Learning to Sample Effective and Diverse Prompts for Text-to-Image Generation

Supplementary Material

Our code is publicly available here.

A. Illustrations of RL and GFlowNets

In this section, we summarize the key difference between Reinforcement Learning (RL) and GFlowNets in more detail. GFlowNets [3] is designed to learn a stochastic policy that generates samples proportional to their rewards (i.e., $p(x) \propto R(x)$), while RL aims to learn a policy that maximizes the reward function as illustrated in Figure 8. Learning a policy that samples proportional to the reward function leads to capturing multi-modal distribution and discovering high-quality and diverse candidates, which is particularly beneficial when the reward proxy is inaccurate [3], and their connections have been studied in [8, 14, 48], which shares equivalences and similarities with entropy-regularized RL [30] in tree-structured sequence generation and directed acyclic graph problems [16, 22]. In prompt adaptation tasks, conventional RL-based approaches [13] based on PPO [43] that naively maximize a reward function can lead to reward overoptimization and hinder generalizability to different text-to-image diffusion models, while it is more desirable to generate not only effective and but also diverse prompts.

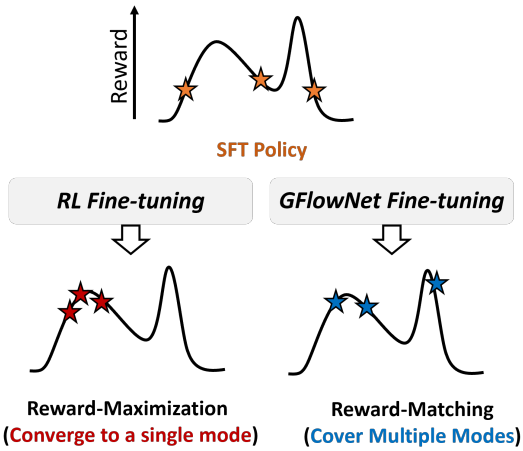


Figure 8. Comparison of the learning objective of reward-maximizing RL and reward-matching GFlowNets.

B. Experiment Setting Details

In this section, we present details of experiment settings.

B.1. Data Preparation

We strictly follow the procedure of Promptist [13] to prepare training and evaluation datasets. Additionally, we introduce a challenging dataset using ChatGPT [32] interface to generate brief prompts that describe images. Specifically, we use the following prompt to query ChatGPT for short image descriptions:

- `Generate N sentences describing photos /pictures/images with length around 5.`

Below are a few example prompts generated by ChatGPT:

- `A bird sitting on a branch.`
- `A tree under a sky.`
- `A car drives on a road.`
- `A train moves through the city.`
- `A boat sails on a lake.`

B.2. Baselines

In this section, we provide more details on the baselines used for our experiments.

Supervised Fine-Tuning. We fine-tune the pretrained GPT-2 policy model with supervised learning on a set of prompt pairs of original user inputs and manually engineered prompts provided by Promptist [13]. As a default, we use the pretrained weights of SFT publicly available[1].

Promptist. We train the policy with a PPO-based approach where the policy is initialized with the supervised fine-tuned model. As a default, we use the pretrained weights of Promptist publicly available to ensure a fair comparison. To evaluate the generalizability of different reward functions, we train the policy with the same hyperparameter configurations.

Rule-Based. Based on the observation that Promptist mostly generates similar postfixes for optimization, we build a rule-based method that appends the most frequently used postfixes in Promptist to the initial prompt. Below are a few example postfixes we used for evaluation:

- `intricate, elegant, highly detailed, digital painting, artstation, concept art, sharp focus, illustration, by justin gerard and artgerm, 8 k.`
- `by greg rutkowski, digital art, realistic painting, fantasy, very detailed, trending on artstation.`
- `highly detailed, digital painting, artstation, concept art, sharp focus, illustration, art by greg rutkowski and alphonse mucha.`

[1] https://github.com/microsoft/LMOps/tree/main/promptist

GFlowNets. We train a vanilla GFlowNets policy with TB [28] objective. As our task is a conditional generation task, a naive implementation of TB should directly estimate a conditional partition function, $Z_\theta(\mathbf{x})$, which makes learning highly unstable [19]. For pratical implementation, we use VarGrad [39] objective to fine-tune the policy, which is widely used for reducing variance in GFlowNets literature [49, 56].

