Calibrated Multi-Preference Optimization for Aligning Diffusion Models

Supplementary Materials

A. Additional description

In this section, we provide additional details to Sec. 3 and Sec. 4 of the main manuscript. Specifically, we review the preliminaries on diffusion models and flow-based models (Sec. A.1), preference optimization for diffusion models (Sec. A.2), and provide details on loss weighting scheme (Sec. A.3).

A.1. Background on diffusion and flow-based models

Diffusion models. Let $q(\mathbf{x})$ be the density of data distribution of a sample $\mathbf{x}$ and $p_\theta(\mathbf{x})$ be a generative model parameterized by θ that approximates q. Given $\mathbf{x} \sim q(\mathbf{x})$, the diffusion model considers a series of latent variables $\mathbf{x}_t$ at time $t \in [0, 1]$. Specifically, the forward process forms a conditional distribution $q(\mathbf{x}_t|\mathbf{x})$, where the marginal distribution is given by

$$\mathbf{x}_t = \alpha_t \mathbf{x} + \sigma_t \boldsymbol{\epsilon}, \tag{9}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and α_t, σ_t are noise scheduling functions such that satisfies $\alpha_0 \approx 1$, $\alpha_1 \approx 0$, and $\sigma_0 \approx 0$, $\sigma_1 \approx 1$. Let us denote $\lambda_t = \log(\alpha_t^2/\sigma_t^2)$ log signal-to-noise ratio (log-SNR), then λ_t is a decreasing function of t. Here, α_t and σ_t (or equivalently λ_t) is chosen to satisfy that $\mathbf{x}_1$ is indiscernible from Gaussian noise (*i.e.*, $p(\mathbf{x}_1) \approx \mathcal{N}(\mathbf{0}, \mathbf{I})$), and conversely, $\mathbf{x}_0$ matches the data density $q(\mathbf{x})$. Then the reverse generative process gradually denoises the random Gaussian noise $\mathbf{x}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ to recover $\mathbf{x}_0$. Specifically, the sampling process is governed by solving time-discretized SDE [17, 60] or probability flow ODE [22, 58], by using the score function $\nabla \log q(\mathbf{x}_t)$. Training diffusion model then optimizes the neural network to approximate the score function by $\mathbf{s}_\theta(\mathbf{x}_t; t)$. Especially, using the noise-prediction model [17] is a common practice, where the training objective can be written as following weighted loss objective [27]:

$$\mathcal{L}_{\text{DM}}(\theta; \mathbf{x}) = \mathbb{E}_{t\sim\mathcal{U}(0,1),\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\big[-\tfrac{1}{2} w_t \lambda_t' \|\boldsymbol{\epsilon}_\theta(\mathbf{x}_t; t) - \boldsymbol{\epsilon}\|_2^2\big], \tag{10}$$

where w_t is a weighting function and λ_t' is a time-derivative of λ_t. Note that when $w_t = 1$ for all $t \in (0, 1)$, it becomes the variational lower bound (vlb) of KL divergence [25], and the original DDPM uses $w_t \lambda_t' = -1$.

Flow models. Alternatively, flow-based models or stochastic interpolants [1, 38, 40] consider approximating the velocity field $\boldsymbol{v}(\mathbf{x}_t, t)$ on $\mathbf{x}$ at time $t \in (0, 1)$, and solve following probability flow ODE to transport noise to data distribution:

$$\mathbf{x}_t' = \boldsymbol{v}(\mathbf{x}_t, t), \tag{11}$$

where the marginal distribution of the solution of ODE matches the distribution $q_t(\mathbf{x}_t)$. Given $\mathbf{x}_t = \alpha_t \mathbf{x} + \sigma_t \boldsymbol{\epsilon}$ for some $t \in (0, 1)$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, the velocity field satisfies following:

$$\boldsymbol{v}(\mathbf{x}_t, t) = \mathbb{E}[\mathbf{x}_t' \,|\, \mathbf{X}_t = \mathbf{x}_t] = \alpha_t' \,\mathbb{E}[\mathbf{x} \,|\, \mathbf{X}_t = \mathbf{x}_t] + \sigma_t' \,\mathbb{E}[\boldsymbol{\epsilon} \,|\, \mathbf{X}_t = \mathbf{x}_t], \tag{12}$$

and training objective for flow matching model is given as follows:

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}(0,1),\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\big[\|\boldsymbol{v}_\theta(\mathbf{x}_t, t) - (\alpha_t' \mathbf{x} + \sigma_t' \boldsymbol{\epsilon})\|_2^2\big]. \tag{13}$$

Note that Eq. (13) is a special case of Eq. (10), when $w_t = -\frac{1}{2}\lambda_t' \sigma_t^2$ [11, 27]. In case of Rectified Flow [38], we set $\alpha_t = 1 - t$, $\sigma_t = t$, and $\lambda_t = 2\log(\frac{1-t}{t})$, and the training objective of rectified flow model is given as follows:

$$\mathcal{L}_{\text{RF}}(\theta) = \mathbb{E}_{t,\boldsymbol{\epsilon}}[\|\boldsymbol{v}_\theta(\mathbf{x}_t, t) - (\boldsymbol{\epsilon} - \mathbf{x})\|_2^2]. \tag{14}$$

For SD3-M [11], we use Eq. (14) to compute the loss.

