FAST DIVERSITY-PRESERVING REWARD FINETUNING OF DIFFUSION MODELS VIA NABLA-GFLOWNETS

Anonymous authors
Paper under double-blind review

ABSTRACT

While one commonly trains large diffusion models by collecting datasets on target downstream tasks, it is often desired to finetune pretrained diffusion models on some reward functions that are either designed by experts or learned from small-scale datasets. Existing methods for finetuning diffusion models typically suffer either 1) lack of diversity in generated samples, or 2) costly finetuning and slow convergence. Inspired by recent successes in generative flow networks (GFlowNets), a class of probabilistic models that sample with the unnormalized density of a reward function, we propose a novel GFlowNet method dubbed Nabla-GFlowNet (abbreviated as ∇-GFlowNet), together with an objective called ∇-DB, plus its variant *residual* ∇-DB for finetuning pretrained diffusion models. These objectives leverage the rich signal in reward gradients for diversity-aware finetuning. We empirically show that our proposed *residual* ∇-DB achieves fast yet diversity- & prior-preserving finetuning of StableDiffusion, a large-scale text-conditioned image diffusion model, on different realistic reward functions.

1 INTRODUCTION

Diffusion models [14, 50, 41] are a powerful class of generative models that model highly complex data distributions as the results of a sequence of multi-scale denoising steps. They prove capable of generating with high-fidelity a wide range of entities, including but not limited to images [41, 9], videos [15], 3D objects [69, 38, 28, 29], molecules [61, 16], languages [43]. State-of-the-arts diffusion models for downstream applications are typically large in network size and demand a significant amount of data to train.

It is however often desirable that one finetunes a pretrained diffusion models with a given reward function —- either from some learned reward function in the scenario of reinforcement learning from human feedback (RLHF) [6, 34] or from some expert design [47, 33]. While existing methods achieve fast convergence of reward maximization [60, 7], typically through reinforcement learning, many of these methods are either mode-seeking and fail in generating diverse samples or too expensive to use in large-scale settings.

Inspired by recent work in generative flow networks (GFlowNets), a class of probabilistic models that aim to sample from unnormalized density distributions, we propose a novel training objective, dubbed ∇-DB, that leverages the rich information in reward gradients. The variant *residual* ∇-DB, by leveraging the structure of diffusion models, allows us to perform fast, diversity preserving and prior-preserving amortized finetuning with rather long sampling sequences of diffusion models.

We summarize our contributions below:

- We propose ∇-DB, a novel objective for GFlowNets that leverages the rich information in reward gradients.
- For the purpose of finetuning with a pretrained prior, we propose a variant called *residual* ∇-DB that leverages the properties of diffusion models for efficient finetuning.
- We empirically show that with the propose *residual* ∇-DB objective, we may achieve diversity-preserving yet fast finetuning of diffusion models.

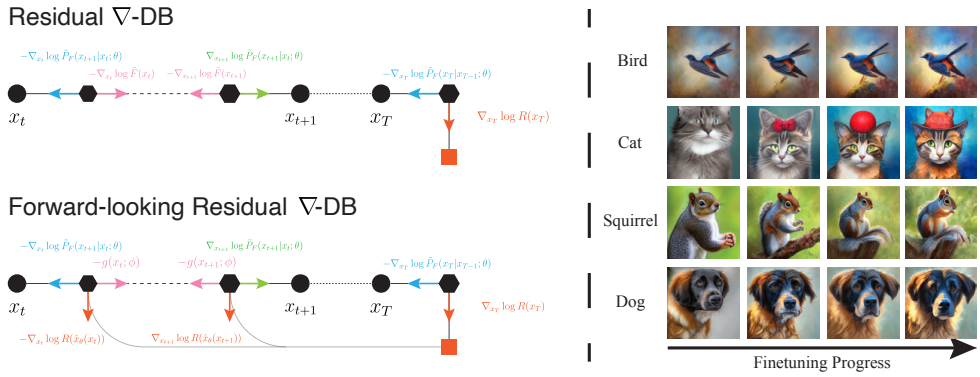

Figure 1: Left: Illustration of the proposed *residual* ∇-DB objective, along with its forward-looking variant. All "forces" (from the finetuned diffusion model, the pretrained model, the reward gradient and the learned flow function) on each hexagon should adds to zero. Right: Generated image from a model finetuned with the proposed *residual* ∇-DB on Aesthetic Score reward. The text prompt for each row is shown on the left. The leftmost figure is the image generated by the pretrained model while the rightmost one is from the model finetuned for 200 iterations.

2 Preliminaries

2.1 Diffusion models and RL-based finetuning

Diffusion models [14, 48, 50] are a class of hierarchical latent models that model the generation process as a sequence of denoising steps. Different from the convention in diffusion model literature, for convenience we adopt in this paper the reverse time of arrow where x_T means samples from the data distribution and the sampling process starts from $t = 0$. Under this convention, the probability of the generated samples is:

$$P_F(x_T) = \int_{x_{0:T-1}} P_0(x_0) \prod P_F(x_t|x_{t-1}) dx_{0:T-1}. \tag{1}$$

Here $P_0(x_0)$ is a fixed initial distribution, $P_F(x_T)$ is the likelihood of the model generating data x_T, and the noisy states x_t in the intermediate time steps are constructed by a pre-defined noising process, and the *forward policy* $P_F(x_t|x_{t-1})$ is the denoising step of the diffusion model[1]. Take DDPM [14] as an example: the corresponding noising process is $q(x_{t-1}|x_t) = \mathcal{N}(\sqrt{\alpha_{t-1}/\alpha_t}x_t, \sqrt{1-\alpha_{t-1}/\alpha_t}I)$, which induces $q(x_t|x_T) = \mathcal{N}(\sqrt{\alpha_{T-t}}x_T, \sqrt{1-\alpha_{T-t}}I)$, where $\{\alpha_t\}_t$ is a noise schedule set. With this noising process defined, the training loss is:

$$\mathop{\mathbb{E}}_{t\sim\text{Uniform}(\{1,\ldots,T\}),\epsilon\sim\mathcal{N}(0,I),x_T\sim\mathcal{D}} w(t)\left\|x_\theta(\sqrt{\alpha_{T-t}}x_T + \sqrt{1-\alpha_{T-t}}\epsilon, t) - x_T\right\|^2, \tag{2}$$

where $\mathcal{D}$ is a dataset, $w(t)$ is a certain schedule weighting function, and $x_\theta(x_t, t)$ is a data prediction model that predict the clean data x_T given a noisy data x_t at time step t.

The sequential sampling process of diffusion models is Markovian and one could construct an Markov desicion process (MDP) to describe its denoising process. We refer to Black et al. [5], Zhang et al. [68] for details of the MDP specification. Given such an MDP defined, one may finetune diffusion models with techniques like DDPO [5] by collecting on-policy sample trajectories $\{(x_1, ..., x_T)\}$ and optimize the forward (denoising) policy with a given terminal reward $R(x_T)$.

2.2 Generative flow networks (GFlowNets)

GFlowNets [4, 2] are a class of probabilistic methods to train a sampling policy $P_F(s'|s)$, where the generation process starts from some initial state s_0, makes a series of stochastic transitions $(s \to s')$ in a direct acyclic graph of states, and eventually reach a terminal state according to an unnormalized probability density, or reward. Similar to the role of noising process in diffusion

[1]The "forward" and "backward" directions in the GFlowNet literature are the opposite of those in the diffusion literature.

models, in GFlowNets a backward policy $P_B(s|s')$ is defined (either fixed or learnable) to distribute the unnormalized density of the terminal target distribution back to its ancestor states. One may imagine that the forward policy $P_F(s'|s)$ distributes density from all states s that lead to s' and similarly the backward policy $P_B(s|s')$ distributes density from all states s' to an ancestor s. If P_F and P_B "matches" each other, one obtain on each state s an unnormalized density, called the flow function $F(s)$. One form of the matching condition states is the the detailed balance condition:

Detailed Balance (DB). A valid GFlowNet with a forward policy $P_F(s'|s)$, a backward policy $P_B(s|s')$, and a flow function $F(s)$ satisfies the following DB condition for all transition $(s \to s')$

$$P_F(s'|s)F(s) = P_B(s|s')F(s'). \tag{3}$$

Hence we have the following GFlowNet DB loss on the logarithm probability space:

$$L_{\text{DB}}(s, s') = \Big(\log P_F(s'|s) + \log F(s) - \log P_B(s|s') - \log F(s')\Big)^2 \tag{4}$$

with an extra terminal constraint $F(s_f) = R(s_f)$ to incorporate target reward information.

