Selective Timestep Weighting and Advantage-Based Replay for Sample-Efficient Diffusion RLHF

Anonymous submission

Paper ID

Abstract

Reinforcement learning from human feedback (RLHF) is key for aligning generative models with human preferences, but applying it to diffusion models is feedback inefficient. We propose two strategies to improve efficiency while maintaining generalization. First, a per-timestep weighting scheme, theoretically grounded in PPO convergence properties, emphasizes informative denoising steps. Second, a replay mechanism prioritizes high-advantage trajectories, reducing new reward queries. Together, these achieve up to 6× better sample efficiency than standard diffusion RLHF baselines.

1. Introduction

Diffusion models [36] have become the leading framework for high-quality image generation. However, because they are trained to reproduce the distribution of their training data, they do not inherently reflect human preferences. Recent work [2, 8] addresses this limitation through reinforcement learning from human feedback (RLHF) [6], which fine-tunes diffusion models using scalar feedback from human or reward models to explicitly optimize for preference alignment.

A key challenge in diffusion RLHF is credit assignment: since feedback is only given on the final image, methods like DDPO [2] assign uniform loss across all timesteps. This ignores that different timesteps edit the image at different granularities [22], leading to inefficient training.

Prior work addresses this by contrasting paired trajectories from the same initial noise that diverge at a designated branching timestep [13, 45]. However, this only isolates the effect at the branching point. Earlier steps remain identical and uninformative, while later steps still receive uniform advantage. This leaves the broader credit assignment problem unresolved and requires multiple reward evaluations per noise sample, reducing efficiency.

We argue that reward information in diffusion trajectories is inherently unevenly distributed: some denoising steps and trajectories carry significantly more learning signal than others. To exploit this, we propose two complementary strategies that emphasize informative training data. First, we introduce a per-timestep weighting scheme for denoising steps during optimization. We theoretically motivate this by connecting Group Relative Policy Optimization (GRPO) and Proximal Policy Optimization (PPO), showing that each timestep should be weighted proportionally to its TD-error advantage variance. Since this is infeasible to estimate efficiently during training, we approximate it using the squared magnitude of per-timestep latent change.

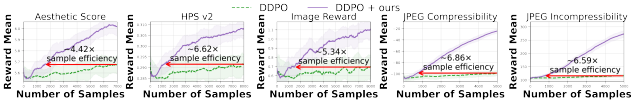


Figure 1. Augmenting existing RLHF frameworks (bottom curve) with our method leads to significant sample efficiency improvement (top curve).

Second, inspired by replay buffers in robotics reinforcement learning (RL) [1, 19, 30], we introduce a trajectory replay mechanism that reuses informative past trajectories rather than discarding them. Following prioritized replay [30], we hard-mine trajectories with the largest advantages [35] for maximal sample efficiency. Together, both strategies reduce the need for new reward evaluations while improving training efficiency.

Overall, our method improves the efficiency of the baseline by 2–6×, as shown in Fig. 1, while remaining simple and compatible with existing diffusion RLHF pipelines.

In summary, our main contributions are:

- A computationally practical and mathematically motivated per-timestep weighting scheme that mitigates the credit assignment problem in diffusion RLHF.
- A replay buffer mechanism that retrieves informative past trajectories during training, reducing the need for repeated reward queries.
- A simple, plug-and-play method that integrates seamlessly into existing diffusion RLHF pipelines without architectural changes.

- Extensive experiments demonstrating consistent gains across diverse reward functions, underscoring the generality of our approach.

2. Related Work

Replay Buffers. Replay buffers are standard in off-policy RL [9, 18, 21] but absent from on-policy methods like PPO [33] and TRPO [32]. Extensions include hindsight experience replay [1], prioritized experience replay [30], energy-based sampling [47], synthetic trajectory generation [19], retrieval-based robotic learning [41], and TD-error-based hard-mining for PPO [17].

RLHF. RLHF was scaled to pretrained LLMs by [24]. GRPO [34] eliminates the reward model, DPO [27] uses pairwise preferences, [4] computes segment-level rewards, and [25] enhances GRPO via prompt hard-mining. In robotics, RLHF has been applied to policy training [15, 39], evolutionary search [44], and diffusion policy finetuning [5, 28].

Diffusion RLHF. [2, 8] first applied GRPO to diffusion models. B2-DiffuRL [13] improves credit assignment via trajectory branching. Preference-based methods [3, 38, 45] learn from image pairs. Extensions cover video [26] and 2-step models [16]. Some methods [7, 20, 42] backpropagate through differentiable rewards, while others [14, 42] localize reward regions. Concurrent work [46] shows timestep reweighting stabilizes preference-based training. TempFlow [10] explores timestep weighting in flow matching but does not employ hard-mining and underperforms our scheme across our 5-reward evaluation.

3. Preliminaries

MDP. An MDP [37] is a tuple (S, A, P, R, γ) with state space S, action space A, transition $P(s'|s, a)$, reward $R(s, a)$, and discount γ.

Diffusion Models. Diffusion models add noise over many steps and train a model to reverse this process. Combined with reward feedback, the reverse process becomes an MDP where actions are predicted noise and states are image latents.

PPO and Diffusion RLHF. PPO optimizes a clipped surrogate objective where $r_t(\theta) = \frac{\pi_\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)}$:

$$L_{\text{PPO}} = \mathbb{E}_{(s,a)\in D}\Big[\min\Big(r_t(\theta)\,A_t,\ \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\,A_t\Big)\Big]$$

In classical RL, $A_t = \gamma V(s_{t+1}) + R_{t+1} - V(s_t)$, but in diffusion RLHF a uniform advantage $A = \frac{r - r_{\text{mean}}}{\text{std}(r)}$ is applied identically to every timestep.

4. Methodology

We address informative signals being drowned out at two levels: (1) timestep-level reweighting (Sec. 4.2) emphasizing denoising steps most responsible for final reward (Fig. 5, left); (2) trajectory-level hard-mining (Sec. 4.3) replaying past trajectories with high absolute advantage (Fig. 5, right).

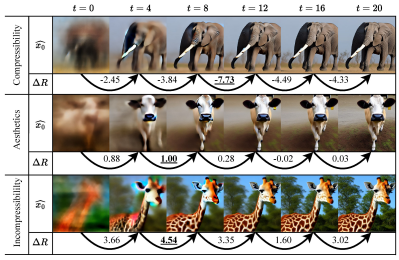


Figure 2. Change in reward of predicted x_0 across timestep intervals. Most image details are determined by step 12.

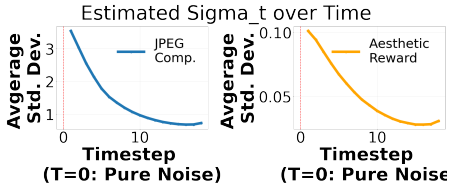


Figure 3. Stochastic estimation of $\sigma_t = \sqrt{\text{Var}(A_t)}$.