For each initial prompt $\mathbf{x}$ sampled in the minibatch, we generate $k = 16$ on-policy samples $\mathbf{y}^{(1)}, \cdots, \mathbf{y}^{(k)}$ with the forward policy. Each sample can be used to implicitly estimate $\log Z(\mathbf{x})$:

$$\log \hat{Z}(\mathbf{x})^{(i)} = R(\mathbf{x}, \mathbf{y}^{(i)}) - \sum_{t=1}^{T} \log P_F(y_t^{(i)} | y_{0:t-1}^{(i)}, \mathbf{x}; \theta)$$

Then we minimize the sample variance across the minibatch as follows:

$$\mathcal{L}(\mathbf{x}; \theta) = \frac{1}{k} \sum_{i=1}^{k} \left(\log \hat{Z}(\mathbf{x})^{(i)} - \frac{1}{k} \sum_{j=1}^{k} \log \hat{Z}(\mathbf{x})^{(j)} \right)^2$$

DPO-Diff. Wang et al. [50] proposed a discrete prompt optimization for diffusion models (DPO-Diff), which is a gradient-based optimization method for discovering effective negative prompts to generate user-aligned images. It first generates a compact subspace comprised of only the most relevant words to user input with ChatGPT API, then uses shortcut text gradients to efficiently compute the text gradient for optimization. As the original reward function of DPO-Diff is a spherical clip loss, we replace the reward function the same as Eq. (1). As we consider a setting where the reward function is a black-box function, we use the evolutionary search module suggested by the paper for a fair comparison. Please refer to the paper for more details.

B.3. Training and Evaluation

For training, we initialize the policy with the SFT policy before the GFlowNet fine-tuning. To parametrize the flow function, we use a separate neural network that takes the last hidden embedding of the prompts as input and outputs a scalar value. We conducted all experiments with 4 NVIDIA A100 GPUs, and the training took approximately 24 hours.

For evaluation, we study two metrics: reward and diversity. To compute the reward, we generate $N = 16$ prompts for each initial prompt via beam search with a length penalty of 1.0. Then, we generate three images per prompt to compute the reward. We aggregate the max score among N prompts and compute the average across initial prompts.

$$\text{Reward}(\mathcal{D}_{\text{eval}}) := \frac{1}{|\mathcal{D}_{\text{eval}}|} \sum_{\mathbf{x} \in \mathcal{D}_{\text{eval}}} \max_{\mathbf{y} \sim p_\theta(\cdot|\mathbf{x})} (r(\mathbf{x}, \mathbf{y}))$$

To compute diversity, we embed the generated prompts using MiniLMv2 [51] encoder, compute the average pairwise cosine distance between embeddings of the prompts, and compute the average across initial prompts. For all evaluations, we conduct experiments with four random seeds and report the mean and standard deviation.

The input format for both training and evaluation is `[Initial Prompt] Rephrase:`, following Promptist [13]. The hyperparameters we used for modeling and training are listed in Table 4. We conduct several ablations studies on the impact of various hyperparameters in Appendix D.

Table 4. Hyperparameters for Training PAG.

Parameters	Values
Batch size	64
Buffer size	5000
Optimizer	Adam
Training Steps	1×10^4
Learning Rate (γ)	1×10^{-5} (Policy), 1×10^{-4} (Flow)
Temperature (β)	0.05
Reset Period (M)	2000

B.4. Robustness across Different Reward Functions

To evaluate the robustness of our framework in terms of different reward functions, we use two widely-used reward functions in diffusion alignment: ImageReward [54] and HPSv2 [53]. We provide a detailed description of each function below.

ImageReward ImageReward is a general-purpose text-to-image preference reward model trained with pairs of prompts and images. To compute the score, ImageReward extracts image and text embeddings using BLIP [23] encoder and combines them with cross attention, and uses MLP to generate a scale value for preference comparison.