In the context of time-indexed sampling processes such as diffusion models, the transition graph of states $s \triangleq (x_t, t)$ is naturally acyclic, as it adheres to the arrow of time [63]. With slight abuse of terminology that we use x_t to represents the tuple (x_t, t) when necessary, for time-indexed settings the *forward policy* is $P_F(x_{t+1}|x_t)$, the *backward policy* is $P_B(x_t|x_{t+1})$, and the *flow function* is $F(x_t)$[2]. The corresponding DB condition is therefore

$$P_F(x_{t+1}|x_t)F(x_t) = P_B(x_t|x_{t+1})F(x_{t+1}). \tag{5}$$

To finetune a diffusion model with DB losses [68], one can simply set $P_F(x_{t+1}|x_t)$ to be the sampling process and fix $P_B(x_t|x_{t+1})$ to be the noising process used by the pretrained diffusion model. We refer to Zhang et al. [63], Lahlou et al. [23], Zhang et al. [67] for more detailed discussion about the theory of diffusion being a specification of GFlowNets.

3 METHOD

3.1 ∇-DB: THE GRADIENT-INFORMED ALTERNATIVE TO DB

In our setting, we do not have access to any dataset of images, but are given an external positive-valued reward function $R(\cdot)$ and we need to train a generative model to learn from it. While the GFlowNet-based algorithm can effectively achieve this and also encourages diversity, it only leverages the zeroth-order reward information and does not require any differentiability of the reward function. Yet, it is often beneficial to directly make use of the reward gradient as signal, since it brings higher dimensional information about the optimization landscape and thus enables more effective optimization. We are therefore motivated to develop ∇-GFlowNet, a method that builds upon GFlowNet-based algorithms to take full advantage of the reward gradient signal. To achieve this, we take derivatives on the logarithms of both sides of the DB condition (logarithm of Equation 5) with respect to x_{t+1} and obtain a necessary condition, which we call the forward[3] ∇-DB condition:

$$\nabla_{x_{t+1}} \log P_F(x_{t+1}|x_t) = \nabla_{x_{t+1}} \log P_B(x_t|x_{t+1}) + \nabla_{x_{t+1}} \log F(x_{t+1}), \tag{6}$$

and hence the corresponding forward ∇-DB objective $L_{\overrightarrow{\nabla}\text{DB}}(x_t, x_{t+1})$ to be

$$\Big\|\nabla_{x_{t+1}} \log P_F(x_{t+1}|x_t) - \nabla_{x_{t+1}} \log P_B(x_t|x_{t+1}) - \nabla_{x_{t+1}} \log F(x_{t+1})\Big\|^2, \tag{7}$$

with the terminal flow loss on the logarithm scale

$$L_{\nabla\text{DB-terminal}}(x_T) = \Big\|\nabla_{x_T} \log F(x_T) - \beta \nabla_{x_T} \log R(x_T)\Big\|^2, \tag{8}$$

where β is a temperature coefficient and serve as a hyperparameter in the experiments. Notice that, by taking derivatives on the logarithms, we obtain the score function $\nabla_{x_{t+1}} \log P_F(x_{t+1}|x_t)$ of the

[2]We write $F_t(x_t)$ as $F(x_t)$ out of simplicity.

[3]Since the derivative is taken with respect to x_{t+1}.

finetuned diffusion model. Indeed, the ∇-DB loss is closely related to a Fisher divergence, also known as Fisher information score [21] (see Appendix B.1).

Similarly, by taking the derivative of both sides in Equation 5 with respect to x_t, one obtains the reverse ∇-DB objective:

$$L_{\overleftarrow{\nabla}\text{DB}}(x_t, x_{t+1}) = \left\| \nabla_{x_t} \log P_F(x_{t+1}|x_t) - \nabla_{x_t} \log P_B(x_t|x_{t+1}) + \nabla_{x_t} \log F(x_t) \right\|^2. \quad (9)$$

Such ∇-GFlowNet objectives constitute a valid GFlowNet algorithm (see the proof in Section B.2):

Proposition 1. *If $L_{\overrightarrow{\nabla}DB}(x_t, x_{t+1}) = L_{\overleftarrow{\nabla}DB}(x_t, x_{t+1}) = 0$ for any denoising transition (x_t, x_{t+1}) over the state space and $L_{\nabla DB\text{-terminal}}(x_T) = 0$ for all terminal state x_T, then the resulting forward policy generate samples x_T with probability proportional to the reward function $R(x_T)^\beta$.*

Remark 2. The original detailed balance condition propagates information from the reward function to each state flow function in the sense of $F(x_{t+1}) \to (F(x_t), P_F(x_{t+1}|x_t))$, assuming the backward (noising) policy is fixed (*i.e.*, there is no learning component in the diffusion noising process). In our case, if we take a close look at Equation 7, we can see that $L_{\overrightarrow{\nabla}\text{DB}}(x_t, x_{t+1})$ could propagate the information from $F(x_{t+1})$ to the forward policy $P_F(x_{t+1}|x_t)$ but not $F(x_t)$.

Remark 3. Compared to previous GFlowNet works which use a scalar-output network to parameterize the (log-) flow function, in ∇-GFlowNet we can directly use a U-Net [42]-like architecture that (whose output and input shares the same number of dimension) to parameterize $\nabla \log F(\cdot)$, which potentially provides more modeling flexibility. Furthermore, it is possible to initialize $\nabla \log F(\cdot)$ with layers from the pretrained model so that it can learn upon known semantic information.

3.2 Residual ∇-DB for reward finetuning of pretrained models

With the ∇-DB losses, one can already finetune a diffusion model to sample from the reward distribution $R(x)$. However, the finetuned model may eventually over-optimize the reward and thus forget the pretrained prior (*e.g.*, how natural images look like). Instead, similar to other amortized inference work [70, 56], we consider the following objective with an augmented reward:

$$P_F(x_T) \propto R(x_T)^\beta P_F^{\#}(x_T)^\eta, \quad (10)$$

where $R(x)$ is the positive-valued reward function, β is the temperature coefficient, η is a parameter to control the strength of the prior, $P_F^{\#}(x_T)$ is the marginal distribution of the pretrained model[4] and $P_F(x_T)$ is the marginal distribution of the finetuned model (as defined in Equation 1).

Because both the finetuned and pretrained model share the same backward policy P_B (the noising process of the diffusion models), we can remove the P_B term and obtain the forward *residual* ∇-DB condition by subtracting the forward ∇-DB equation for the pretrained model from the that of the finetuned model:

$$\underbrace{\nabla_{x_{t+1}} \log P_F(x_{t+1}|x_t) - \eta \nabla_{x_{t+1}} \log P_F^{\#}(x_{t+1}|x_t)}_{\nabla_{x_{t+1}} \log \tilde{P}_F(x_{t+1}|x_t)\text{: residual policy score function}} = \underbrace{\nabla_{x_{t+1}} \log F(x_{t+1}) - \nabla_{x_{t+1}} \log F^{\#}(x_{t+1}; \eta)}_{\nabla_{x_{t+1}} \log \tilde{F}(x_{t+1})\text{: residual flow score function}}. \quad (11)$$

With the two residual terms defined above, we obtain the forward *residual* ∇-DB objective:

$$L_{\overrightarrow{\nabla}\text{DB-res}}(x_t, x_{t+1}) = \left\| \nabla_{x_{t+1}} \log \tilde{P}_F(x_{t+1}|x_t) - \nabla_{x_{t+1}} \log \tilde{F}(x_{t+1}) \right\|^2 \quad (12)$$

with the terminal flow loss in Equation 8. Similarly, we have the reverse *residual* ∇-DB loss:

$$L_{\overleftarrow{\nabla}\text{DB-res}}(x_t, x_{t+1}) = \left\| \nabla_{x_t} \log \tilde{P}_F(x_{t+1}|x_t) + \nabla_{x_t} \log \tilde{F}(x_t) \right\|^2 \quad (13)$$

The terminal loss of the residual ∇-DB method stays the same form as in Equation 8

$$L_{\nabla\text{DB-terminal}}(x_T) = \left\| \nabla_{x_T} \log \tilde{F}(x_T) - \beta \nabla_{x_T} \log R(x_T) \right\|^2. \quad (14)$$

[4]We use the notation of # to indicate quantities of the pretrained model.