A.2. Diffusion preference optimization

For preference optimization with diffusion models, we consider following relaxation of original RLHF objective:

$$\max_\theta \bar{R}(\mathbf{x}_{0:1}, \mathbf{c}) - \beta D_{\text{KL}}\big(p_\theta(\mathbf{x}_{0:1}|\mathbf{c}) \,\|\, p_{\text{ref}}(\mathbf{x}_{0:1}|\mathbf{c})\big), \tag{15}$$

where $\bar{R}(\mathbf{x}_{0:1}, \mathbf{c})$ satisfies following:

$$R(\mathbf{x}, \mathbf{c}) = \mathbb{E}_{q(\mathbf{x}_{0:1}|\mathbf{x})}\big[\bar{R}(\mathbf{x}_{0:1}, \mathbf{c})\big]. \tag{16}$$

Then by rearranging the equation derived from the closed solution of Eq. (15), we have following:

$$\bar{R}(\mathbf{x}_{0:1}, \mathbf{c}) = \beta \log \frac{p_\theta(\mathbf{x}_{0:1}|\mathbf{c})}{p_{\text{ref}}(\mathbf{x}_{0:1}|\mathbf{c})} - \beta \log Z(\mathbf{c}), \tag{17}$$

where $Z(\mathbf{c})$ is a partition function. From Eq. (17) and by rearranging $q(\mathbf{x}_{0:1}|\mathbf{x})$ in the inside term, we have

$$\begin{aligned}\mathbb{E}_{q(\mathbf{x}_{0:1}|\mathbf{x})}[\bar{R}(\mathbf{x}_{0:1}, \mathbf{c}) - \beta \log Z(\mathbf{c})] &= \mathbb{E}_{q(\mathbf{x}_{0:1}|\mathbf{x})}\left[\beta \log \frac{p_\theta(\mathbf{x}_{0:1}|\mathbf{c})}{q(\mathbf{x}_{0:1}|\mathbf{x})} - \beta \log \frac{p_{\text{ref}}(\mathbf{x}_{0:1}|\mathbf{c})}{q(\mathbf{x}_{0:1}|\mathbf{x})}\right] \\ &= \beta\big(D_{\text{KL}}(q(\mathbf{x}_{0:1}|\mathbf{x}) \,\|\, p_{\text{ref}}(\mathbf{x}_{0:1}|\mathbf{c})) - D_{\text{KL}}(q(\mathbf{x}_{0:1}|\mathbf{x}) \,\|\, p_\theta(\mathbf{x}_{0:1}|\mathbf{c}))\big).\end{aligned} \tag{18}$$

Note that the KL divergence satisfies following (see [27] for details):

$$\frac{\mathrm{d}}{\mathrm{d}t} D_{\text{KL}}\big(q(\mathbf{x}_{t:1}|\mathbf{x} \,\|\, p_\theta(\mathbf{x}_{t:1}|\mathbf{c})\big) = \frac{1}{2}\lambda_t' \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\big[\|\boldsymbol{\epsilon}_\theta(\mathbf{x}_t; \mathbf{c}, t) - \boldsymbol{\epsilon}\|_2^2\big]. \tag{19}$$

By taking integration of Eq. (19) over $t \in (1, 0)$, one can rewrite $R(\mathbf{x}, \mathbf{c})$ as follows:

$$R(\mathbf{x}, \mathbf{c}) = \frac{\beta}{2}\mathbb{E}_{t\sim\mathcal{U}(0,1),\boldsymbol{\epsilon}\sim\mathcal{N}(0,1)}\big[\lambda_t'\big(\|\boldsymbol{\epsilon}_\theta(\mathbf{x}_t; \mathbf{c}, t) - \boldsymbol{\epsilon}\|_2^2 - \|\boldsymbol{\epsilon}_\phi(\mathbf{x}_t; \mathbf{c}, t) - \boldsymbol{\epsilon}\|_2^2\big)\big], \tag{20}$$

For a triplet $(\mathbf{c}, \mathbf{x}^+, \mathbf{x}^-)$, we consider following upper bound of a training objective for any convex function $g : \mathbb{R} \to \mathbb{R}$:

$$\begin{aligned}\bar{\ell}(\theta) &= g\big(R(\mathbf{x}^+, \mathbf{c}) - R(\mathbf{x}^-, \mathbf{c})\big) \\ &= g\left(\frac{\beta}{2}\mathbb{E}_{t,\boldsymbol{\epsilon}^+,\boldsymbol{\epsilon}^-}\big[\lambda_t'\big(\|\boldsymbol{\epsilon}_\theta(\mathbf{x}_t^+; \mathbf{c}, t) - \boldsymbol{\epsilon}^+\|_2^2 - \|\boldsymbol{\epsilon}_{\text{ref}}(\mathbf{x}_t^+; \mathbf{c}, t) - \boldsymbol{\epsilon}^+\|_2^2 - \|\boldsymbol{\epsilon}_\theta(\mathbf{x}_t^-; \mathbf{c}, t) - \boldsymbol{\epsilon}^-\|_2^2 + \|\boldsymbol{\epsilon}_{\text{ref}}(\mathbf{x}_t^-; \mathbf{c}, t) - \boldsymbol{\epsilon}^-\|_2^2\big)\big]\right) \\ &\le \mathbb{E}_{t,\boldsymbol{\epsilon}^+,\boldsymbol{\epsilon}^-}\left[g\left(\tfrac{1}{2}\beta\lambda_t'\big(\|\boldsymbol{\epsilon}_\theta(\mathbf{x}_t^+; \mathbf{c}, t) - \boldsymbol{\epsilon}^+\|_2^2 - \|\boldsymbol{\epsilon}_{\text{ref}}(\mathbf{x}_t^+; \mathbf{c}, t) - \boldsymbol{\epsilon}^+\|_2^2 - \|\boldsymbol{\epsilon}_\theta(\mathbf{x}_t^-; \mathbf{c}, t) - \boldsymbol{\epsilon}^-\|_2^2 + \|\boldsymbol{\epsilon}_{\text{ref}}(\mathbf{x}_t^-; \mathbf{c}, t) - \boldsymbol{\epsilon}^-\|_2^2\big)\right)\right],\end{aligned}$$