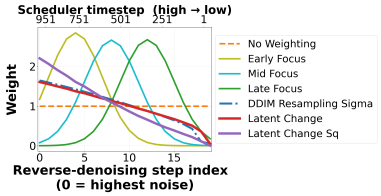


Figure 4. Various timestep weighting schemes, all normalized to mean 1.

4.1. Non-Uniform Credit Assignment

Standard diffusion RLHF computes $A = \frac{R - R_{\text{mean}}}{\text{std}(R)}$ and applies it uniformly:

$$\mathcal{L}_{\text{ddpo}} = \sum_{t=1}^{n} -A \cdot \log\left(P_\theta(x_{t-1}|x_t, c)\right)$$

This ignores how much each step contributes to the final reward.

4.1.1. Case Study.

Let $R(x_t) = R(\hat{x}_0(x_t))$ denote the reward at timestep t. For a small update $x_{t+1} = x_t + h_t$, first-order expansion gives $\Delta R_t \approx h_t^\top \nabla R(x_t)$, and telescoping yields $R(x_T) - R(x_0) \approx \sum_t \Delta R_t$. As shown in Fig. 2, ΔR_t varies significantly across timesteps, confirming unequal contributions. However, ΔR_t captures only immediate reward change, not long-term effects.

4.1.2. Advantage Variance Proportional Weighting.

We connect diffusion RLHF to PPO to derive principled timestep weights. In a diffusion trajectory $(s_0, \ldots, s_{20})$,

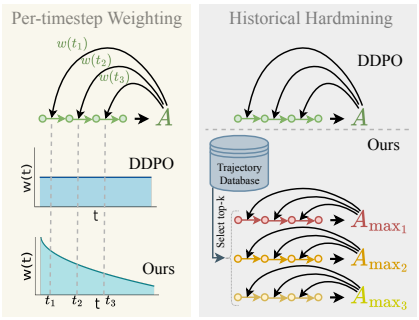


Figure 5. Our two-part method: per-timestep weighting emphasizes important denoising steps; historical hardmining replays high-advantage trajectories.

only s_{20} receives reward R_{final} and $\gamma = 1$, so TD-error advantages simplify to $A_t = V(s_{t+1}) - V(s_t)$, which telescope:

$$A_0 + A_1 + \ldots + A_{19} = V(s_{20}) - V(s_0) = R_{\text{final}} - V(s_0)$$

Approximating $V(s_0) \approx R_{\text{mean}}$ gives $\sum_i A_i \approx A_{\text{final}} \cdot \text{std}(R)$. Modeling each $A_t \sim \mathcal{N}(0, \sigma_t^2)$ and conditioning on this linear constraint (see appendix):

$$A_k \Big| \textstyle\sum_i A_i = A_{\text{final}} \cdot \text{std}(R) \;\sim\; \mathcal{N}\Big(\frac{\sigma_k^2 \,\text{std}(R)}{\sum_i \sigma_i^2} A_{\text{final}},\; \sigma_k^2 \Big(1 - \frac{\sigma_k^2}{\sum_i \sigma_i^2}\Big)\Big)$$

Thus $\mathbb{E}[A_k] = w(t) \cdot A_{\text{final}}$ where:

$$w(t) = \frac{\sigma_t^2 \cdot \text{std}(R)}{\sum_{i=0}^{n} \sigma_i^2}$$

Since σ_t depends only on timestep t, $w(t)$ applies universally across trajectories.

4.1.3. Empirical Measurement of $\text{Var}(A_t)$.

We estimate σ_t^2 by partially denoising to timestep t, branching into multiple trajectories, and computing final reward variance (Fig. 3). This confirms non-uniform weighting is needed and reveals a monotonically decreasing trend.

4.2. Strategy 1: Per-Timestep Weighting

While PPO timestep advantages can be theoretically derived, exact estimation during training is impractical. We experiment with multiple weighting schemes (Fig. 4): reverse diffusion standard deviation $\sqrt{\tilde{\beta}_t}$ (monotonically decreasing; similar to [10]), mean absolute latent change $|z_t - z_{t-1}|$, mean squared latent change $|z_t - z_{t-1}|^2$ (both dynamic and monotonic), and Gaussian schemes with early/middle/late focus.

Our reweighted loss replaces uniform advantage with $w(t)A_{\text{final}}$:

$$\mathcal{L}_{\text{reweighted}} = \sum_{t=1}^{n} -A_{\text{final}} \cdot w(t) \cdot \log\left(P_\theta(x_{t-1}|x_t, c)\right)$$

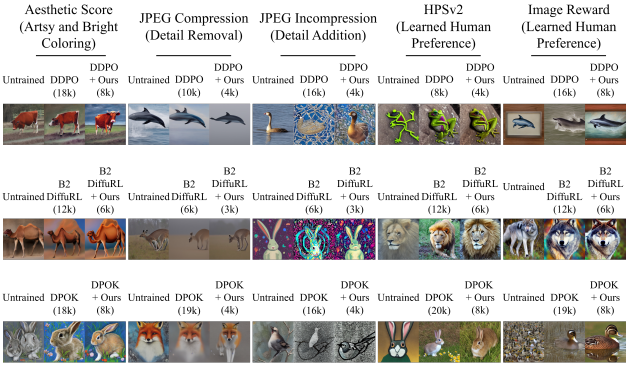


Figure 6. Qualitative comparisons: our method *vs*. default RLHF *vs*. no training. Parentheses show reward queries used. Same seed and noise within each triplet.

4.3. Strategy 2: Historical Hardmining

GRPO discards trajectories after each epoch, wasting informative samples. We introduce a replay buffer prioritizing trajectories with high $|A|$, as these contain timesteps that significantly changed the reward. We use two replay methods: (1) trajectory-level hardmining selecting top-k by $|A|$, and (2) random sampling. Only the last few epochs are kept to avoid out-of-distribution trajectories.

Algorithm 1 Overall Two Strategy Algorithm

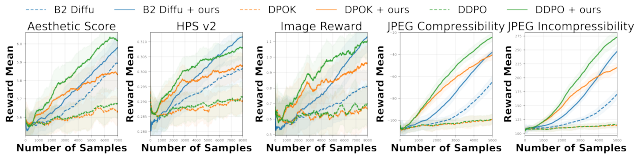
```
Initialize buffer D = {}, pretrained diffusion model π
for epoch = 1 to num_epochs do
    for i = 1 to n do
        Sample trajectory τ_i = (x_T, x_{T-1}, ..., x_0)
    end for
    for i = 1 to n do
        Compute weighted loss L_reweighted using Eq. (2) with trajectory τ_i
        θ = θ − ∇_θ L_reweighted
    end for
    if D is not empty then
        Retrieve top k samples from D
        Compute L_reweighted using all top k samples
        θ = θ − ∇_θ L_reweighted
    end if
    D ← D ∪ {τ_1, ..., τ_n}
    D ← Remove_Old_Entries(D)
end for
```

5. Experiments

We apply our method to DDPO [2], DPOK [8], and B2-DiffuRL [13] using the same animal prompts as [2], with LoRA [12] rank-4 on Stable Diffusion v1.5 [29]. Full hyperparameters are in the appendix.