Proposition 4. *If $L_{\overrightarrow{\nabla}\text{DB-res}}(x_t, x_{t+1}) = L_{\overleftarrow{\nabla}\text{DB-res}}(x_t, x_{t+1}) = 0$ for any denoising transition (x_t, x_{t+1}) over the state space and $L_{\nabla\text{DB-terminal}}(x_T) = 0$ for all terminal state x_T, then the resulting forward policy generate samples x_T with probability proportional to $R(x_T)^\beta P_F^\#(x_T)$.*

Remark 5. We point out that perform the same way of deriving Equation 11, *i.e.*, subtraction between GFlowNet conditions from the finetuned and pretrained model, on the DB condition without gradient, we can obtain a *residual* DB condition $\tilde{F}(x_t)P_F(x_{t+1}|x_t) = P_F^\#(x_{t+1}|x_t)\tilde{F}(x_{t+1})$. Multiplying this condition across time and eliminate the term of intermediate $\tilde{F}(x_t)$ will lead to the objective derived in the relative GFlowNet work [55] as shown in Section B.4, which is a prior paper that proposes to work on reward finetuning GFlowNets with a given pretrained model.

Remark 6. One may completely eliminate the need for any residual flow score function with the *residual* ∇-DB conditions of both directions: $\nabla_{x_{t+1}} \log \tilde{P}_F(x_{t+1}|x_t) = -\nabla_{x_{t+1}} \log \tilde{P}_F(x_{t+2}|x_{t+1})$. The bidirectional *residual* ∇-DB condition can be analogously understood as the balance condition of two forces from x_t and x_{t+2} acting on x_{t+1}: if not balanced, one can locally find some other x_{t+1} that makes both transitions more probable.

Flow reparameterization through forward-looking (FL) trick. Though mathematically solid, such a bidirectional condition suffers from inefficient credit assignment for long sequences, a problem commonly observed in RL settings [52, 54]. Instead, we may leverage the priors we have from the pretrained diffusion model to speed up the finetuning process, by considering the individual conditions for the forward and reverse directions. Specifically, we employ the forward-looking (FL) technique for GFlowNets [35, 68] and parameterize the residual flow score function with a "baseline" of the "one-step predicted reward gradient":

$$\nabla_{x_t} \log \tilde{F}(x_t) \triangleq \beta\gamma_t \nabla_{x_t} \log \underbrace{R(\hat{x}_\theta(x_t))}_{\text{predicted reward}} + g_\phi(x_t) \tag{15}$$

where γ_t is the scalar to control the strength of forward looking (with $\gamma_T = 1$) and $g_\phi(x_t)$ is the actual neural network with parameters ϕ satisfying a terminal constraint $g_\phi(x_T) = 0$. Here $\hat{x}_\theta(\cdot)$ is the one-step clean data prediction defined in Equation 2. This FL technique is to involve a useful external information, which is a partial signal $\nabla_{x_t} \log R(\hat{x}_\theta(x_t))$ about state x_t, such that the learning of the flow function becomes easier, as the scale of the difference to its optimal value becomes smaller [35].

We therefore obtain the forward-looking version of *residual* ∇-DB losses of both directions:

$$L_{\overrightarrow{\nabla}\text{DB-FL-res}}(x_t, x_{t+1}) = \left\| \nabla_{x_{t+1}} \log \tilde{P}_F(x_{t+1}|x_t;\theta) - \left[\beta\gamma_t \nabla_{x_{t+1}} \log R(\hat{x}_\theta(x_{t+1})) + g_\phi(x_{t+1})\right] \right\|^2. \tag{16}$$

$$L_{\overleftarrow{\nabla}\text{DB-FL-res}}(x_t, x_{t+1}) = \left\| \nabla_{x_t} \log \tilde{P}_F(x_{t+1}|x_t) + \left[\beta\gamma_t \nabla_{x_t} \log R(\hat{x}_\theta(x_t)) + g_\phi(x_t)\right] \right\|^2. \tag{17}$$

What's more, the corresponding terminal loss objective now becomes

$$L_{\nabla\text{DB-FL-terminal}}(x_T) = \left\| \nabla_{x_T} \log \tilde{F}(x_T) - \beta\gamma_t \nabla_{x_T} \log R(x_T) \right\|^2 = \left\| g_\phi(x_T) \right\|^2, \tag{18}$$

which indicates that the actual parameterized flow network g_ϕ should take a near-zero value for terminal states x_T. The total loss on a collected trajectory $\tau = (x_1, ..., x_T)$ is therefore

$$\sum_t \left[w_F(t) L_{\overrightarrow{\nabla}\text{DB-FL-res}}(x_t, x_{t+1}) + w_B(t) L_{\overleftarrow{\nabla}\text{DB-FL-res}}(x_t, x_{t+1}) \right] + L_{\nabla\text{DB-FL-terminal}}(x_T) \tag{19}$$

where $w_F(t)$ and $w_B(t)$ are scalar weights to control the relative importance of each term.

We summarize the resulting algorithm in Algorithm 1 in Appendix.

Choice of FL scale. Naïvely setting $\gamma_t = 1$ can be too aggressive especially when the reward scale η and the learning rate are set to a relatively high value. Inspired by the fact that in diffusion models $F_t(x_t)$ can be seen as $R(x)$ smoothed with a Gaussian kernel, we propose to set $\gamma_t = \alpha_{T-t}$. We compare this design choice with the naïve one in the appendix.

4 EXPERIMENTS

4.1 BASELINES

For gradient-free methods, we consider DAG-DB [68] (*i.e.*, GFlowNet finetuning with the DB objective) and DDPO [5]. Since the original DB objective aims to finetune with $R^\beta(x)$ instead of $P_F^\#(x_T)R^\beta(x_T)$, we also consider the residual DB loss, defined as

$$L_{\text{DB-FL-res}}(x_t, x_{t+1}) = \Big(\log \tilde{P}_F(x_{t+1}|x_t) - \beta \log R(\hat{x}_\theta(x_{t+1})) - g(x_{t+1};\phi)\Big)^2. \tag{20}$$

For other gradient-aware finetuning methods, we consider ReFL [60] and DRaFT [7]. ReFL samples a trajectory and stops some random time step t, with which it maximizes $R(\hat{x}_\theta(\nabla\!\!\!/(x_t))$ where $\hat{x}_\theta(\cdot)$ is the one-step sample prediction function and $\nabla\!\!\!/$ is the stop-gradient operation. Different from ReFL, DRaFT samples some time step $T-K$ (typically $K=1$) and expand the computational graph of DDPM from $\nabla\!\!\!/(x_{T-K})$ to x_T so that the reward signal $R(x_T)$ can be backpropated to x_{T-K}, with all x_t in the previous time steps removed from this computational graph. A variant of DRaFT called DRaFT-LV performs few extra steps of "noising-denoising" on the sampled x_T before feeding it into the reward function $R(\cdot)$.

4.2 REWARD FUNCTIONS, PROMPT DATASETS AND METRICS

For the main experiments, we consider two reward functions: Aesthetic Score [24], Human Preference Score (HPSv2) [57, 58] and ImageReward [60], all of which trained on large-scale human preference datasets such as LAION-aesthetic [24]. For base experiments with Aesthetic Score, we use a set of 45 simple animal prompts as used in DDPO [5]; for those with HPSv2, we use photo+painting prompts from the human preference dataset (HPDv2) [57]. To measure the diversity of generated images, we follow Domingo-Enrich et al. [10] and compute the variance of latent features extracted from a batch of generated images (we use a batch of size 64). Using the same set of examples, we evaluate the capability of prior following, we compute the per-prompt FID score between images generated from the pretrained model and from the finetuned model and take the average FID score over all evaluation prompts.