where $t \sim \mathcal{U}(0, 1)$, $\boldsymbol{\epsilon}^+ \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $\boldsymbol{\epsilon}^- \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and the last inequality comes from the Jensen's inequality. Using the equation we defined in our main paper, *i.e.*,

$$R_\theta(\mathbf{x}_t, \mathbf{c}, t) = \lambda_t'\big(\|\boldsymbol{\epsilon}_\theta(\mathbf{x}_t; \mathbf{c}, t) - \boldsymbol{\epsilon}\|_2^2 - \|\boldsymbol{\epsilon}_{\text{ref}}(\mathbf{x}_t; \mathbf{c}, t) - \boldsymbol{\epsilon}\|_2^2\big), \tag{21}$$

we derive following training objectives for DPO, IPO, and CaPO:

$$\begin{aligned}\ell_{\text{DPO}}(\theta) &= \mathbb{E}_{t,\boldsymbol{\epsilon}^+,\boldsymbol{\epsilon}^-}\Big[-\log\sigma\big(\beta\big(R_\theta(\mathbf{x}^+, \mathbf{c}, t) - R_\theta(\mathbf{x}^-, \mathbf{c}, t)\big)\big)\Big] \\ \ell_{\text{IPO}}(\theta) &= \mathbb{E}_{t,\boldsymbol{\epsilon}^+,\boldsymbol{\epsilon}^-}\left[\Big(1 - \beta\big(R_\theta(\mathbf{x}^+, \mathbf{c}, t) - R_\theta(\mathbf{x}^-, \mathbf{c}, t)\big)\Big)^2\right] \\ \ell_{\text{CaPO}}(\theta) &= \mathbb{E}_{t,\boldsymbol{\epsilon}^+,\boldsymbol{\epsilon}^-}\left[\Big(R(\mathbf{x}^+, \mathbf{c}) - R(\mathbf{x}^-, \mathbf{c}) - \beta\big(R_\theta(\mathbf{x}^+, \mathbf{c}, t) - R_\theta(\mathbf{x}^-, \mathbf{c}, t)\big)\Big)^2\right],\end{aligned} \tag{22}$$

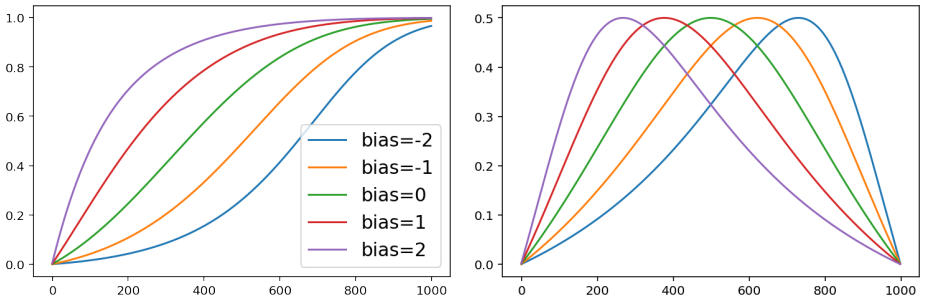
where $R(\mathbf{x}, \mathbf{c})$ is a reward from the external reward model.

Independent noise sampling. Note that in original Diffusion-DPO paper [63], the author proposed to use same noise for $\mathbf{x}^+$ and $\mathbf{x}^-$, *i.e.*, $\boldsymbol{\epsilon}^+ = \boldsymbol{\epsilon}^-$, while we sample independent noise for $\boldsymbol{\epsilon}^+$ and $\boldsymbol{\epsilon}^-$. We believe this is more theoretically grounded, and empirically found that it has slightly better performance than using the same noise (even for DPO and IPO).

A.3. Loss weighting

In practice, we multiply w_t to the noise-prediction loss for diffusion preference optimization. One can consider this as setting timestep-wise different $\beta_t = \beta w_t$, *i.e.*, giving different regularization hyperparameters at each time $t \in (0, 1)$. Thus, we have

$$R_\theta(\mathbf{x}_t, \mathbf{c}, t) = w_t\lambda_t'\big(\|\boldsymbol{\epsilon}_\theta(\mathbf{x}_t; \mathbf{c}, t) - \boldsymbol{\epsilon}\|_2^2 - \|\boldsymbol{\epsilon}_{\text{ref}}(\mathbf{x}_t; \mathbf{c}, t) - \boldsymbol{\epsilon}\|_2^2\big), \tag{23}$$



(a) Weighting for noise-prediction loss (b) Weighting for flow matching loss

Figure 5. **Loss weighting.** We plot the weighting function with bias $b \in \{-2, -1, 0, 1, 2\}$ for each noise prediction loss and flow matching loss.

and applies to each DPO, IPO, and CaPO loss. As we mentioned in Sec. 4.4 in our main draft, we use sigmoid loss weighting [27], where the loss weights are sigmoid function of log-SNR λ_t with bias b:

$$w_t = w(\lambda_t) = \frac{1}{1 + \exp(b - \lambda_t)}. \tag{24}$$

Note that SDXL uses a modified DDPM schedule [17], where $\beta = \left(\sqrt{\beta_0} + \frac{t}{T-1}(\sqrt{\beta_{T-1}} - \sqrt{\beta_0})\right)^2$, and $\alpha_t = (\prod_{s=0}^{t}(1 - \beta_s))^{1/2}$. Since it is impractical to compute λ_t', we simply set it as constant (*i.e.*, linear λ_t, which empirically holds when $\lambda_t \in [-15, 5]$, and for $\lambda_t > 0.5$ the weight w_t is close to 0, so one can ignore it).