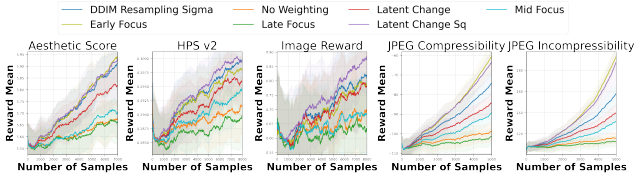
Reward Functions. We test on 5 rewards: JPEG compressibility (smooth textures), JPEG incompressibility (detailed textures), Aesthetic Score [31] (human aesthetic preferences), HPS v2 [40], and Image Reward [43] (both trained on aesthetic and prompt-adherence feedback).

Sample Efficiency. Figure 7a shows our modifications achieve higher reward than all baselines within the same budget – up to 6× efficiency gains across all rewards. All comparisons use identical hyperparameters between each

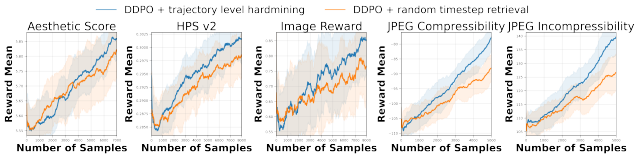


(a) Our augmentation trains significantly faster. Averaged over 3 runs, smoothed with 500-query running average.

(b) Ablation: each technique contributes; combining them is most effective. Averaged over 3 seeds.



(c) Weighting schemes. Latent change squared is best across all rewards.



(d) Hardmining *vs.* random replay. Hardmining is consistently more effective.

Figure 7. Experimental results. All plots use 3 seeds and a 500-query running average.

Table 1. Generalization To Novel Prompts at 4k Reward Queries.

	Aesthetic (↑)	Jpeg Comp (↑)	Jpeg Incomp (↑)	Image Reward (↑)	HPS v2 (↑)
Untrained	5.44	-118.75	118.758	0.48	0.284
DDPO	5.48	-112.79	124.48	0.53	0.2888
DDPO + Ours	**5.66**	**-44.25**	**233.28**	**0.71**	**0.3004**

Table 2. Prompt Adherence via CLIP Score (↑) at 4k Reward Queries.

	Aesthetic	Jpeg Comp	Jpeg Incomp	Image Reward	HPS v2
DDPO	0.289	**0.305**	0.299	0.308	**0.304**
DDPO + Ours	**0.311**	0.298	**0.310**	**0.312**	0.300

Figure 8. Training samples at matched reward levels. Green boxes show DDPO and our method reaching the same reward; we achieve it in less than half the steps. Later images show over-optimization, common to all RLHF methods.

baseline and its augmented version. For qualitative examples, see Figures 6 and 8. We also test generalization to novel prompts (generated by ChatGPT [23] for animals not in training). Table 1 shows our method generalizes better on the same query budget.

Ablations. Figure 7b shows all four combinations: neither, hardmining only, weighting only, and both. Each technique independently improves efficiency, and combining them is consistently best.

Timestep Weighting. Among Gaussian schemes, early focus outperforms middle, which outperforms late, suggesting monotonically decreasing weights are preferable. The best scheme is mean squared latent change $|z_t - z_{t-1}|^2$ (Fig. 7c), outperforming TempFlow's [10] DDIM standard deviation. We attribute this to prioritizing timesteps where the model makes the largest latent changes.

Historical Hardmining. Figure 7d shows hardmining consistently outperforms random replay across all rewards, validating that high-advantage trajectories are more informative.

CLIP Score and Generalization. We measure CLIP score [11] on 1k generated images (Table 2), finding no degradation of prompt adherence. Table 1 confirms our method generalizes well to novel prompts.

6. Conclusion

In this paper, we provide a simple but effective method to enhance the sample efficiency of diffusion RLHF. Easily adaptable to existing platforms, it can speed up training up to 6 times while retaining prompt-adherence and generalization to novel prompts. Our method departs from the common assumption that each timestep affects output reward equally, grounded in the intuition that certain timesteps and trajectories are more informative than others. Future directions could analyze how different schedulers and α-schedules affect training efficiency or extend the method to preference-based models such as D3PO.

References

[1] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay, 2018. 1, 2

[2] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning, 2024. 1, 2, 3, 8

[3] Miaomiao Cai, Simiao Li, Wei Li, Xudong Huang, Hanting Chen, Jie Hu, and Yunhe Wang. Dspo: Direct semantic preference optimization for real-world image super-resolution, 2025. 2

[4] Yekun Chai, Haoran Sun, Huang Fang, Shuohuan Wang, Yu Sun, and Hua Wu. Ma-rlhf: Reinforcement learning from human feedback with macro actions, 2025. 2

[5] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023. 2

[6] Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences, 2023. 1

[7] Kevin Clark, Paul Vicol, Kevin Swersky, and David J Fleet. Directly fine-tuning diffusion models on differentiable rewards, 2024. 2

[8] Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models, 2023. 1, 2, 3

[9] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018. 2

[10] Xiaoxuan He, Siming Fu, Yuke Zhao, Wanli Li, Jian Yang, Dacheng Yin, Fengyun Rao, and Bo Zhang. Tempflow-grpo: When timing matters for grpo in flow models. *arXiv preprint arXiv:2508.04324*, 2025. 2, 3, 4

[11] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning, 2022. 4

[12] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. 3

[13] Zijing Hu, Fengda Zhang, Long Chen, Kun Kuang, Jiahui Li, Kaifeng Gao, Jun Xiao, Xin Wang, and Wenwu Zhu. Towards better alignment: Training diffusion models with reinforcement learning against sparse rewards, 2025. 1, 2, 3

[14] Qihan Huang, Weilong Dai, Jinlong Liu, Wanggui He, Hao Jiang, Mingli Song, and Jie Song. Patchdpo: Patch-level dpo for finetuning-free personalized image generation, 2025. 2

[15] Shuaiyi Huang, Mara Levy, Anubhav Gupta, Daniel Ekpo, Ruijie Zheng, and Abhinav Shrivastava. Trend: Tri-teaching for robust preference-based reinforcement learning with demonstrations, 2025. 2