4.3 EXPERIMENT SETTINGS

For all methods, we use 50-step DDPM sampler to construct the MDP. We use StableDiffusion-v1.5 [41] as the base models. For the finetuned diffusion model policies, we use low-rank adaptation (LoRA) [17]. The residual flow score function in *residual* ∇-DB is set to be a scaled-down version of the StableDiffusion U-Net, whereas the flow function (in DAG-DB and *residual* DB) is set to be a similar network but without the U-Net decoding structure (since the desired output is a scalar instead of an image vector). Both networks are initialized with tiny weights in the final output layers.

As the landscape of $R(x)$ can be highly non-smooth, we approximate $\nabla_{x_t}\log R(\hat{x}_\theta(x_t))$ with $\mathbb{E}_{\epsilon\sim\mathcal{N}(0,c)}\nabla_{x_t}\log R(\hat{x}_\theta(x_t)+\epsilon)$ where c is a tiny constant. For StableDiffusion [41], since the diffusion process runs in the latent space, the reward function is instead $\mathbb{E}_{\epsilon\sim\mathcal{N}(0,c)}\nabla_{x_t}\log R(\text{decode}(\hat{x}_\theta(x_t)+\epsilon))$ in which decode$(\cdot)$ is the pretrained (and frozen) VAE decoder and c is set to 2×10^{-3}, slightly smaller than one pixel (i.e., $1/255$). We approximate this expectation with 3 independent samples for each transition in each trajectory. For all experiments, we try 3 random seeds. Unless otherwise specified, we set $w_B(t)=1$.

To stabilize the training process of our method (*residual* ∇-DB), we follow the official repo of DAG-DB and uses output regularization: $\lambda\|\epsilon_\theta(x_t)-\epsilon_{\theta^\dagger}(x_t)\|^2$ where $\theta^\dagger$ is the diffusion model parameters in the previous update step[5]. We set the output regularization strength $\lambda=1000$ in Aesthetic Score experiments and $\lambda=100$ in HPSv2 experiments. For all experiments with *residual* ∇-DB, we set the learning rate to 1×10^{-3} and ablate over a set of choices of reward temperature β, in a range such that the reward gradients are more significant than the residual policy score function $\nabla_{x_t}\log P_F(x_{t+1}|x_t)$ of the pretrained model. We always set $w_F(t)=1$ for all t's and unless

[5]Essentially a Fisher divergence between the pretrained and the finetuned distributions conditioned on x_t. Similar regularization with KL divergence has been seen in popular on-policy RL algorithms like TRPO [44] and PPO [45].

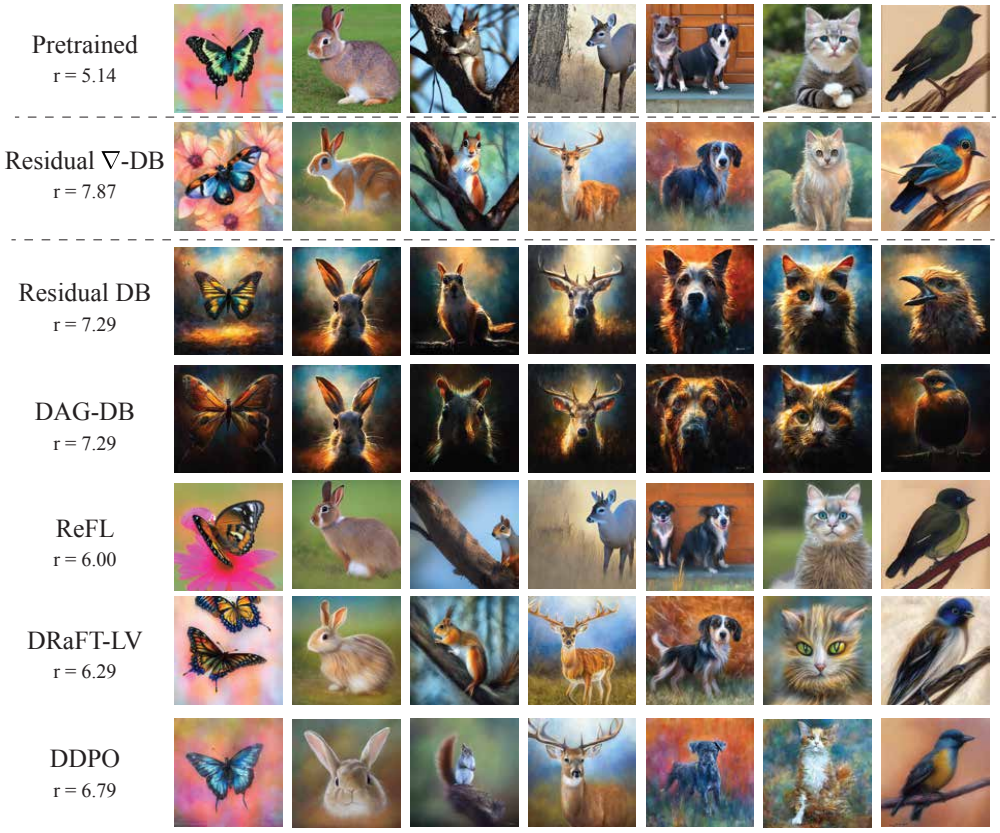

Figure 2: Comparison between images generated by models finetuned with different methods for a maximum of 200 update steps. For each method, we pick the model trained that produces the most visually-appealing figure among all model checkpoints, as methods like ReFL and DRaFT-LV easily collapses (as illustrated in Fig. 3). For each method, we show the average reward of the corresponding presented images.

otherwise specified we set $w_b(t) = 1$. For each epoch, we collect 64 generation trajectories for each of which we randomly shuffle the orders of transitions. We use the number of gradient accumulation steps to 4 and for each 32 trajectories we update both the forward policy and the residual flow score function. For *residual* ∇-DB in most of the experiments, we sub-sample 10% of the transitions in each collected trajectory for training by taking one single uniformly sample in uniformly split time-step intervals but ensure that the final transition step always included.

For *residual* DB and DAG-DB, we set the learning rate to 3×10^{-4} with output regularization strength $\lambda = 1$ The sampling and training procedures are similar to that of *residual* ∇-DB experiments. For ReFL, we follow the official repo and similarly set the random stop time steps to between 35 and 49. For DRaFT, since the official code is not released, we follow the settings in AlignProp [39], a similar concurrent paper. We set the loss for both ReFL and DRaFT to $-\mathbb{E}_{x_T \sim P_F} \text{ReLU}(R(\text{decode}(x_T)))$ where the ReLU function is introduced for training stability in the case of the ImageReward reward function.

Figure 3: Finetuning with our ∇-GFlowNet is stable compared to other baselines.

4.4 Results

General experiments. In Figure 6 and Table 1, we show the evolution of reward, DreamSim diversity of all methods with the mean curves and the corresponding standard deviations (on 3 random seeds). Our proposed *residual* ∇-DB is able to achieve comparable convergence speed, measured in update steps, to that of the gradient-free baselines while those diversity-aware baselines fail to do so. In Figure 7, we plot the diversity-reward tuples and FID-reward tuples for models evaluated

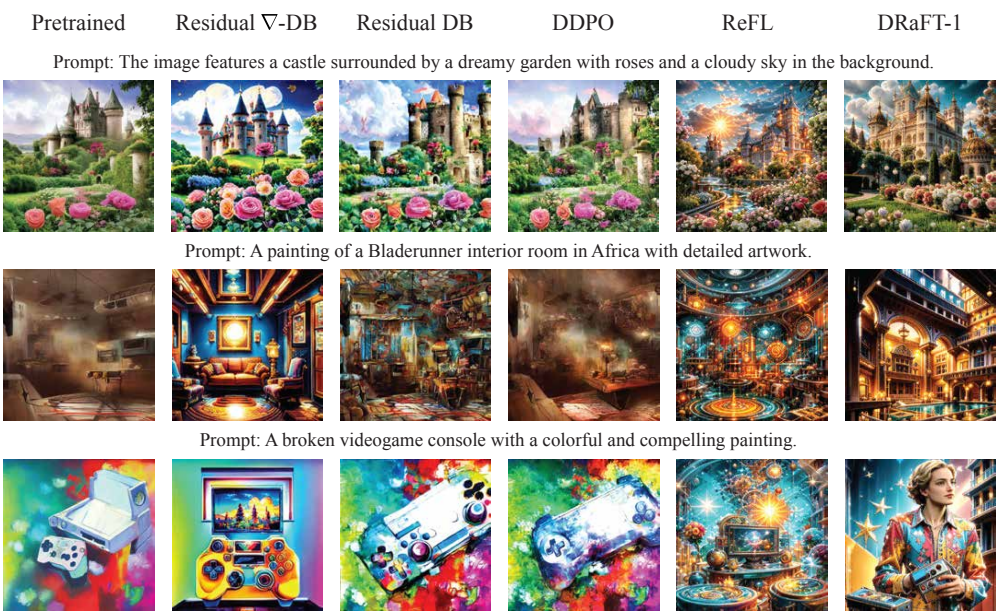

Figure 4: Qualitative results on HPSv2.