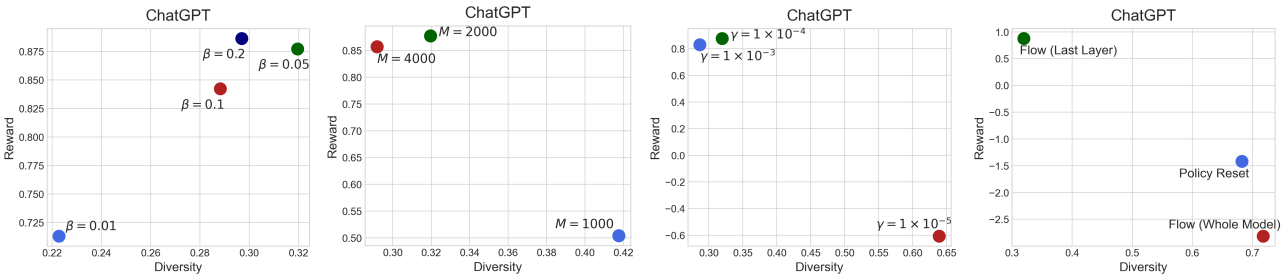
HPSv2 HPSv2 is a scoring model that accurately predicts human preferences on generated images. To accurately predict the score, it fine-tunes the CLIP [36] with the HPDv2 dataset, a large-scale dataset that captures human preferences on images from various sources.

B.5. Transferability to Different Text-to-Image Diffusion Models

To validate the transferability of our framework to different text-to-image diffusion models, we prepare several widely-used text-to-image diffusion models: SD v1.5 [40], SSD-1B [12], SDXL-Turbo [42], and SD3 [10]. As SDXL-Turbo and SD3 models do not align with DPM solver [26], we use

Table 5. Performance of prompt generated by each method. Aes score indicates aesthetic quality improvement compared to the image generated with the original input. Experiments are conducted with four random seeds, and mean and standard deviation are reported. **Bold** represent the best entry.

Method	Lexica		DiffusionDB		COCO		ChatGPT	
	Reward	Diversity	Reward	Diversity	Reward	Diversity	Reward	Diversity
Initial Prompt	-0.16 ± 0.03	-	-0.22 ± 0.02	-	-0.35 ± 0.01	-	-0.42 ± 0.03	-
SFT	0.64 ± 0.02	0.13 ± 0.00	0.58 ± 0.01	0.13 ± 0.00	0.54 ± 0.07	0.11 ± 0.02	0.76 ± 0.03	0.19 ± 0.00
Promptist	0.76 ± 0.02	0.09 ± 0.00	0.76 ± 0.03	0.10 ± 0.00	0.65 ± 0.02	0.07 ± 0.00	0.76 ± 0.03	0.12 ± 0.00
Rule-Based	0.72 ± 0.02	0.26 ± 0.00	0.69 ± 0.02	0.15 ± 0.00	0.75 ± 0.01	0.12 ± 0.00	0.82 ± 0.03	0.17 ± 0.00
GFlowNets	0.96 ± 0.01	0.13 ± 0.00	0.85 ± 0.03	0.13 ± 0.00	0.73 ± 0.02	0.09 ± 0.00	0.63 ± 0.03	0.10 ± 0.00
DPO-Diff	0.13 ± 0.02	-	0.28 ± 0.06	-	-0.03 ± 0.06	-	-0.17 ± 0.03	-
PAG (Ours)	**0.99 ± 0.05**	**0.32 ± 0.00**	**0.91 ± 0.04**	**0.33 ± 0.00**	**0.88 ± 0.02**	**0.32 ± 0.00**	**0.88 ± 0.04**	**0.32 ± 0.00**



(a) Analysis on β (b) Analysis on M (c) Analysis on γ for flow function (d) Analysis on reset strategies

Figure 9. Extended ablation studies on various components of PAG.

a standard generation pipeline, which uses a PNDM scheduler [24] with 20 inference steps. Furthermore, as SDXL-Turbo does not use the guidance scale, we set the guidance scale to 0. for SDXL-Turbo, and 7.5 (default) for others.

C. Extended Main Results

In this section, we provide additional discussion and analysis of our main experimental results.

C.1. Main Results

We summarize the main experiment results in Table 5 including comparisons with DPO-Diff [50], a recent relevant work. Note that as DPO-Diff tries to optimize negative prompts and the initial prompt is always the same, it is meaningless to compute diversity between prompts.

C.2. Discussion

As shown in the table, we observe that DPO-Diff shows modest improvements in terms of the reward compared to other baselines. We find that while DPO-Diff can effectively improve the CLIP scores, it shows limited capability in improving the aesthetic score.

D. Extended Ablation Studies

In this section, we include additional ablation studies that complement our main text due to space limitations.