[16] Zhiwei Jia, Yuesong Nan, Huixi Zhao, and Gengdai Liu. Reward fine-tuning two-step diffusion models via learning differentiable latent-space surrogate reward, 2025. 2

[17] Xingxing Liang, Yang Ma, Yanghe Feng, and Zhong Liu. Ptr-ppo: Proximal policy optimization with prioritized trajectory replay, 2021. 2

[18] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015. 2

[19] Cong Lu, Philip J. Ball, Yee Whye Teh, and Jack Parker-Holder. Synthetic experience replay, 2023. 1, 2

[20] Grace Luo, Jonathan Granskog, Aleksander Holynski, and Trevor Darrell. Dual-process image generation, 2025. 2

[21] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013. 2

[22] Soumik Mukhopadhyay, Matthew Gwilliam, Yosuke Yamaguchi, Vatsal Agarwal, Namitha Padmanabhan, Archana Swaminathan, Tianyi Zhou, Jun Ohya, and Abhinav Shrivastava. Do text-free diffusion models learn discriminative visual representations?, 2024. 1

[23] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing

Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2024. 4

[24] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, et al. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022. 2

[25] Benjamin Pikus, Pratyush Ranjan Tiwari, and Burton Ye. Hard examples are all you need: Maximizing grpo post-training under annotation budgets, 2025. 2

[26] Mihir Prabhudesai, Russell Mendonca, Zheyang Qin, Katerina Fragkiadaki, and Deepak Pathak. Video diffusion alignment via reward gradients, 2024. 2

[27] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2024. 2

[28] Allen Z. Ren, Justin Lidard, Lars L. Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization, 2024. 2

[29] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[30] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay, 2016. 1, 2

[31] Christoph Schuhmann and Romain Beaumont. Laion-aesthetics. *LAION. AI*, 2022. 3

[32] John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization, 2017. 2

[33] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. 2

[34] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan

Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. 2

[35] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining, 2016. 1

[36] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics, 2015. 1

[37] R.S. Sutton and A.G. Barto. Reinforcement learning: An introduction. *IEEE Transactions on Neural Networks*, 9(5):1054–1054, 1998. 2

[38] Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization, 2023. 2

[39] Yufei Wang, Zhanyi Sun, Jesse Zhang, Zhou Xian, Erdem Biyik, David Held, and Zackory Erickson. Rl-vlm-f: Reinforcement learning from vision language foundation model feedback, 2024. 2

[40] 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. 3

[41] Amber Xie, Rahul Chand, Dorsa Sadigh, and Joey Hejna. Data retrieval with importance weights for few-shot imitation learning, 2025. 2

[42] Xiaoying Xing, Avinab Saha, Junfeng He, Susan Hao, Paul Vicol, Moonkyung Ryu, Gang Li, Sahil Singla, Sarah Young, Yinxiao Li, Feng Yang, and Deepak Ramachandran. Focus-n-fix: Region-aware fine-tuning for text-to-image generation, 2025. 2

[43] 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*, 36:15903–15935, 2023. 3

[44] Brian Yang, Huangyuan Su, Nikolaos Gkanatsios, Tsung-Wei Ke, Ayush Jain, Jeff Schneider, and Katerina Fragkiadaki. Diffusion-es: Gradient-free planning with diffusion for autonomous driving and zero-shot instruction following, 2024. 2

[45] Kai Yang, Jian Tao, Jiafei Lyu, Chunjiang Ge, Jiaxin Chen, Qimai Li, Weihan Shen, Xiaolong Zhu, and Xiu Li. Using human feedback to fine-tune diffusion models without any reward model, 2024. 1, 2, 8

[46] Xiaomeng Yang, Zhiyu Tan, Junyan Wang, Zhijian Zhou, and Hao Li. Sdpo: Importance-sampled direct preference optimization for stable diffusion training, 2025. 2

[47] Rui Zhao and Volker Tresp. Energy-based hindsight experience prioritization, 2020. 2

A. Appendix

A.1. Hyperparameters For Baselines

We chose to use the same hyperparameters for baselines as those chosen for all baselines in the D3PO [45] paper. The only difference is that we chose a smaller number samples per epoch as we found that sampling more often led to better performance. We also changed to AdamW over Adam, as consistent with most diffusion RLHF methods. Between baseline and baseline + our method, we kept all hyperparameters the same. All results were obtained with 4 RTX A5000 GPUs or 4 RTX A4000 GPUs. See a comprehensive list of hyperparameters in Tables 3 and 4.

A.2. Prompt List

We used the same training prompt list as in [2]. The list of animals is listed below:

cat dog horse monkey rabbit zebra spider bird
sheep deer cow goat lion tiger bear raccoon fox
wolf lizard beetle ant butterfly fish shark whale
dolphin squirrel mouse rat snake turtle frog
chicken duck goose bee pig turkey fly llama
camel bat gorilla hedgehog kangaroo

For generalization experiments, we used the following list of animals generated by ChatGPT:

elephant giraffe hippopotamus rhinoceros leopard
cheetah hyena bison moose elk reindeer antelope
armadillo sloth otter beaver badger lynx bobcat
cougar jaguar capybara porcupine platypus echidna
koala wallaby wombat manatee walrus seal
narwhal orca penguin albatross flamingo peacock
owl eagle hawk parrot crocodile alligator
chameleon salamander

A.3. Comparisons with Hardmining

When using hardmining, even though we use lesser number of reward calls, we use more backpropagation through the network. Hence, an obvious question might be whether using hardmining helps just because there are more samples whose gradients are being backpropagrated or is hardmining helping because of the selection of more important trajectories. In this section, we compare hardmining with two more methods. First, we compare to simply increasing the learning rate proportional to how many more gradient steps we take. Second, we compare to randomly replaying entries from the current epoch. These two settings correspond to using faster learning rate and more gradient calls respectively, both of which are alternatives to hardmining. Note that this is different from Figure. 7d where we compared to random retrieval from previous epochs as opposed to only the current epoch. As seen in Figure 9, our hardmining based on the absolute advantages does better than both increasing the learning rate and repeating samples from the current epoch, showing that hardmining helps not just because of increased samples.

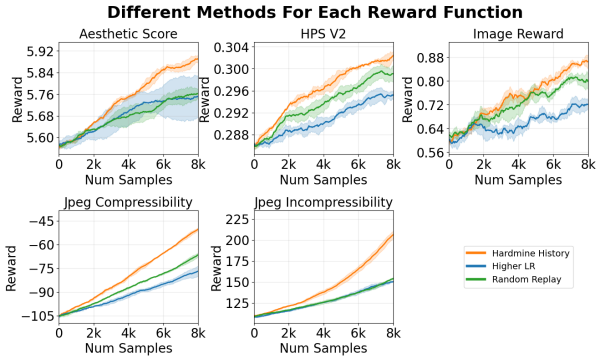


Figure 9. Comparison of Hardmining to increasing the LR and to randomly replaying samples from the current epoch. All results are averaged over 3 seeds.