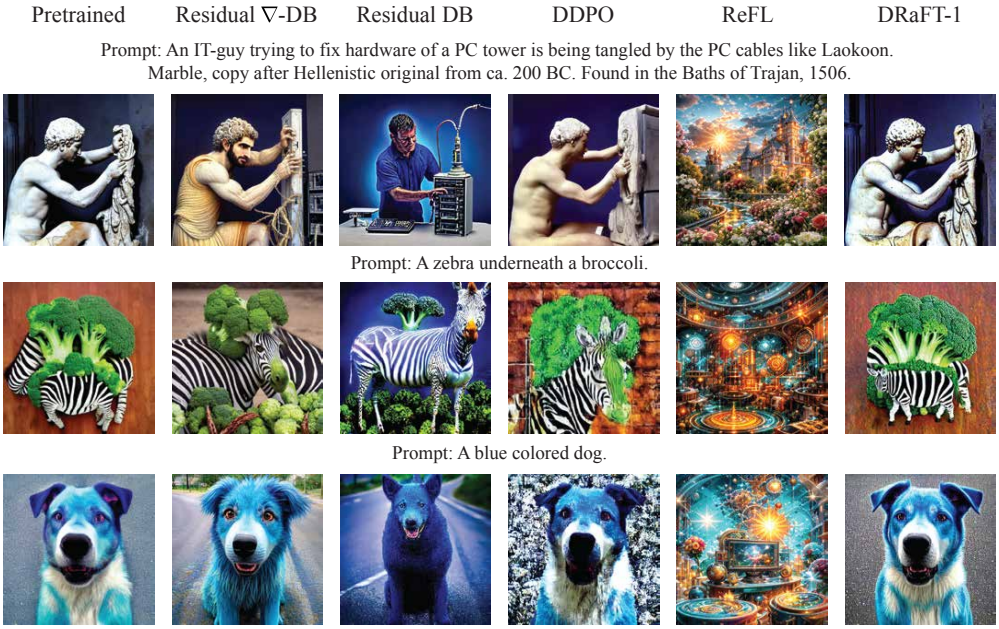

Figure 5: Qualitative results on ImageReward.

at different checkpoints (every 5 update steps) and show that our method achieves better trade-off between diversity and reward and between prior-following and reward. The gradient-informed baselines, ReFL and the DRaFT variants, generally behave worse than *residual* ∇-DB due to their mode-seeking nature. Qualitatively, we show that the model finetuned with *residual* ∇-DB on Aesthetic Score generate more aesthetic and more diverse samples in both style and subject identity (Fig. 2, and Fig. 24 in the appendix), while the other baselines exhibit mode collapse or even catastrophic forgetting of pretrained image prior. We also demonstrate some of the images generated by the diffusion model finetuned with *residual* ∇-DB on HPSv2 in Figure 4 and on ImageReward in Figure 5. Furthermore, we qualitatively show that *residual* ∇-DB is robust while with the gradient-informed baseline methods are prone to training collapse (Fig. 3). Due to the limited space, we some ablation studies to Appendix F, and only show the most important ones in the main text. For the same reason, we also leave plots for experiments on HPSv2 and ImageReward to Appendix E.

Method	Aesthetic Score			HPSv2			ImageReward		
	Reward (↑)	Diversity DreamSim (↑, 10^{-2})	FID (↓)	Reward (↑, 10^{-1})	Diversity DreamSim (↑, 10^{-2})	FID (↓)	Reward (↑, 10^{-1})	Diversity DreamSim (↑, 10^{-2})	FID (↓)
Base Model	$5.83_{\pm 0.01}$	$35.91_{\pm 0.00}$	$216_{\pm 1}$	$2.38_{\pm 0.13}$	$37.75_{\pm 0.21}$	$563_{\pm 5}$	$-0.38_{\pm 0.12}$	$41.09_{\pm 0.03}$	$468_{\pm 1}$
Ours (w_B=1)	$7.90_{\pm 0.09}$	**2.97** ± 0.05	$317_{\pm 15}$	$3.53_{\pm 0.04}$	$24.39_{\pm 0.87}$	$1000_{\pm 39}$	$2.23_{\pm 0.34}$	**39.85** ± 0.67	**501** ± 7
Ours (w_B=0)	$7.86_{\pm 0.06}$	$2.92_{\pm 0.02}$	$318_{\pm 13}$	$3.53_{\pm 0.03}$	$24.40_{\pm 0.56}$	$973_{\pm 5}$	$5.33_{\pm 0.62}$	$35.85_{\pm 0.66}$	$638_{\pm 15}$
residual DB	$7.20_{\pm 0.92}$	$1.94_{\pm 0.41}$	$1065_{\pm 587}$	$2.55_{\pm 0.06}$	$32.49_{\pm 0.47}$	$840_{\pm 107}$	**6.47** ± 0.54	$25.52_{\pm 0.35}$	$772_{\pm 19}$
DAG-DB	$7.73_{\pm 0.07}$	$1.59_{\pm 0.07}$	$595_{\pm 87}$	$2.52_{\pm 0.06}$	**32.50** ± 0.59	$866_{\pm 41}$	$4.70_{\pm 0.75}$	$26.78_{\pm 0.64}$	$809_{\pm 34}$
DDPO	$6.68_{\pm 0.14}$	$3.30_{\pm 0.10}$	**312** ± 9	$2.52_{\pm 0.04}$	$3.49_{\pm 0.03}$	**681** ± 16	$0.27_{\pm 0.38}$	$38.51_{\pm 1.49}$	$714_{\pm 25}$
ReFL	$9.53_{\pm 0.46}$	$0.82_{\pm 0.31}$	$1765_{\pm 51}$	$3.67_{\pm 0.06}$	$19.84_{\pm 1.70}$	$1191_{\pm 46}$	$1.36_{\pm 0.30}$	$36.50_{\pm 0.52}$	$597_{\pm 10}$
DRaFT-1	$10.16_{\pm 0.13}$	$0.42_{\pm 0.05}$	$1665_{\pm 182}$	$3.70_{\pm 0.06}$	$18.96_{\pm 1.35}$	$1222_{\pm 84}$	$1.59_{\pm 0.25}$	$37.27_{\pm 0.49}$	$531_{\pm 13}$
DRaFT-LV	**10.21** ± 0.34	$0.64_{\pm 0.17}$	$1854_{\pm 296}$	**3.75** ± 0.08	$21.13_{\pm 1.19}$	$1164_{\pm 43}$	$1.44_{\pm 0.25}$	$37.56_{\pm 0.09}$	$529_{\pm 18}$

Table 1: Comparison between the models finetuned with our proposed method *residual* ∇-DB and the baselines. All models are finetuned with 200 update steps. For each method, the model with the best mean reward is used for evaluation. Note that while baselines like DDPO can achieve better scores on some metric, it often comes with the price of much worse performance on some other.