D.1. Analysis on β

First, we analyze the effect of inverse temperature β in Eq. (8), which controls the balance between the task reward term $r(\mathbf{x}, \mathbf{y})$ and the reference LM likelihood term $p_{\text{ref}}(\mathbf{y}|\mathbf{x})$. As a default setting, we set $\beta = 0.05$.

To analyze the effect of β, we fine-tune the GFlowNet policy with different β values: $\{0.01, 0.05, 0.1, 0.2\}$. As shown in the Figure 9a, there are no significant differences in terms of the performance with different β values while using too small β, which leads to the policy focus on a high-reward region, suffers from mode collapse similar to naive GFlowNet and exhibits poor performance.

D.2. Analysis on M

We also conduct experiments by varying the flow function reset period (M) to analyze the effect of flow reactivation. If we reset the flow function too frequently, it is hard to capture high-rewarding multi-modal distribution, while rarely applying reset leads to the mode collapse issue. As a default setting, we use $M = 2000$, implying that we reactivate the flow function four times over the whole training procedure.

To analyze the effect of M, we fine-tune the GFlowNet policy with different M values: $\{1000, 2000, 4000\}$. As shown in the Figure 9b, we find that too frequent reactivation ($M = 1000$) does not capture high-reward regions and suffers from a significant drop in performance. While there is no big difference between $M = 2000$ and $M = 4000$, we

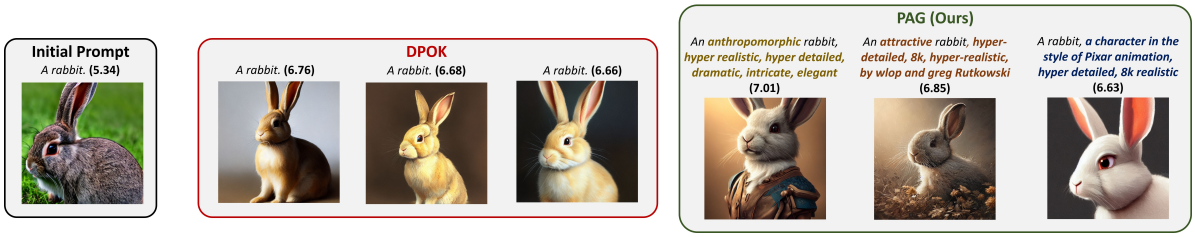


Figure 10. Comparison with DPOK and PAG. We report the aesthetic score of images in bold.

empirically find that set $M = 2000$ achieves the best performance in terms of both reward and diversity. This empirical finding is also aligned with the other papers, which utilize a reset strategy: Nikishin et al. [31] also reset the last layer of the neural networks four times over the course of training.

D.3. Analysis on learning rate of flow function

Based on the observation that most actor-critic RL methods use slightly higher learning rates for the critic than the actor [43], we use a higher learning rate (1×10^{-4}) for the flow function training than the learning rate of the policy (1×10^{-5}). Using a higher learning rate is also crucial for quickly recovering from the flow reactivation.

To analyze the effect of learning rate (γ) for the flow function, we fine-tune the GFlowNet policy with different γ values: $\{1 \times 10^{-3}, 1 \times 10^{-4}, 1 \times 10^{-5}\}$. As shown in the Figure 9c, we find that using the same learning rate for the policy and the flow function leads to poor performance as expected. While there is no big difference between $\gamma = 1 \times 10^{-3}$ and $\gamma = 1 \times 10^{-4}$, we empirically find that set $\gamma = 1 \times 10^{-4}$ achieves the best performance in terms of both reward and diversity.

D.4. Analysis on flow reactivation scheme

To prevent a significant drop in performance and unstable training, we employ a targeted reset strategy that resets only the last layer of the flow function. To analyze the effect of our strategy, we conduct experiments with two additional reset strategies: (1) reset the whole layer of the flow function and (2) reset the policy. For resetting the policy, we randomly reset 0.01% of neurons in the policy parameters.

Figure 9d shows the performance of various reset strategies. As depicted in the figure, resetting the whole layer of the flow function does not recover policy towards high-scoring regions. Resetting the policy parameters also exhibits poor performance, as the policy directly affects the acquisition of online samples.