A.4. Proof That A_k has mean $\frac{\sigma_k^2 \cdot \text{std}(R)}{\sum_{i=1}^{n} \sigma_i^2} A_{\text{final}}$

We want to find the probability distribution of A_k where we assume that each A_i has mean 0 and standard deviation σ_i for $i \in \{0, 1, 2, ..., n\}$. Furthermore, we have the condition that for a given constant C,

$$A_0 + A_1 + A_2 + ... + A_n = C$$

For simplicity of notation, assume that $k = 0$. The proof can easily generalize to other values of k.

Let $B = A_1 + A_2 + ... + A_n$. The mean of B is the sum of the means of $A_1 + A_2, ...A_n$ which is 0. The variance of B is the sum of the variances of $A_1 + A_2, ...A_n$. This means that $\sigma_B^2 = \sigma_1^2 + \sigma_2^2 + ... + \sigma_n^2$. Finally, we have that from the summation of Gaussian distributions is still a Gaussian distribution. Thus,

$$B = A_1 + A_2 + A_3 + ... + A_n \sim \mathcal{N}\left(0, \sigma_B^2\right)$$

We can rewrite our question as finding the distribution $A_0 \mid A_0 + B = C$. We have that the for a given value a such that $A_0 = a$,

$$
\begin{aligned}
&P(A_0 = a \mid A_0 + ... + A_n = C) \\
&= P(A_0 = a \mid A_0 + B = C) \\
&= P(A_0 = a \;\&\; A_0 + B = C)/P(A_0 + B = C) \\
&= P(A_0 = a \;\&\; B = C - a)/P(A_0 + B = C) \\
&\propto P(A_0 = a \;\&\; B = C - a) \\
&= P(A_0 = a)P(B = C - a) \\
&= \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp\left(-\frac{a^2}{2\sigma_0^2}\right) \cdot \frac{1}{\sqrt{2\pi\sigma_B^2}} \exp\left(-\frac{(C-a)^2}{2\sigma_B^2}\right) \\
&= \frac{1}{2\pi\sigma_0\sigma_B} \exp\left(-\frac{a^2}{2\sigma_0^2} - \frac{(C-a)^2}{2\sigma_B^2}\right) \\
&\propto \exp\left(-\frac{a^2}{2\sigma_0^2} - \frac{(C-a)^2}{2\sigma_B^2}\right) \\
&= \exp\left(-\frac{a^2}{2\sigma_0^2} - \frac{(C-a)^2}{2\sigma_B^2}\right) \\
&= \exp\left(\frac{-a^2\sigma_B^2 - (C-a)^2\sigma_0^2}{2\sigma_0^2\sigma_B^2}\right) \\
&= \exp\left(\frac{-a^2(\sigma_0^2 + \sigma_B^2) + 2C\sigma_0^2 a - C^2\sigma_0^2}{2\sigma_0^2\sigma_B^2}\right) \\
&= \exp\left(\frac{-a^2(\sigma_0^2 + \sigma_B^2) + 2C\sigma_0^2 a - C^2\sigma_0^2}{2\sigma_0^2\sigma_B^2}\right) \\
&= \exp\left(\frac{-(\sigma_0^2 + \sigma_B^2)\left(a - C\frac{\sigma_0^2}{\sigma_0^2+\sigma_B^2}\right)^2}{2\sigma_0^2\sigma_B^2}\right) \times \exp\left(\frac{C^2\frac{\sigma_0^4}{\sigma_0^2+\sigma_B^2} - C^2\sigma_0^2}{2\sigma_0^2\sigma_B^2}\right) \\
&= \underbrace{\exp\left(\frac{C^2\frac{\sigma_0^4}{\sigma_0^2+\sigma_B^2} - C^2\sigma_0^2}{2\sigma_0^2\sigma_B^2}\right)}_{\text{constant}} \times \exp\left(\frac{-(\sigma_0^2 + \sigma_B^2)\left(a - C\frac{\sigma_0^2}{\sigma_0^2+\sigma_B^2}\right)^2}{2\sigma_0^2\sigma_B^2}\right) \\
&\propto \exp\left(\frac{-(\sigma_0^2 + \sigma_B^2)\left(a - C\frac{\sigma_0^2}{\sigma_0^2+\sigma_B^2}\right)^2}{2\sigma_0^2\sigma_B^2}\right) \\
&= \exp\left(\frac{-\left(a - C\frac{\sigma_0^2}{\sigma_0^2+\sigma_B^2}\right)^2}{2 \cdot \frac{\sigma_0^2\sigma_B^2}{\sigma_0^2+\sigma_B^2}}\right)
\end{aligned}
$$

We have that the end result is a gaussian with mean

$$\mu = C\frac{\sigma_0^2}{\sigma_0^2 + \sigma_B^2} = C\frac{\sigma_0^2}{\sigma_0^2 + \sigma_1^2 + ... + \sigma_n^2}$$

and variance with value

$$\frac{\sigma_0^2\sigma_B^2}{\sigma_0^2 + \sigma_B^2} = \sigma_0^2\left(1 - \frac{\sigma_0^2}{\sum_{i=0}^n \sigma_i^2}\right)$$

Letting $C = \text{std}(R)A_{\text{final}}$, we get that the distribution of A_0 under the assumption that $A_0 + A_1 + ... + A_n = \text{std}(R)A_{\text{final}}$ has mean $\mu = \frac{\sigma_0^2}{\sigma_0^2+\sigma_1^2+...+\sigma_n^2}\text{std}(R)A_{\text{final}}$

Table 3. Hyperparameters of All Experiments

Name	Description	Value
lr	learning rate	3e-5
$optimizer$	type of optimizer	AdamW
ξ	weight decay of optimizer	1e-4
ϵ	gradient clip norm	1.0
β_1	β_1 of Adam	0.9
β_2	β_2 of Adam	0.999
T	total timesteps of inference	20
n	samples per GPU per epoch	5
η	eta parameter for the DDIM sampler	1.0
G	effective batch size (collectively over all gpus)	4
w	classifier-free guidance weight	5.0
num_gpu	number of GPUs	4

Table 4. Additional Hyperparameters for Our Method

Name	Description	Value
k	top k samples to retrieve from buffer per GPU	4
$hist\_size$	number of previous epochs to retrieve hardmined samples	3
$w(t)$	per-timestep weights	Latent Change Squared

A.5. More Image Samples

In figures 10, 11, 12, 13, and 14, we give a randomly selected sample of DDPO vs DDPO with our method. We use 30 random seeds in the grid for the 30 images and denoise using finetuned weights.