Effect of reward temperature. We perform ablation study on Aesthetic Score with $\beta \in \{5000, 7000, 1000\}$ in *residual* ∇-DB. Not surprisingly, a higher reward temperature leads to faster convergence at the cost of worse diversity and worse prior-following, as observed in Figure 8.

Effect of sub-sampling. Typically, sub-sampling results in worse gradient estimates. We empirically study how sub-sampling may effect the performance and show the results in Figure 10 and **??** in the appendix. We empirically do not observe huge performance drop due to subsampling strategy, potentially because the rich gradient signals in both the reward and the flow are sufficient.

Effect of different prior strengths. We experiment with choices of prior strengths η and observed in Fig. 16 and 17 that lower η lead to better diversity-reward trade-off and faster reward convergence.

Reward finetuning with other sampling algorithms. To show that our method generalizes to different sampling diffusion algorithms, we construct another MDP based on SDE-DPM-Solver++ [30], with 20 inference steps. In Fig. 18 and 19, we obverse that our *residual* ∇-DB can still achieve a good balance between reward convergence speed, diversity preservation and prior following.

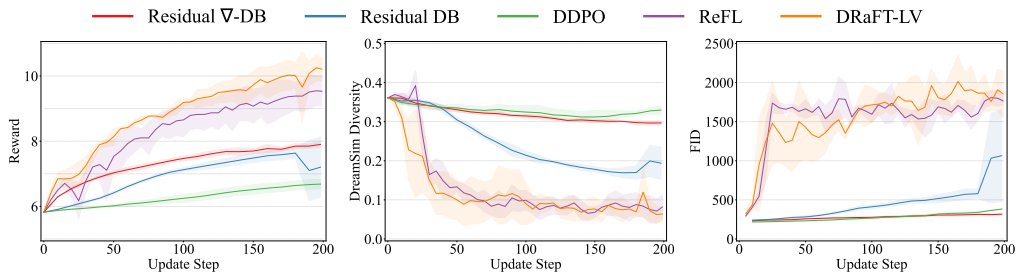

Figure 6: Convergence curves of different metrics for different methods throughout the finetuning process on Aesthetic Score. Finetuning with our proposed *residual* ∇-DB converges faster than the non-gradient-informed methods and with better diversity-preserving and prior-following capability.

5 Related Work

Reward finetuning of diffusion models. The demand for reward finetuning is probably most commonly seen in alignment, where one obtains utilize a human reference reward function to align the behavior of generative models [18, 1, 57] for better instruction following capability and better AI safety. With a reward function, typically obtained by learning from human preference datasets [72, 51], one may use reinforcement learning (RL) algorithms, for instance PPO [45], to

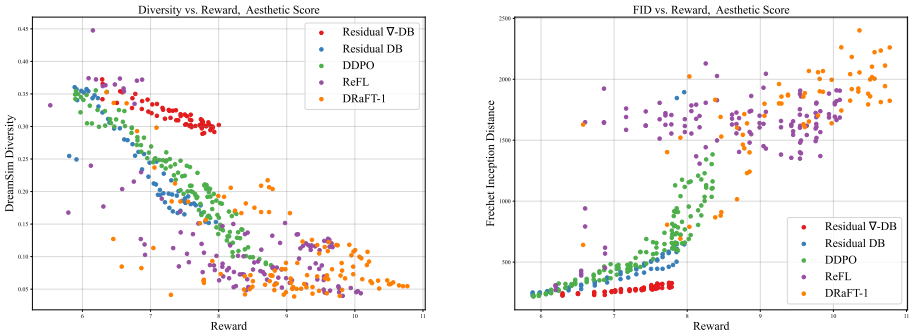

Figure 7: Trade-offs between reward, diversity and priority following (measured by FID) for different reward finetuning methods. Dots represent the evaluation results of models checkpoint saved after every 5 iterations of finetuning, where ones with larger reward, larger diversity scores and smaller FID scores are considered better.

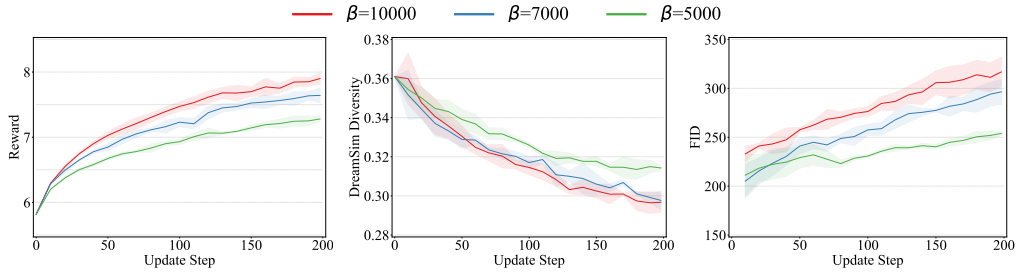

Figure 8: Higher temperature β leads to faster convergence but with less diversity and less prior following.

adapt not only autoregressive language models [34] which naturally admits a Markov decision process (MDP), but also diffusion models [5, 12]. Specifically, one can construct MDPs from some diffusion sampling algorithm [48, 30] by considering each noisy image at some inference step as a state and each denoising step as an action. Besides RL algorithms, there exist some other approaches, including stochastic optimal control [10, 53], GFlowNets [68] and some other ones akin to RL methods [26, 11]. While most of the aforementioned approaches train with only black-box rewards, once we have access to a differentiable reward function we may accelerate the finetuning process with reward gradient signals. For instance, methods exist to construct a computational graph from sampled generation trajectories to directly optimize for rewards [7, 39, 59], yet with these methods models are not trained to correctly sample according to the reward function. While one may also generate samples from the reward function without finetuning using plug-in guidance methods for diffusion models [9, 49, 22, 13] as an alternative, but the generated distributions are often very biased. Besides, reward finetuning for diffusion models is typically memory consuming as many methods require a large computational graph rolled out from long generation trajectories, for which it is typical to employ efficient finetuning techniques [17, 40, 27].

GFlowNets. Generative flow network [3], or GFlowNet in short, is a high-level algorithmic framework that introduces sequential decision-making into generative modeling [63], bridging methodology between reinforcement learning [66, 36, 37, 35, 25] and energy-based modeling [64]. GFlowNets perform amortized variational inference [32] and generate samples with probability proportional to a given density or reward function, in contrast to the typical reward maximization objective in reinforcement learning. GFlowNets can therefore be used to generate high-quality and diverse samples for applications including but not limited to drug discovery [19, 20, 46], structure learning [8], phylogenetic inference [71] and combinatorial optimization [62, 65].

6 Conclusion

We propose ∇-GFlowNet, a fast, diversity-preserving and prior-following reward finetuning method for diffusion models, by leveraging gradient information in the probabilistic framework of GFlowNets that aims to sample according to a given unnormalized density function. Our empirical results show that ∇-GFlowNet achieves a better trade-off between convergence speed, diversity in generated samples and prior following. We hope that our method sheds lights on future studies on more efficient reward finetuning strategies of diffusion models as well as related applications.

REFERENCES

[1] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022.

[2] Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. In *Advances in Neural Information Processing Systems*, 2021.

[3] Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J Hu, Mo Tiwari, and Emmanuel Bengio. GFlowNet foundations. *Journal of Machine Learning Research*, (24):1–76, 2023.

[4] Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J Hu, Mo Tiwari, and Emmanuel Bengio. Gflownet foundations. *The Journal of Machine Learning Research*, 24(1):10006–10060, 2023.

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

[6] Paul Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[7] 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.

[8] Tristan Deleu, António Góis, Chris Emezue, Mansi Rankawat, Simon Lacoste-Julien, Stefan Bauer, and Yoshua Bengio. Bayesian structure learning with generative flow networks. *Uncertainty in Artificial Intelligence (UAI)*, 2022.

[9] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *Advances in neural information processing systems*, 2021.

[10] Carles Domingo-Enrich, Michal Drozdzal, Brian Karrer, and Ricky T. Q. Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. *arXiv preprint arXiv:2409.08861*, 2024.

[11] Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. Raft: Reward ranked finetuning for generative foundation model alignment, 2023.

[12] Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, P. Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models. *ArXiv*, abs/2305.16381, 2023.