SD3-M uses a rectified flow scheduler [38], where $\lambda_t = 2\log(\frac{1-t}{t})$. Note that we have

$$\mathbb{E}_{t\sim\mathcal{U}(0,1),\boldsymbol{\epsilon}}\left[\|\boldsymbol{v}_\theta(\mathbf{x}_t, t) - (\boldsymbol{\epsilon} - \mathbf{x})\|_2^2\right] = \mathbb{E}_{\lambda\sim\mathcal{U}(\lambda_{\text{min}},\lambda_{\text{max}}),\boldsymbol{\epsilon}}\left[e^{-\lambda/2}\|\boldsymbol{\epsilon}_\theta(\mathbf{x}_\lambda, \lambda) - \boldsymbol{\epsilon}\|_2^2\right], \tag{25}$$

where $\mathbf{x}_\lambda$ denotes forward process of $\mathbf{x}$ with log-SNR value λ, and λ_{min} and λ_{max} denotes the minimal and maximal value for log-SNR (see [27] Appendix D.3 for details). As such, multiplying $w = \sigma(-\lambda)$ to noise-prediction loss is equivalent to multiplying $(e^{\lambda/2} + e^{-\lambda/2})^{-1}$ to flow matching objective:

$$\sigma(-\lambda)\|\boldsymbol{\epsilon}_\theta(\mathbf{x}_\lambda, \lambda) - \boldsymbol{\epsilon}\|_2^2 = \sigma(-\lambda) \cdot \frac{e^{-\lambda/2}}{e^{-\lambda/2}}\|\boldsymbol{\epsilon}_\theta(\mathbf{x}_\lambda, \lambda) - \boldsymbol{\epsilon}\|_2^2 = \frac{1}{e^{\lambda/2} + e^{-\lambda/2}}\|\boldsymbol{v}_\theta(\mathbf{x}_\lambda, \lambda) - (\boldsymbol{\epsilon} - \mathbf{x})\|_2^2. \tag{26}$$

If we shift with bias b, it becomes $w_\lambda = (e^{-(\lambda-b)/2} + e^{(\lambda-b)/2})^{-1}$. In Fig. 5, we plot loss weighting functions for noise-prediction loss and flow matching loss with different bias $b \in \{-2, -1, 0, 1, 2\}$. In practice, we select $\lambda \sim \mathcal{U}[-10, 10]$ and multiply w_λ to flow matching loss. Note that multiplying w_λ has a similar effect in log-normal sampling proposed in [11, 22], where we empirically find similar performance. To ensure consistency with SDXL experiments, we use loss weighting instead of logit-normal sampling for SD3-M experiments.

B. Implementation Details

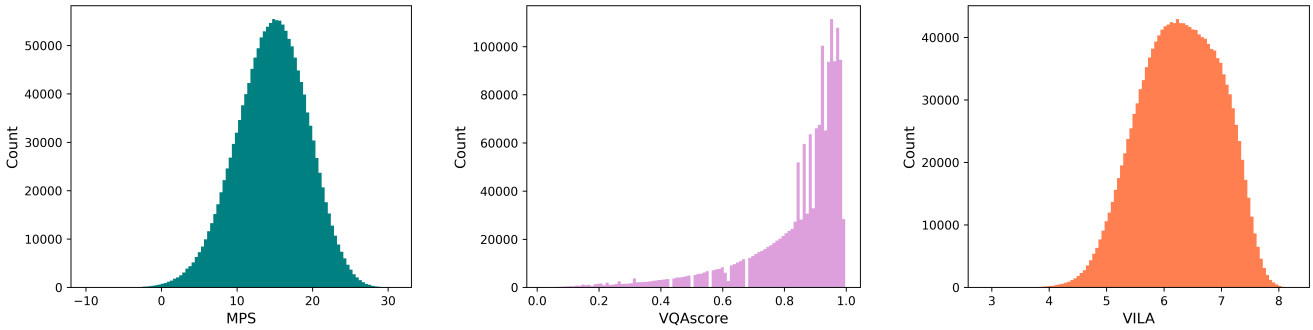
B.1. Dataset

Reward models. For reward models learned by fine-tuning CLIP models (*e.g.*, Pickscore [28], MPS [71]), we compute the reward by the dot product between the image embedding and the text embedding. To compute MPS score, we additionally multiply the text embedding from condition textual description (*e.g.*, textual description for aesthetic quality). For VQAscore, we use CLIP-FlanT5-XXL [37], and compute the score by probability of `"Yes"` token given the image and question provided to the model:

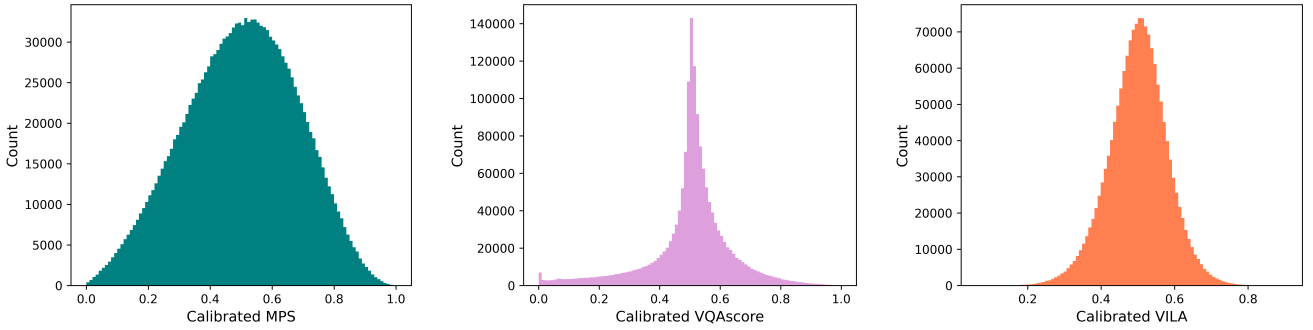
$$P\big(\texttt{"Yes"}|\mathbf{x}, \texttt{"Does this figure shows \{prompt\}? Please answer yes or no."}\big), \tag{27}$$

