in Section 5.1. Table 3 summarizes the shared and backbone-specific numerical settings; other objective-specific settings follow the original GRPO-Guard and DiffusionNFT implementations. Table 4 lists the fixed scalar reward weights for each prompt source in the SD3.5-M experiments, shared by both backbones. Dashes indicate rewards not used for that prompt source.

C.2 Reward scales and rescaling

We use the following output conventions for the five training rewards. PickScore is the raw model score divided by 26, without clipping to $[0, 1]$. CLIPScore is the cosine similarity between the image and text embeddings, with no additional rescaling. HPSv2 is the image–text feature inner product returned by the reward model, also without additional rescaling. GenEval uses the evaluator's score in $[0, 1]$ directly. OCR is the normalized Levenshtein similarity computed from PP-OCRv6 transcriptions: one minus the edit-distance ratio capped at one, giving a score in $[0, 1]$.

The three held-out evaluation rewards (AestheticScore, UniReward, and CycleReward) are reported as their original reward-model outputs, without additional rescaling. Table 1 reports reward means on these scales, and Overall is computed from the eight reported means using the shifted geometric mean defined in Section 5.2. These output conventions are distinct from the within-group standardization used to compute single-reward advantages and the scalarized advantage.

Table 5: Combining SRC-Reweight with SRC-Evolve.

Evaluation Set	GenEval	OCR	PaP + OCR + GenEval			DrawBench			Overall
Rewards	GenEval	OCR	PickScore	CLIP	HPSv2	Aesth.	UniRwd	CycleRwd	
DiffusionNFT (3×)	0.8714	0.9328	0.8996	0.3028	0.3266	5.7946	3.3501	1.2564	1.0781
SRC-Evolve	0.8673	0.9371	0.9005	0.3032	0.3297	6.0898	3.4489	1.2591	1.0907
SRC-Evolve & SRC-Reweight	0.8627	0.9338	0.9013	0.3021	0.3316	6.0912	3.4512	1.2486	1.0889
GRPO-Guard (3×)	0.7920	0.9142	0.9024	0.3024	0.3334	5.9034	3.3344	1.3442	1.0766
SRC-Evolve	0.7884	0.9093	0.9046	0.3057	0.3394	6.0555	3.3467	1.3249	1.0814
SRC-Evolve & SRC-Reweight	0.7848	0.9089	0.9041	0.3051	0.3436	6.0627	3.3465	1.3114	1.0807

C.3 Discussion on applied rewards

We apply SRC-Reweight and SRC-Evolve only to PaP rollout groups, using PickScore, HPSv2, and CLIPScore; GenEval and OCR prompts retain the backbone's original four-reward training procedure. This scope was chosen empirically after comparing reward combinations. GenEval and OCR measure task-specific correctness with attainable ceilings at 1. At these ceilings, tied scores reduce within-group discrimination even when the three general rewards still distinguish the rollouts, which can limit SRC's minimum-contribution criterion.

D Additional Experimental Results

This section supplements the main results and analyses in Section 5 with results for combining SRC-Reweight with SRC-Evolve, a weight-concentration diagnostic, and the larger-model evaluation.

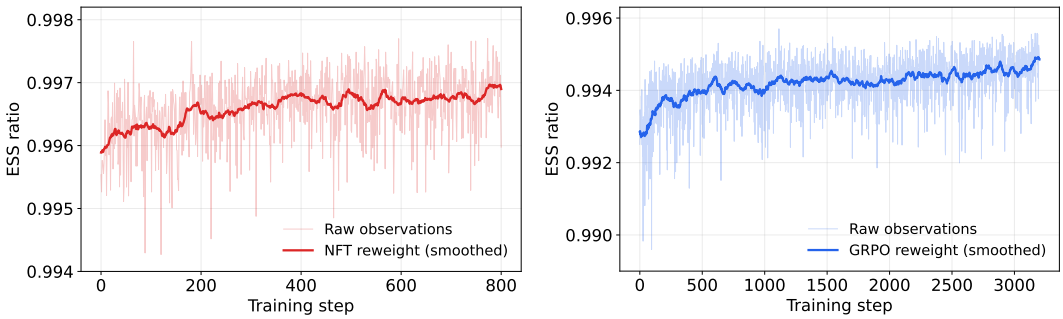
D.1 Combining SRC-Reweight with SRC-Evolve