D.5. Analysis on Diversity of Images

We also compute the diversity between the final images sampled from text-to-image diffusion models conditioned on prompts generated by different methods. To compute diversity, we compute the average pairwise distance between feature vectors extracted by the pre-trained InceptionV3 model [47]. As shown in Table 6, PAG exhibits high diversity on images compared to baselines.

Table 6. Diversity evaluation on images.

Method	COCO		ChatGPT	
	Reward	Diversity	Reward	Diversity
SFT	0.54 ± 0.07	0.20 ± 0.01	0.76 ± 0.03	0.19 ± 0.01
Promptist	0.65 ± 0.02	0.19 ± 0.01	0.76 ± 0.03	0.17 ± 0.01
GFlowNets	0.73 ± 0.02	0.18 ± 0.01	0.63 ± 0.03	0.16 ± 0.01
PAG (Ours)	**0.88 ± 0.02**	**0.21 ± 0.01**	**0.88 ± 0.04**	**0.20 ± 0.01**

E. Comparison with Directly Fine-tuning Diffusion Models

In this section, we explain in detail the comparison with directly fine-tuning diffusion models to generate images with desired properties.

E.1. Experiment Setup

We strictly follow the experiment setup of DDPO[2] and DPOK[3] for fine-tuning diffusion models. We fine-tune diffusion models with aesthetic quality as a reward function and use prompts from a list of 45 common animals.

E.2. Comparison with DPOK

We also compare our method with DPOK [11], another representative method for fine-tuning diffusion models with black-box reward functions. As shown in Figure 10, generated images from DPOK converge to similar styles, whereas PAG generates diverse and high-quality images.

F. Additional Visualizations

In this section, we present additional visualizations to show the effectiveness of PAG for text-to-image generation as shown in Figures 11-12 (besides Figure 5 in the main text). We also summarize the results for robustness across different reward functions and transferability to different text-to-image diffusion models as shown in Figure 13-14.

[2]https://github.com/jannerm/ddpo
[3]https://github.com/google-research/google-research/tree/master/dpok

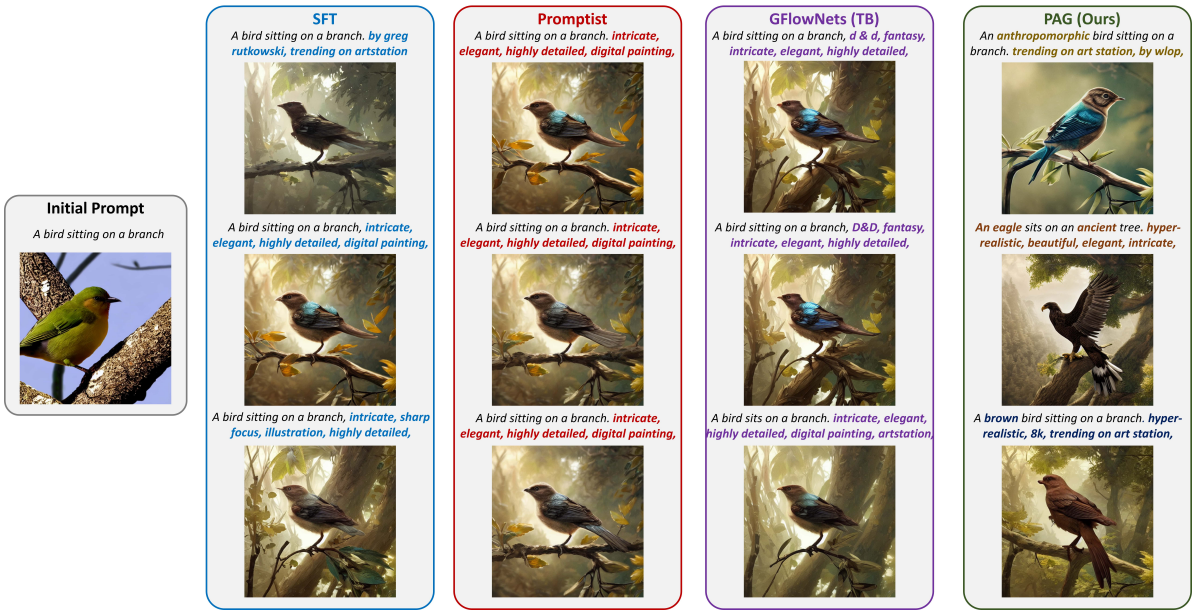


Figure 11. Additional images generated by optimized prompts using Stable Diffusion v1.4. We use the same seed to visualize the effect solely on prompt adaptation.