where $\mathbf{x}$, `prompt` are image and text input. While VQAscore is not a Bradley-Terry model, we simply approximate the win-rate by following:

$$\mathbb{P}(\mathbf{x} \succ \mathbf{x}'|\mathbf{c}) = \frac{s(\mathbf{x}, \mathbf{c})^\alpha}{s(\mathbf{x}, \mathbf{c})^\alpha + s(\mathbf{x}', \mathbf{c})^\alpha}, \tag{28}$$



(a) MPS scores of training dataset (b) VQAscores of training dataset (c) VILA scores of training dataset



(d) Calibrated MPS scores of training dataset (e) Calibrated VQAscores of training dataset (f) Calibrated VILA scores of training dataset

Figure 6. **Training dataset score distribution.** We plot the histogram of rewards (top row) and calibrated rewards (bottom row) of training dataset. By using calibration, the scores are centered and bounded in range $[0, 1]$.

where $s(\mathbf{x}, \mathbf{c})$ is a VQAscore and $\alpha > 0$ is a hyperparameter to control the temperature. We find $\alpha = 1$ works well in our experiments. Lastly, VILA-R score [24] outputs the aesthetic score between 1-10, and we apply the Bradley-Terry model to compute the win-rate. In Fig. 6, we plot the histogram of reward scores and calibrated rewards of our training dataset.

Training dataset. We use 100K prompts from DiffusionDB [64] and generate $N = 16$ images per prompt. For SDXL, we use DDIM [58] scheduler, guidance scale of 7.5 and sampling steps of 50. For SD3-M, we use the DPM solver [39] for flow-based models, guidance scale of 5.0 and sampling steps of 50. Furthermore, as described in [11], we shift the timestep schedules to reside more on higher timesteps, *i.e.*, we set $t \leftarrow \frac{ts}{1+t(s-1)}$ with shift scale $s = 3.0$.

B.2. Training and evaluation

Training configuration. Throughout experiments, we use Jax [14] and train models using the Optax library on TPU chips. For both SDXL and SD3-M experiments, we use Adam [26] optimizer. Regarding training configuration for SDXL experiments, we use batch size of 256, learning rate of 1e-5 with linear warmup for first 1000 steps, and train for maximum 10000 steps. For SD3-M, we use batch size of 256, learning rate of 1.5e-5 with linear warmup for first 1000steps, and train for maximum 5000 steps. We choose hyperparameter β by sweeping over $\{300, 500, 1000\}$ for CaPO, $\{500, 1000, 2000\}$ for IPO, and $\{2000, 3000, 4000\}$ for DPO when training SDXL model. For SD3-M, we sweep over $\{30, 50, 100\}$ for CaPO, $\{50, 100, 200\}$ for IPO, and $\{100, 200, 300\}$ for DPO. For all training, we use sigmoid loss weighting with $b = 1.5$ for SDXL and $b = -1.0$ for SD3-M (including all DPO, IPO, and CaPO). During training, we generate images using subset of Parti prompts at each 1000-th iteration, and choose the final model with maximum validation win-rate (average of win-rates for multi-reward signals).

Model soup. For model merging experiments, we follow [51]. Specifically, suppose θ_0 be weights of a pretrained model and θ_1, θ_2 be weights of fine-tuned models. Then the spherical linear interpolation (SLERP) between θ_1 and θ_2 is given by

$$\text{SLERP}(\theta_0, \theta_1, \theta_2, \lambda) = \theta_0 + \frac{\sin((1-\lambda)\Omega)}{\sin(\Omega)}(\theta_1 - \theta_0) + \frac{\sin(\lambda\Omega)}{\sin(\Omega)}(\theta_2 - \theta_0), \tag{29}$$

where Ω is the angle between two task vectors $\theta_1 - \theta_0$ and $\theta_2 - \theta_0$, and $\lambda \in (0, 1)$ is a coefficient. To merge three fine-tuned models, we first merge two models with $\lambda = 0.5$ to obtain $\theta_{12} = \text{SLERP}(\theta_0, \theta_1, \theta_2, 0.5)$, then merge θ_{12} and θ_3 with $\lambda = 1/3$ to obtain final model.