Table 5 includes the 3× baselines, SRC-Evolve, and SRC-Evolve + SRC-Reweight. The baseline and SRC-Evolve results are repeated from Table 1. Table shading and Overall follow the conventions of Table 1. In SRC-Evolve + SRC-Reweight, reweighting is applied directly to the population P_T selected by SRC-Evolve within each update, with SRC recomputed using the reward statistics of P_T. This requires no additional sampling or reward evaluations. Both methods generate $3K$ rollouts per PaP prompt and K per GenEval or OCR prompt. Static reward weights and evaluation seeds are held fixed between the two methods. Under DiffusionNFT, adding SRC-Reweight raises four of the eight reported rewards but changes Overall from 1.0907 to 1.0889; under GRPO-Guard, it raises two rewards and changes Overall from 1.0814 to 1.0807. Thus, reweighting after selection has a mixed effect on individual rewards and slightly lowers Overall in these runs. These mean comparisons are descriptive: the available aggregate results do not support uncertainty estimates for the differences, including those against the 3× baselines.

D.2 Effective sample size

We use effective sample size only as a weight-concentration diagnostic for SRC-Reweight. For normalized rollout weights, it is $\mathrm{ESS}(\boldsymbol{\omega}) = 1/\sum_{i=1}^{K} \omega_i^2$. Uniform weighting gives $\mathrm{ESS} = K$, while a smaller value indicates concentration on fewer rollouts. We report ESS/K only across PaP groups where reweighting is applied, to check that SRC-Reweight does not substantially reduce the effective number of rollouts relative to uniform weighting.

Across logged training steps, DiffusionNFT has mean ESS/K of 0.9966 (observed range 0.9943–0.9977), while GRPO-Guard has mean ESS/K of 0.9942 (observed range 0.9896–0.9957). Thus, for both backbones, SRC-Reweight preserves nearly the full effective rollout-group size throughout training; its performance gains are not accompanied by substantial weight concentration.



(a) DiffusionNFT + SRC-Reweight.

(b) GRPO-Guard + SRC-Reweight.

Figure 7: **Normalized effective sample size ratio during post-training.** Light curves show the recorded ESS/K observations and dark curves show smoothed trends for visualization.

D.3 Evaluations on a larger model

We evaluate post-training on Klein9B, the undistilled checkpoint of the FLUX.2 [klein] 9B family (Black Forest Labs, 2026). The baseline post-training method is Flow-GRPO (Liu et al., 2025); SRC-Reweight and SRC-Evolve are applied to this backbone. The runs mix PaP, GenEval, and OCR prompts at a 1:1:1 ratio. The training rewards are PickScore, a Realism reward, GenEval, and OCR. The Realism reward combines scores from OmniAID (Guo et al., 2025) and Forensic-Chat (Lin et al., 2025). Among the reported columns, GenEval, PickScore, and OCR are training rewards; Realism has no separate result column.

Table 6 reports Qwen-Image-Bench scores (Li et al., 2026a) and the other evaluation results. Scores follow the reporting scales of the respective benchmarks. The Qwen-Image-Bench Overall score increases under both SRC variants, with the largest value for SRC-Evolve. Gains are mixed across the remaining measures: both variants improve OCR, PickScore, and aesthetic scores over the baseline, but both decrease GenEval and Creative Generation. Because this setting changes the model, post-training backbone, and reward combination together, it provides evidence in an additional configuration rather than isolating the effect of model size.

Table 6: Evaluation of Klein9B and its post-trained variants. **Bold** and underlined values indicate the best and second-best scores among the Flow-GRPO and SRC rows, respectively, using the displayed precision.

Evaluation set	Qwen-Image-Bench					
Method	Overall	Quality	Aesth.	Alignment	Fidelity	Creative
Klein9B	49.80	49.80	52.52	50.45	46.10	46.53
w/ Flow-GRPO	50.98	51.53	53.82	51.97	44.98	**48.00**
+ SRC-Reweight	51.14	51.56	53.98	52.17	**45.90**	47.68
+ SRC-Evolve	**51.35**	**52.02**	**54.22**	**52.34**	45.63	47.54

Evaluation set	GenEval	DPGBench	DrawBench			OCR
Method	GenEval	DPGBench	HPSv2	PickScore	Aesthetic	OCR
Klein9B	81.24	85.65	28.52	22.64	5.26	70.60
w/ Flow-GRPO	**84.54**	86.71	30.82	23.19	5.43	77.21
+ SRC-Reweight	83.91	**86.82**	**31.14**	**23.27**	**5.48**	77.51
+ SRC-Evolve	84.22	86.71	30.72	23.21	5.44	**77.87**

D.4 QUALITATIVE COMPARISONS

Figure 8 shows example outputs from the DiffusionNFT backbone and its SRC-Reweight and SRC-Evolve variants on SD3.5-M. Each row uses the same prompt across all three columns.