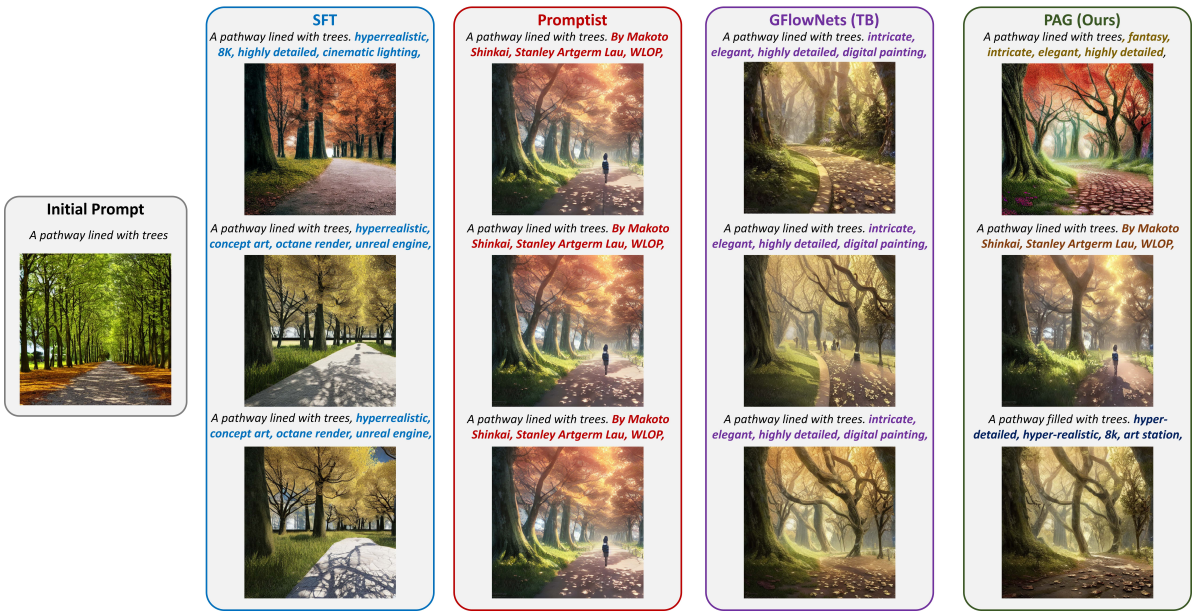


Figure 12. Additional images generated by optimized prompts using Stable Diffusion v1.4. We use the same seed to visualize the effect solely on prompt adaptation.