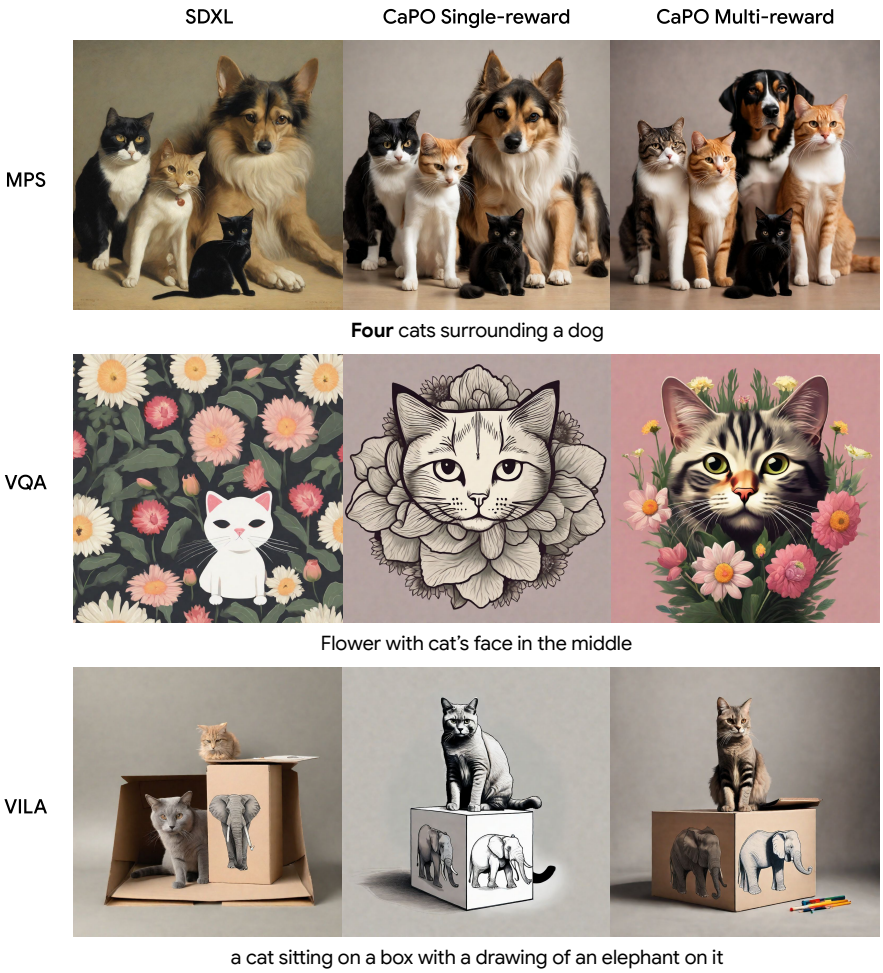


Figure 7. **Effect of multi-reward CaPO**. We demonstrate the qualitative comparison between CaPO with single-reward with each MPS, VQAscore, and VILA, and CaPO with multi-reward on SDXL. We see that optimizing with single-reward improves upon the base model, yet multi-reward CaPO shows the best overall improvement. For example, while using VQAscore alone improves the image-text alignment, the image aesthetics are significantly improved when using multi-reward. Also, when using VILA score, the image aesthetics improve, but it often lose the image-text alignment (*e.g.*, the image becomes drawing style, while only the elephant should be in a drawing style).

Evaluation. For evaluation, we generate images with the same configuration as in Sec. B.1 for different benchmark prompt dataset. For SDXL, we use Parti [69] prompts, and for SD3-M we use DPG-bench [20]. Then we compute the win-rate against the base model by comparing one-by-one comparison for each image, *e.g.*, if we have K images from base model and K images from fine-tuned model, we make K^2 comparison and count the number of win and divide by K^2. We also report the average reward scores. Remark that the average reward scores and win-rate could show different trends, as the model achieves a higher score gain for some prompts, but it fails to improve on others. Thus, we found win-rate is a more general metric to see the generalization over different prompts.

Benchmark evaluation. We use GenEval [16] and T2I-Compbench [21] to evaluate our models. For T2I-Compbench, we use BLIP-VQA model [34] to evaluate Color, Shape, Texture, Complex, and UniDet [72] for Spatial, and CLIP [46] for Non Spatial. For baselines, we compare with the state-of-the-art open-source text-to-image diffusion models Flux-dev (12B) [29], Flux-schnell (12B) [29], and Stable Diffusion 3.5-Large (8B) [11]. Since those models are much larger than SDXL (2.6B) and SD3 (2B), we remark that this is not a fair comparison, yet we show the comparable performance of our method.

	MPS	VQA	VILA
Constant weighting	54.9	53.6	55.7
Sigmoid weighting ($b = 0$)	57.0	53.9	66.9
Sigmoid weighting ($b = -1.0$)	59.0	55.7	69.3

Table 6. **Ablation on loss weighting for SD3-M.** We show the results of CaPO multi-reward fine-tuning SD3-M with constant weighting (*i.e.*, $-w_t\lambda'_t = 1$), and sigmoid weighting by varying bias $b = 0.0, -1.0$. Similar to SDXL, using sigmoid weighting shows better results than constant weighting, and $b = -1.0$ performs the best.

CaPO+SDXL	SDXL	Improvement	CaPO+SDXL	Diffuion-DPO	Improvement	CaPO+SD3-M	SD3-M	Improvement
54.5%	45.5%	**+10%**	52.0%	48.0%	**+4%**	53.5%	46.5%	**+7%**

Table 7. **User study results.** We report the win-rate from the user study by using 200 images. We compare CaPO+SD3-M vs SD3-M, CaPO+SDXL vs SDXL, and CaPO+SDXL vs Diffusion-DPO [63]. CaPO achieves consistent win against baseline.

C. Additional ablation study

Effect of multi-reward. We demonstrate the effect of multi-reward CaPO compared to single-reward CaPO. As we demonstrated in Tab. 1 and Tab. 2 in our main draft, the single-reward model achieves the best score in which they have trained with, but the other metrics score below the multi-reward cases. We showcase the qualitative examples on the effect of multi-reward preference optimization compared to single-reward cases in Fig. 7. We notice that single-reward fine-tuning is often imperfect, *e.g.*, fine-tuning with only VILA score loses image-text alignment, and fine-tuning with only VQAscore lacks image aesthetics. On the other hand, multi-reward fine-tuning complements those issues and improves the overall image quality.