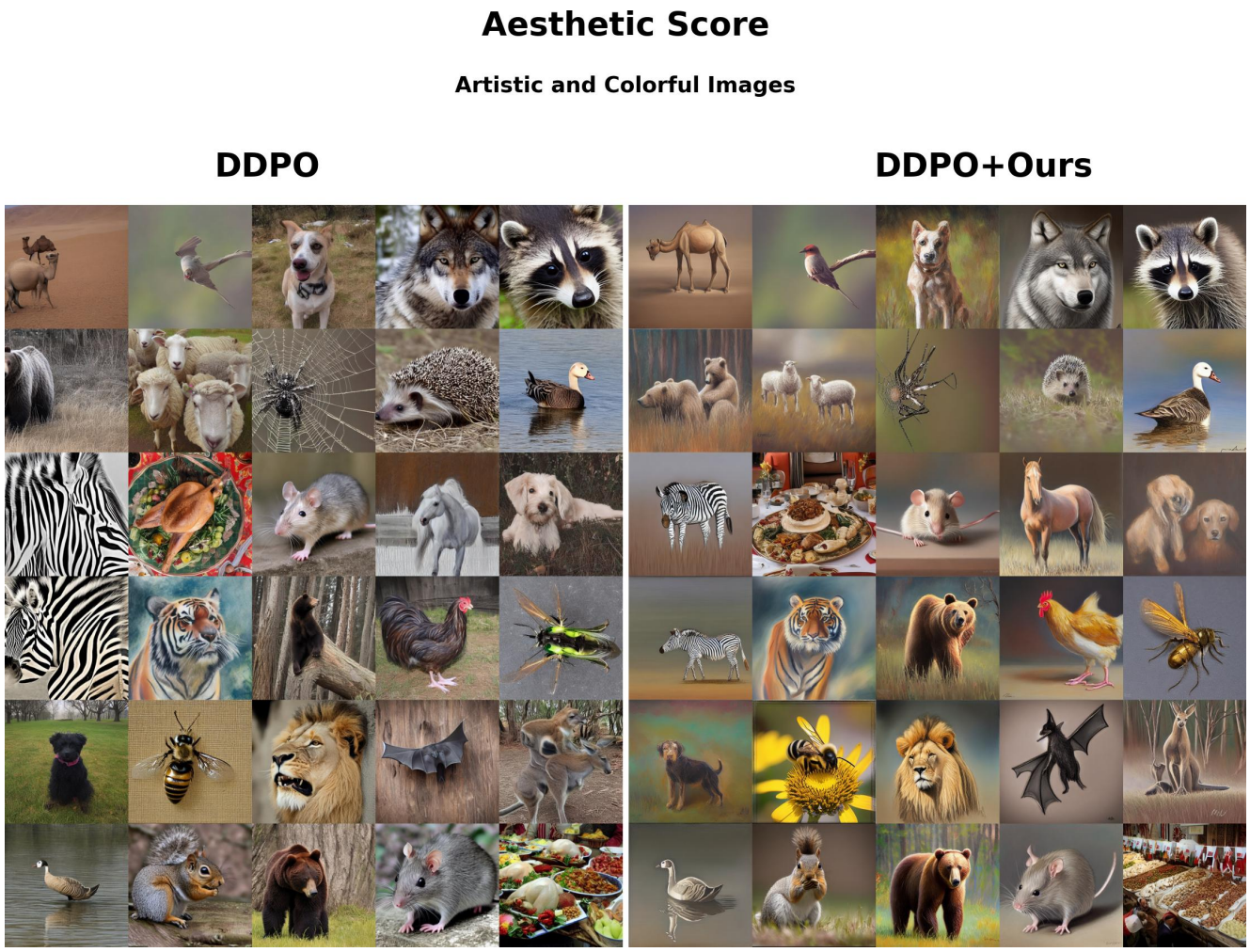


Figure 10. The following is DDPO (left) vs DDPO+ours (right) after 6k reward images on Aesthetic Score. Aesthetic score mostly values bright colors, artistic value, and point of view.