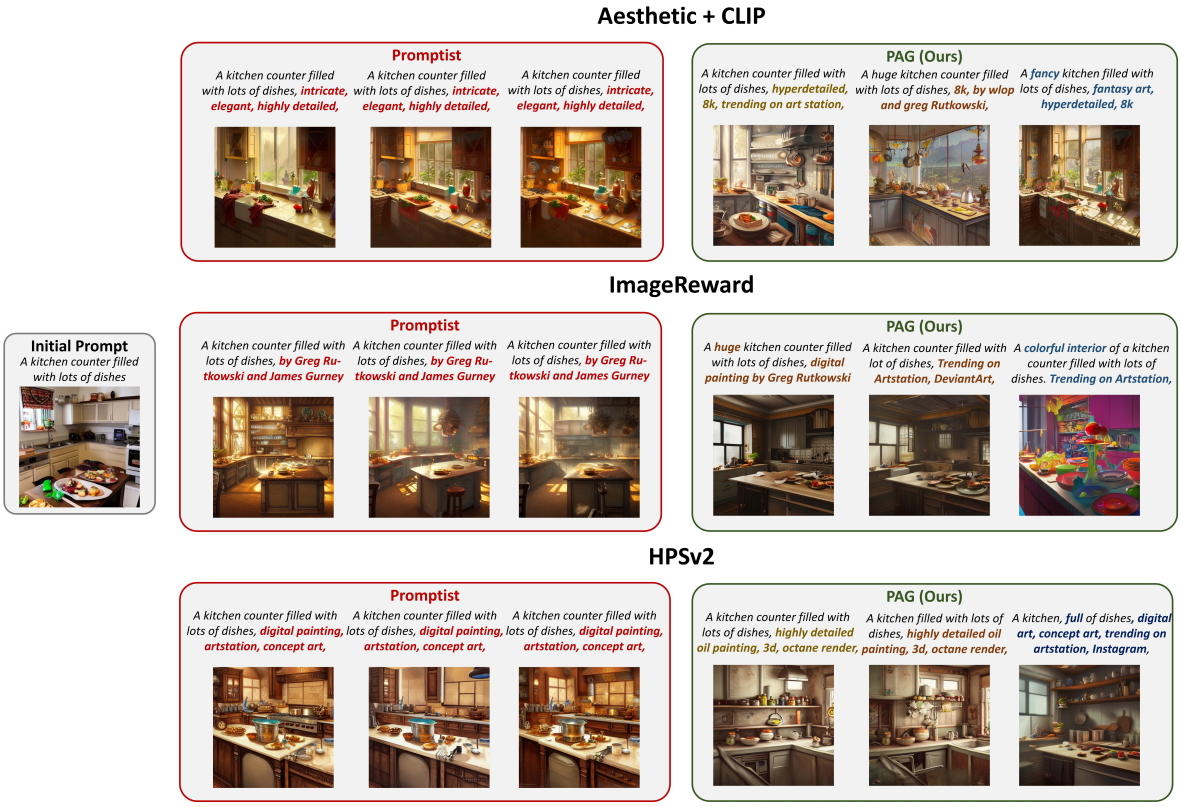


Figure 13. Images generated with prompts fine-tuned with different reward functions. We use the same seed to visualize the effect solely on prompt adaptation.

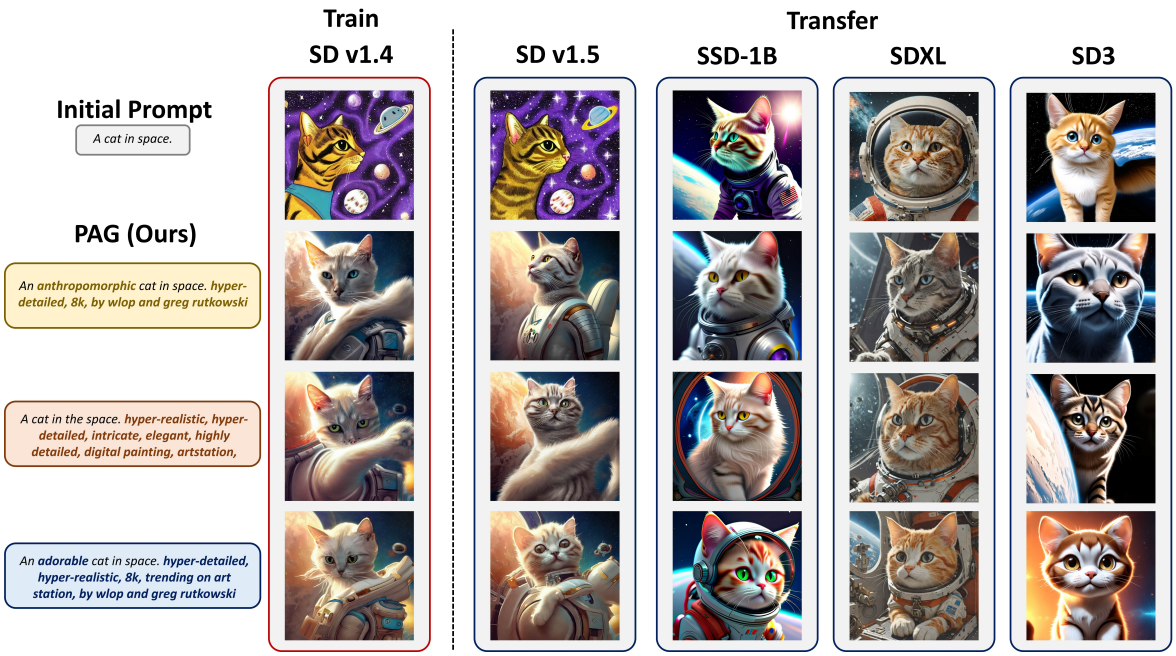


Figure 14. Images generated with different text-to-image diffusion models. We use the same seed to visualize the effect solely on prompt adaptation.