Figure 8: **Qualitative comparison on SD3.5-M after post-training with SRC variants.** The first two rows are from DrawBench, the third is from GenEval, and the fourth is from the OCR prompt set. These examples illustrate differences in visual appearance and prompt following.

Figure 9 compares outputs from Flow-GRPO and its SRC-Reweight and SRC-Evolve variants on Klein9B. Each row uses the same prompt across all three columns. The examples span DrawBench, Qwen-Image-Bench, DPGBench, and the OCR prompt set.

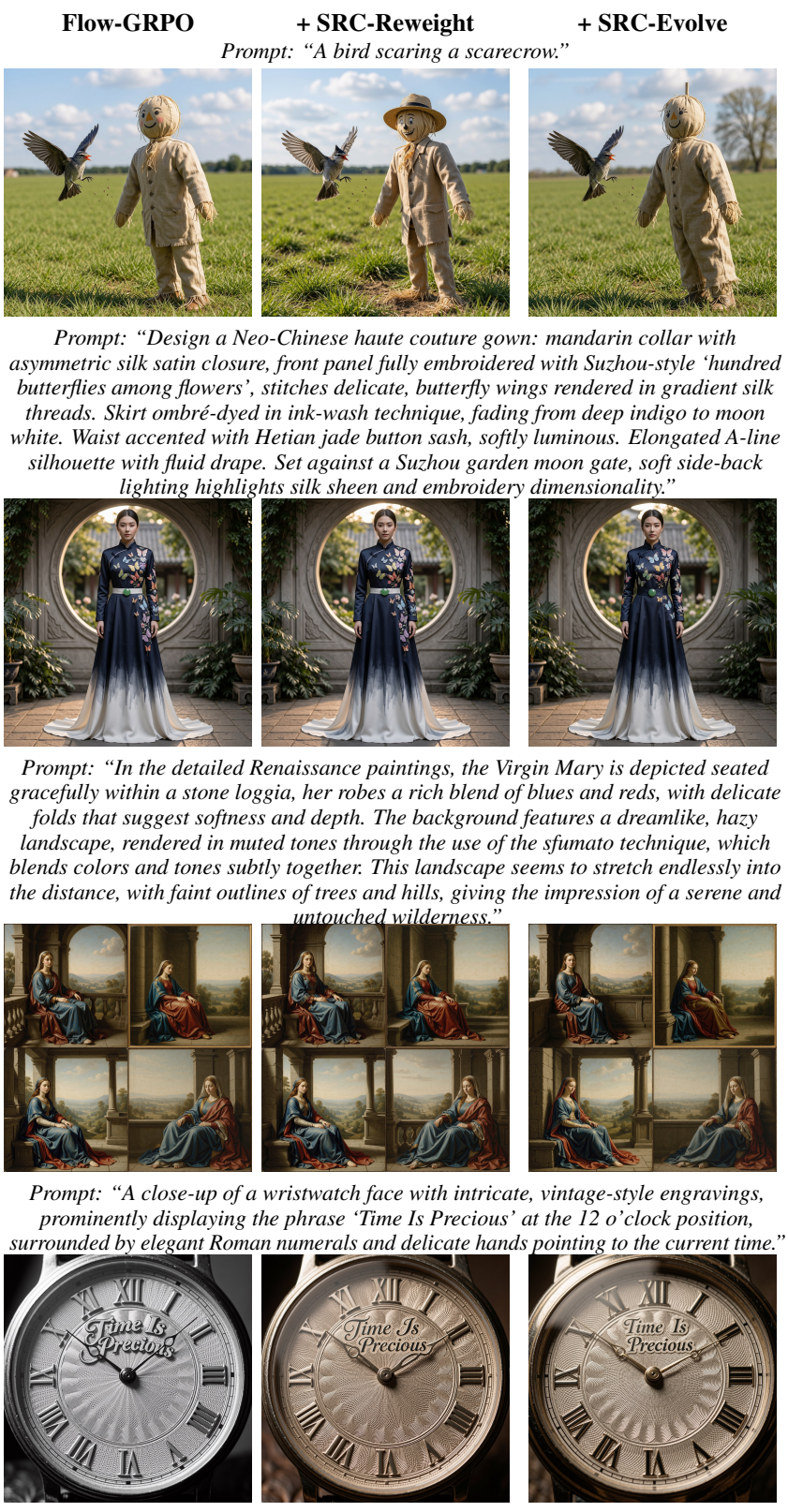


Figure 9: **Qualitative comparison on Klein9B after Flow-GRPO post-training with SRC variants.** The rows show DrawBench, Qwen-Image-Bench, DPGBench, and OCR prompts, respectively. Each DPGBench image is a four-panel grid.