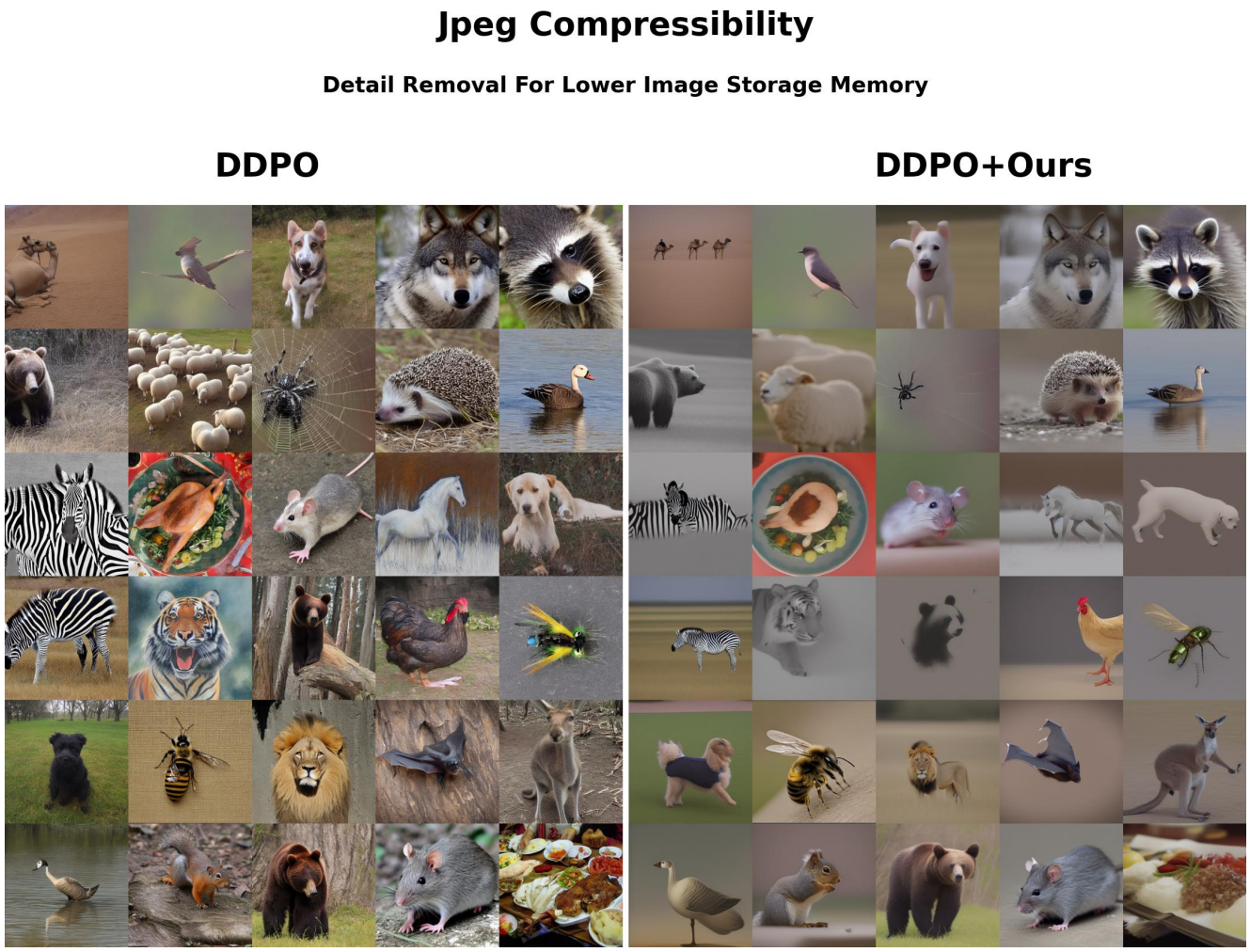


Figure 11. The above is DDPO (left) vs DDPO+ours (right) after 3k reward queries on Jpeg Compression. Note that for this reward, we want the least amount of detail as possible to reduce the Jpeg memory size. Thus, the highest reward images are those with the least detail.

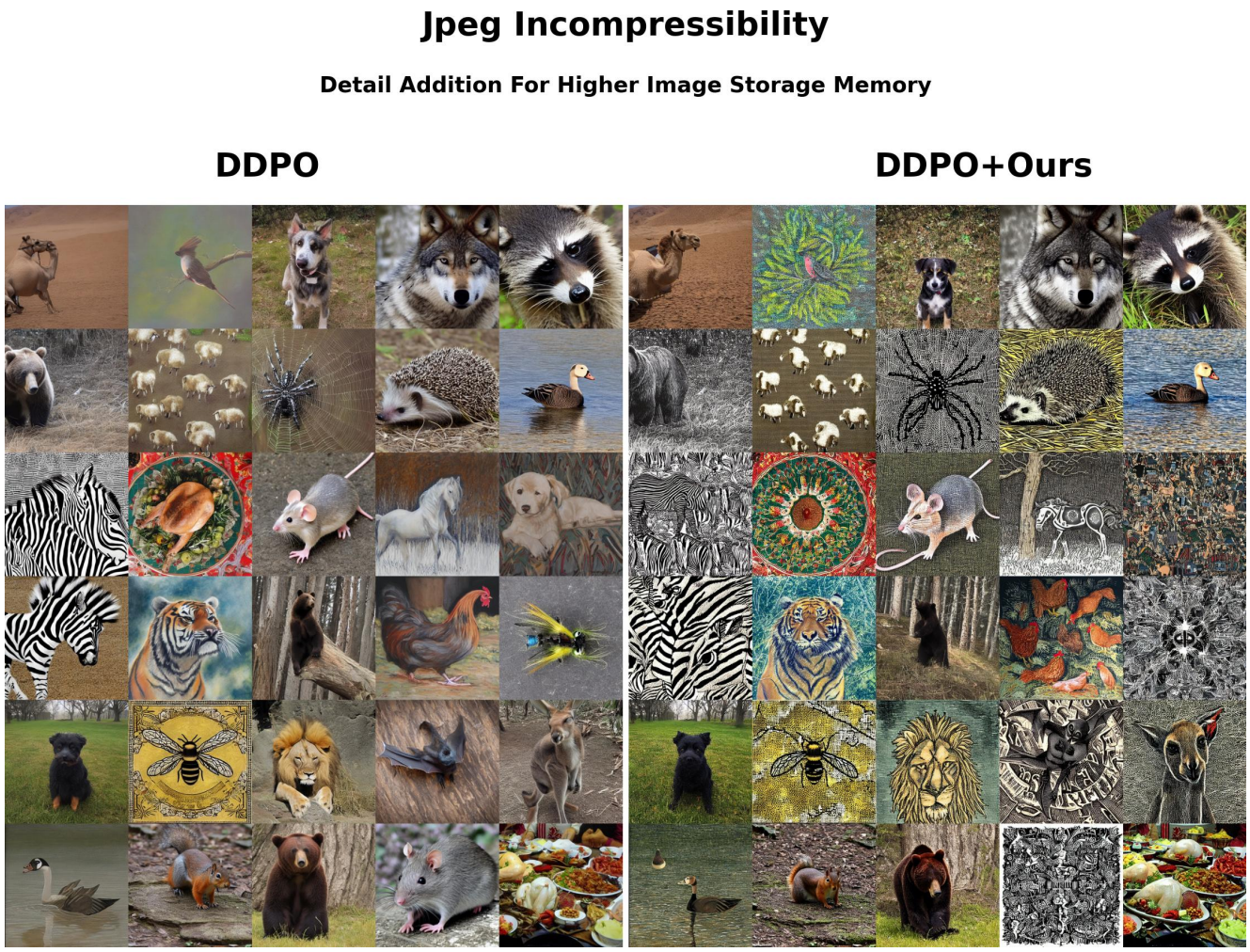


Figure 12. The above is DDPO (left) vs DDPO+ours (right) after 2k reward images on Jpeg Incompression. Note that we want the most amount of detail as possible to increase the Jpeg memory size. Thus, the highest reward images are those with the most detail.