[13] Yingqing Guo, Hui Yuan, Yukang Yang, Minshuo Chen, and Mengdi Wang. Gradient guidance for diffusion models: An optimization perspective, 2024. URL `https://arxiv.org/abs/2404.14743`.

[14] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in neural information processing systems*, 2020.

[15] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in Neural Information Processing Systems*, 35, 2022.

[16] Emiel Hoogeboom, Vıctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In *International conference on machine learning*, 2022.

[17] 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.

[18] Borja Ibarz, Jan Leike, Tobias Pohlen, Geoffrey Irving, Shane Legg, and Dario Amodei. Reward learning from human preferences and demonstrations in atari, 2018.

[19] Moksh Jain, Emmanuel Bengio, Alex Hernandez-Garcia, Jarrid Rector-Brooks, Bonaventure F.P. Dossou, Chanakya Ekbote, Jie Fu, Tianyu Zhang, Micheal Kilgour, Dinghuai Zhang, Lena Simine, Payel Das, and Yoshua Bengio. Biological sequence design with GFlowNets. *International Conference on Machine Learning (ICML)*, 2022.

[20] Moksh Jain, Tristan Deleu, Jason S. Hartford, Cheng-Hao Liu, Alex Hernández-García, and Yoshua Bengio. Gflownets for ai-driven scientific discovery. *ArXiv*, abs/2302.00615, 2023. URL https://api.semanticscholar.org/CorpusID:256459319.

[21] Oliver Johnson. *Information theory and the central limit theorem*. World Scientific, 2004.

[22] Lingkai Kong, Yuanqi Du, Wenhao Mu, Kirill Neklyudov, Valentin De Bortol, Haorui Wang, Dongxia Wu, Aaron Ferber, Yi-An Ma, Carla P Gomes, et al. Diffusion models as constrained samplers for optimization with unknown constraints. *arXiv preprint arXiv:2402.18012*, 2024.

[23] Salem Lahlou, Tristan Deleu, Pablo Lemos, Dinghuai Zhang, Alexandra Volokhova, Alex Hernández-García, Léna Néhale Ezzine, Yoshua Bengio, and Nikolay Malkin. A theory of continuous generative flow networks. *International Conference on Machine Learning (ICML)*, 2023.

[24] LAION. Laion aesthetic score predictor. https://laion.ai/blog/laion-aesthetics/, 2024. Accessed: 2024-09-27.

[25] Elaine Lau, Stephen Zhewen Lu, Ling Pan, Doina Precup, and Emmanuel Bengio. Qgfn: Controllable greediness with action values, 2024. URL https://arxiv.org/abs/2402.05234.

[26] Kimin Lee, Hao Liu, Moonkyung Ryu, Olivia Watkins, Yuqing Du, Craig Boutilier, P. Abbeel, Mohammad Ghavamzadeh, and Shixiang Shane Gu. Aligning text-to-image models using human feedback. *ArXiv*, abs/2302.12192, 2023.

[27] Weiyang Liu, Zeju Qiu, Yao Feng, Yuliang Xiu, Yuxuan Xue, Longhui Yu, Haiwen Feng, Zhen Liu, Juyeon Heo, Songyou Peng, et al. Parameter-efficient orthogonal finetuning via butterfly factorization. In *International Conference on Learning Representations*, 2024.

[28] Zhen Liu, Yao Feng, Michael J Black, Derek Nowrouzezahrai, Liam Paull, and Weiyang Liu. Meshdiffusion: Score-based generative 3d mesh modeling. In *International Conference on Learning Representations*, 2023.

[29] Zhen Liu, Yao Feng, Yuliang Xiu, Weiyang Liu, Liam Paull, Michael J Black, and Bernhard Schölkopf. Ghost on the shell: An expressive representation of general 3d shapes. In *International Conference on Learning Representations*, 2024.

[30] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. In *ICLR*, 2023.

[31] Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: Improved credit assignment in gflownets. In *Advances in Neural Information Processing Systems*, 2022.

[32] Nikolay Malkin, Salem Lahlou, Tristan Deleu, Xu Ji, Edward Hu, Katie Everett, Dinghuai Zhang, and Yoshua Bengio. GFlowNets and variational inference. *International Conference on Learning Representations (ICLR)*, 2023.

[33] Azalia Mirhoseini, Anna Goldie, Mustafa Yazgan, Joe Jiang, Ebrahim Songhori, Shen Wang, Young-Joon Lee, Eric Johnson, Omkar Pathak, Sungmin Bae, et al. Chip placement with deep reinforcement learning. *arXiv preprint arXiv:2004.10746*, 2020.

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

[35] Ling Pan, Nikolay Malkin, Dinghuai Zhang, and Yoshua Bengio. Better training of gflownets with local credit and incomplete trajectories. In *International Conference on Machine Learning*, 2023.

[36] Ling Pan, Dinghuai Zhang, Aaron Courville, Longbo Huang, and Yoshua Bengio. Generative augmented flow networks. *International Conference on Learning Representations (ICLR)*, 2023.

[37] Ling Pan, Dinghuai Zhang, Moksh Jain, Longbo Huang, and Yoshua Bengio. Stochastic generative flow networks. *Uncertainty in Artificial Intelligence (UAI)*, 2023.

[38] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *International Conference on Learning Representations*, 2023.

[39] Mihir Prabhudesai, Anirudh Goyal, Deepak Pathak, and Katerina Fragkiadaki. Aligning text-to-image diffusion models with reward backpropagation. *arXiv preprint arXiv:2310.03739*, 2023.

[40] Zeju Qiu, Weiyang Liu, Haiwen Feng, Yuxuan Xue, Yao Feng, Zhen Liu, Dan Zhang, Adrian Weller, and Bernhard Schölkopf. Controlling text-to-image diffusion by orthogonal finetuning. In *Advances in Neural Information Processing Systems*, 2023.

[41] 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*, 2022.

[42] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. URL `https://arxiv.org/abs/1505.04597`.

[43] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. In *Advances in neural information processing systems*, 2022.

[44] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *ICML*, 2015.

[45] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[46] Max W. Shen, Emmanuel Bengio, Ehsan Hajiramezanali, Andreas Loukas, Kyunghyun Cho, and Tommaso Biancalani. Towards understanding and improving gflownet training. *ArXiv*, abs/2305.07170, 2023. URL `https://api.semanticscholar.org/CorpusID:258676487`.

[47] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, L. Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Vedavyas Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy P. Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–489, 2016. URL `https://api.semanticscholar.org/CorpusID:515925`.

[48] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

[49] Jiaming Song, Qinsheng Zhang, Hongxu Yin, Morteza Mardani, Ming-Yu Liu, Jan Kautz, Yongxin Chen, and Arash Vahdat. Loss-guided diffusion models for plug-and-play controllable generation. In *International Conference on Machine Learning*, 2023.

[50] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[51] Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. Learning to summarize from human feedback, 2022. URL `https://arxiv.org/abs/2009.01325`.

[52] Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988.

[53] Masatoshi Uehara, Yulai Zhao, Kevin Black, Ehsan Hajiramezanali, Gabriele Scalia, Nathaniel Lee Diamant, Alex M Tseng, Tommaso Biancalani, and Sergey Levine. Fine-tuning of continuous-time diffusion models as entropy-regularized control. *arXiv preprint arXiv:2402.15194*, 2024.

[54] Hado van Hasselt, Yotam Doron, Florian Strub, Matteo Hessel, Nicolas Sonnerat, and Joseph Modayil. Deep reinforcement learning and the deadly triad, 2018. URL `https://arxiv.org/abs/1812.02648`.

[55] Siddarth Venkatraman, Moksh Jain, Luca Scimeca, Minsu Kim, Marcin Sendera, Mohsin Hasan, Luke Rowe, Sarthak Mittal, Pablo Lemos, Emmanuel Bengio, et al. Amortizing intractable inference in diffusion models for vision, language, and control. *arXiv preprint arXiv:2405.20971*, 2024.

[56] Luhuan Wu, Brian Trippe, Christian Naesseth, David Blei, and John P Cunningham. Practical and asymptotically exact conditional sampling in diffusion models. In *Advances in Neural Information Processing Systems*, 2024.