Loss weighting for SD3-M. We show the effect of loss weighting when training SD3-M models. Similar to SDXL, we compare the results of CaPO multi-reward fine-tuning with different bias parameters. In Tab. 6, we show that sigmoid weighting with bias $b = -1.0$ shows the best result, outperforming the constant weighting counterpart. Note that for SDXL, $b = 1.5$ performs the best, while for SD3-M, negative bias $b = -1.0$ performs the best. Remark that as SD3-M performs diffusion modeling on $16 \times 128 \times 128$, and SDXL performs on $4 \times 128 \times 128$, the bias shifts toward negative as the total variance becomes higher, and the log-SNR should be increased [18, 19].

User study evaluation. We conduct additional user evaluations to compare our method with base models. For SDXL vs CaPO+SDXL, we randomly select 200 prompts from Parti prompt dataset [69], and for SD3-M vs CaPO+SD3-M, we randomly select 200 prompts from GenAI bench prompt dataset [33]. Additionally, we compare CaPO+SDXL with Diffusion-DPO [63] again with 200 randomly selected prompts. We give following instructions to the raters:

- Instruction: Given the text below, pick the left or the right image with better looking.
- Good example: Images are beautiful and following text description.
- Bad example: Images are not looking good or not following text description.

We use Amazon mechanical Turk [2] and 5 raters answered to each pair. In Tab. 7, we show the results of user study. We observe that CaPO+SD3-M and CaPO+SDXL consistently outperform SD3-M and SDXL, respectively. Also, CaPO+SDXL outperforms Diffusion-DPO, yet the margin is smaller than CaPO+SDXL vs SDXL.

Three cameras on the table

A cat with visible ears is **riding**

... room with a **painting of a corgi** on the wall above a couch and a round coffee table in front of a couch...

A rabbit in a **fluffy dress** is hopping through a garden of flowers.

A hiking trail marker with '**Journey Begins Here.**'

Five purple umbrellas open in a line.

a painting of a house on a mountain (**Aesthetic ↑**)

A train going to the moon (**Aesthetic ↑**)

Figure 8. **Additional qualitative comparison between CaPO SDXL and SDXL.** We provide additional qualitative comparison between CaPO SDXL and SDXL. The CaPO SDXL model demonstrates better image-text alignment (*e.g.*, counting, attribute binding, etc), as well as image aesthetics (*e.g.*, artistic style, detail, etc). We bold the text to highlight the prompts that demonstrate improvement in image-text alignment, and (**Aesthetic** ↑) to demonstrate the improvement in image aesthetic quality.