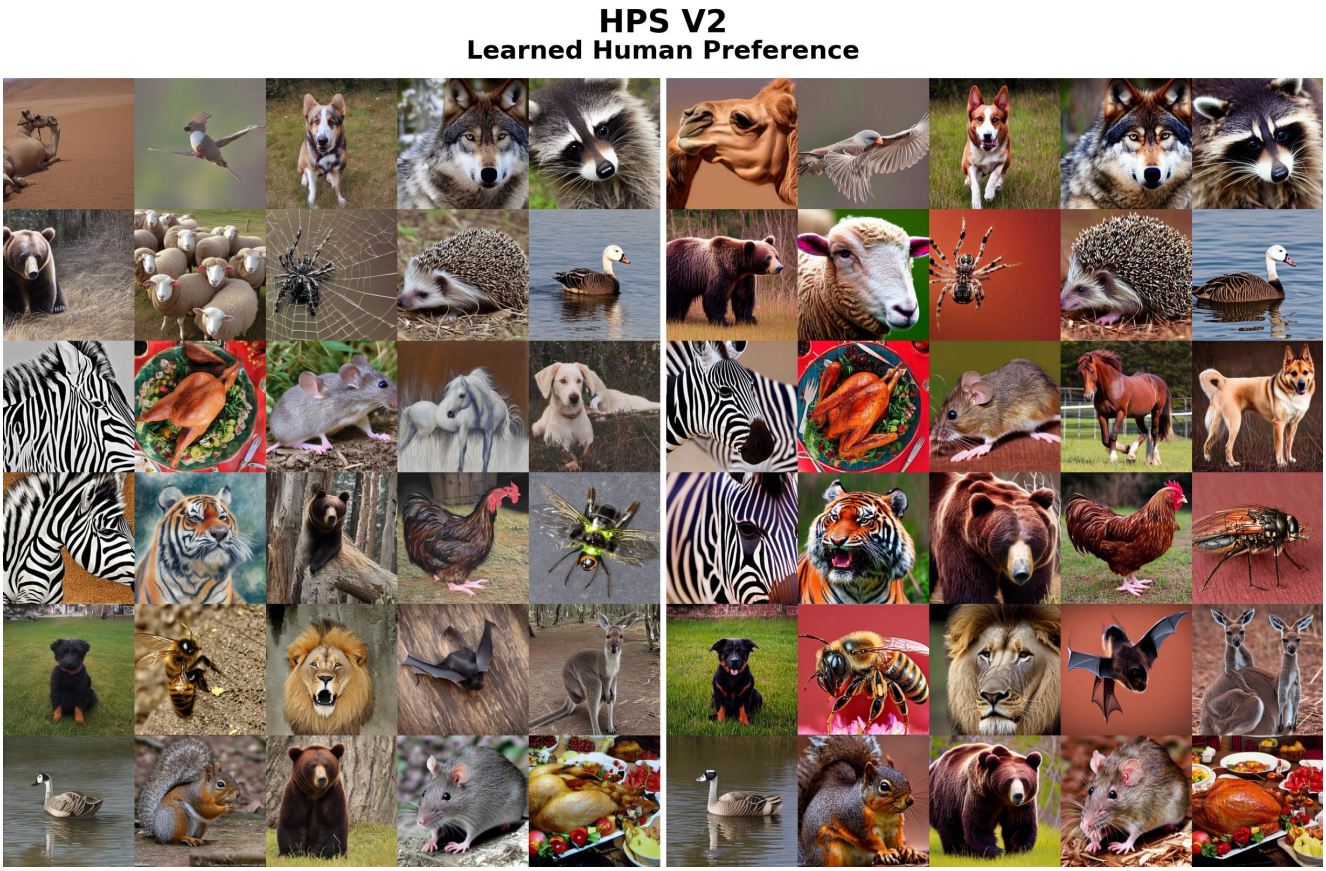


Figure 13. The above is DDPO (left) vs DDPO+ours (right) after 7k reward images on HPS V2. Note that HPS V2 is trained to mimic human preference.

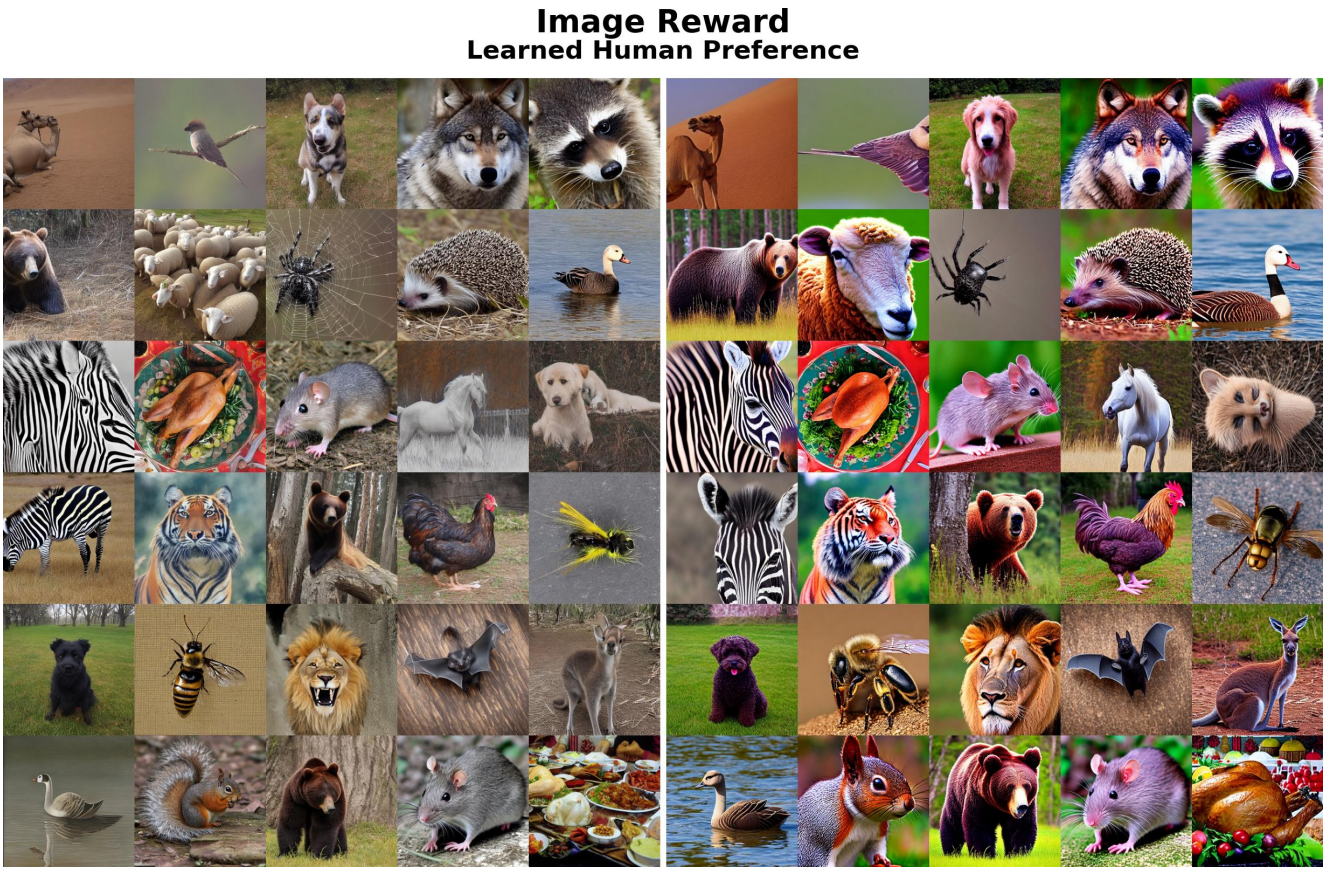


Figure 14. The above is DDPO (left) vs DDPO+ours (right) after 9k reward images on Image Reward. Note that Image Reward is trained to mimic human preferences on images.