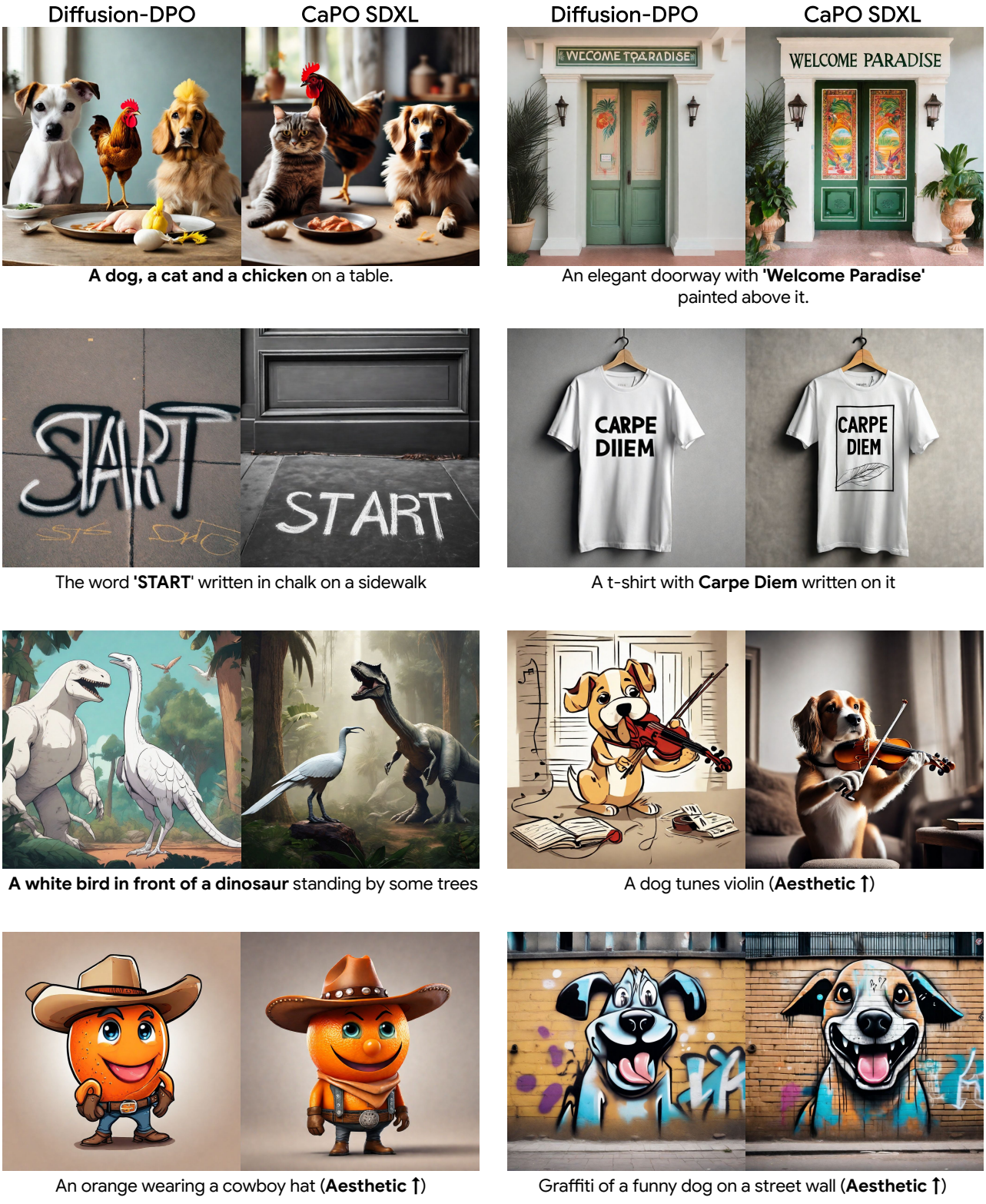


Figure 9. **Additional qualitative comparison between CaPO SDXL and Diffusion-DPO [63].** We provide additional qualitative comparison between CaPO SDXL and Diffusion-DPO [63]. CaPO SDXL shows better image-text alignment and image aesthetics compared to Diffusion-DPO without using any human annotated data. We bold the text to highlight the prompts that demonstrate improvement in image-text alignment, and (**Aesthetic** ↑) to demonstrate the improvement in image aesthetic quality.

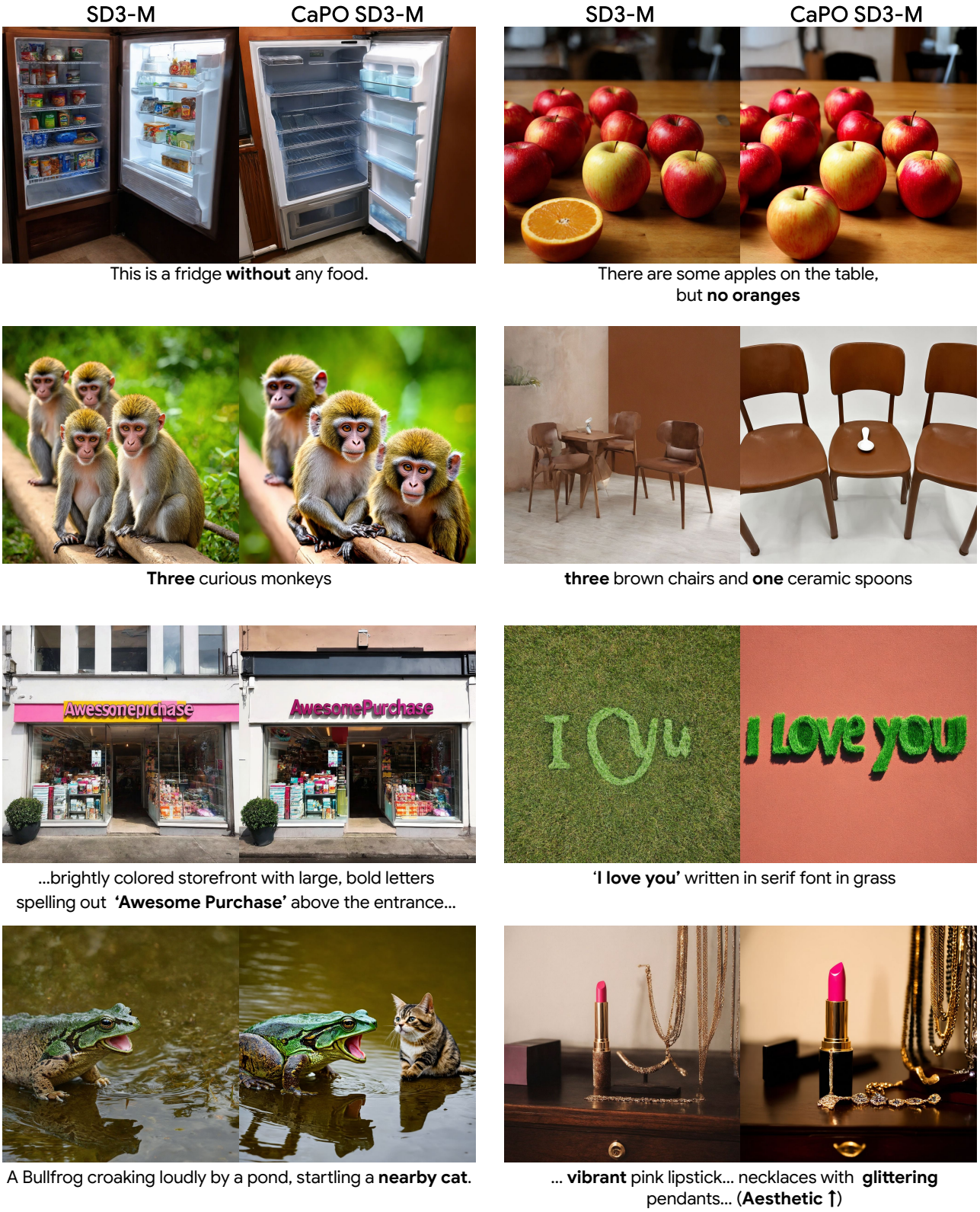


Figure 10. **Additional qualitative comparison between CaPO SD3-M and SD3-M.** We provide additional qualitative comparison between CaPO SD3-M and SD3-M. CaPO SD3-M shows better image-text alignment, *e.g.*, negation (first row), counting (second row), visual text rendering (third row). Also it demonstrates better image aesthetics (fourth row right). We bold the text to highlight the prompts that demonstrate improvement in image-text alignment, and (**Aesthetic** ↑) to demonstrate the improvement in image aesthetic quality.