[57] 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.

[58] Xiaoshi Wu, Keqiang Sun, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score: Better aligning text-to-image models with human preference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[59] 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. *ArXiv*, abs/2304.05977, 2023.

[60] 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. In *Advances in Neural Information Processing Systems*, 2024.

[61] Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. Geodiff: A geometric diffusion model for molecular conformation generation. In *International Conference on Learning Representations*, 2022.

[62] David W. Zhang, Corrado Rainone, Markus F. Peschl, and Roberto Bondesan. Robust scheduling with gflownets. *ArXiv*, abs/2302.05446, 2023. URL `https://api.semanticscholar.org/CorpusID:256827133`.

[63] Dinghuai Zhang, Ricky T. Q. Chen, Nikolay Malkin, and Yoshua Bengio. Unifying generative models with GFlowNets and beyond. *arXiv preprint arXiv:2209.02606v2*, 2022.

[64] Dinghuai Zhang, Nikolay Malkin, Zhen Liu, Alexandra Volokhova, Aaron Courville, and Yoshua Bengio. Generative flow networks for discrete probabilistic modeling. In *International Conference on Machine Learning*, 2022.

[65] Dinghuai Zhang, Hanjun Dai, Nikolay Malkin, Aaron C. Courville, Yoshua Bengio, and Ling Pan. Let the flows tell: Solving graph combinatorial optimization problems with gflownets. *ArXiv*, abs/2305.17010, 2023.

[66] Dinghuai Zhang, L. Pan, Ricky T. Q. Chen, Aaron C. Courville, and Yoshua Bengio. Distributional gflownets with quantile flows. *arXiv preprint arXiv:2302.05793*, 2023.

[67] Dinghuai Zhang, Ricky T. Q. Chen, Cheng-Hao Liu, Aaron Courville, and Yoshua Bengio. Diffusion generative flow samplers: Improving learning signals through partial trajectory optimization, 2024.

[68] Dinghuai Zhang, Yizhe Zhang, Jiatao Gu, Ruixiang Zhang, Josh Susskind, Navdeep Jaitly, and Shuangfei Zhai. Improving gflownets for text-to-image diffusion alignment. *arXiv preprint arXiv:2406.00633*, 2024.

[69] Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. Clay: A controllable large-scale generative model for creating high-quality 3d assets. *ACM Transactions on Graphics (TOG)*, 43(4):1–20, 2024.

[70] Stephen Zhao, Rob Brekelmans, Alireza Makhzani, and Roger Baker Grosse. Probabilistic inference in language models via twisted sequential monte carlo. In *International Conference on Machine Learning*, 2024.

[71] Mingyang Zhou, Zichao Yan, Elliot Layne, Nikolay Malkin, Dinghuai Zhang, Moksh Jain, Mathieu Blanchette, and Yoshua Bengio. Phylogfn: Phylogenetic inference with generative flow networks, 2024.

[72] Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences, 2020. URL `https://arxiv.org/abs/1909.08593`.

Appendix

Table of Contents

A Algorithm **17**

B Algorithmic details **17**
B.1 ∇-DB objective as a statistical divergence 17
B.2 Proof of Proposition 1 17
B.3 Proof of Proposition 4 18
B.4 Relationship between residual DB and Trajectory Balance 19

C Corrections for non-ideal pretrained models **19**

D Finetuning Convergence in Wall Time **20**

E Figures for ablation results **21**

F Additional ablation experiments **24**

G More samples (Aesthetic Score) **26**

H More samples (HPSv2) **27**

A ALGORITHM

Algorithm 1 Diffusion Finetuning with *residual* ∇-DB

1: **Inputs:** Pretrained diffusion model $f_{\theta\#}$, reward function $R(\cdot)$.
2: **Initialization:** Model to finetune f_θ with $\theta = \theta^\#$, residual flow score function $g_\phi(\cdot)$.
3: Sample the initial batch of trajectories $\mathcal{D}_{\text{prev}} = \{(x_1, ..., x_T)_i\}_{i=1...N}$ with the current finetuned diffusion model f_θ.
4: **while** not converged **do**
5: Sample a batch of trajectories $\mathcal{D}_{\text{curr}} = \{(x_1, ..., x_T)_i\}_{i=1...N}$ with the finetuned diffusion model.
6: Subsample the time steps to train with: the full set $\mathcal{T}_i = \{1, ..., T\}$ or the sampled set $\mathcal{T}_i = \text{Sample-N}(\{1, ..., T\})$.
7: Compute the loss $\sum_{t\in\mathcal{T}_i,(x_{1:T})_i\in\mathcal{D}_{\text{prev}}} L_{\nabla\text{DB-FL-res}}(x_t, x_{t+1}; \theta, \theta^\#, \phi) + L_{\nabla\text{DB-FL-terminal}}(x_T; \phi) + \lambda\|f_\theta(x_t) - f_{\theta\#}(x_t)\|^2$.
8: Update the diffusion model and the residual flow score function.
9: Set $\mathcal{D}_{\text{prev}} \leftarrow \mathcal{D}_{\text{curr}}$.
10: **end while**
11: **return** finetuned model f_θ.

B ALGORITHMIC DETAILS

B.1 ∇-DB OBJECTIVE AS A STATISTICAL DIVERGENCE

$L_{\nabla\text{DB}}$ (Equation 7) is analogous to a Fisher divergence (up to a constant scale) if we always use on-policy samples to update the diffusion model and the flow function:

$$\begin{aligned}
&D_{\text{Fisher}}\Big(P_F(x_{t+1}|x_t)\Big|\Big|\frac{P_B(x_t|x_{t+1})F(x_{t+1})}{F(x_t)}\Big) \\
&= \frac{1}{2}\mathop{\mathbb{E}}_{x_{t+1}\sim P_F(x_{t+1}|x_t)}\Big|\Big|\nabla_{x_{t+1}}\log P_B(x_t|x_{t+1}) - \nabla_{x_{t+1}}\log\frac{P_B(x_t|x_{t+1})F(x_{t+1})}{F(x_t)}\Big|\Big|^2 \\
&= \frac{1}{2}\mathop{\mathbb{E}}_{x_{t+1}\sim P_F(x_{t+1}|x_t)} L_{\nabla\text{DB}}(x_t, x_{t+1}).
\end{aligned} \tag{21}$$

B.2 PROOF OF PROPOSITION 1

Proof. When the training objectives equal 0 for all states, we would have

$$\nabla_{x_{t+1}}\log P_F(x_{t+1}|x_t) = \nabla_{x_{t+1}}\log P_B(x_t|x_{t+1}) + \nabla_{x_{t+1}}\log F_{t+1}(x_{t+1}) \tag{22}$$

$$\nabla_{x_t}\log P_F(x_{t+1}|x_t) = \nabla_{x_t}\log P_B(x_t|x_{t+1}) - \nabla_{x_t}\log F_t(x_t) \tag{23}$$

$$\nabla_{x_T}\log F(x_T) = \beta\nabla_{x_T}\log R(x_T) \tag{24}$$

for any trajectory $(x_0, \ldots, x_T)$.

Through indefinite integral, these indicate that there exist a function $C_t(x_t)$ satisfies

$$C_t(x_t)P_F(x_{t+1}|x_t) = F_{t+1}(x_{t+1})P_B(x_t|x_{t+1}) \tag{25}$$

$$F_t(x_t)P_F(x_{t+1}|x_t) = C_{t+1}(x_{t+1})P_B(x_t|x_{t+1}) \tag{26}$$

$$F(x_T) \propto R(x_T)^\beta. \tag{27}$$

Therefore, we have

$$\frac{C_t(x_t)}{F_t(x_t)} = \frac{F_{t+1}(x_{t+1})}{C_{t+1}(x_{t+1})}, \quad \forall(x_t, x_{t+1}). \tag{28}$$

The right hand side does not depend on x_t, therefore, the left hand side is a constant. So we have

$$C_t(x_t) \propto F_t(x_t), \quad \forall t. \tag{29}$$

The probability of generating a data x_T then equals

$$P_F(x_T) = \int P_0(x_0|\oslash) \prod_t P_F(x_{t+1}|x_t) dx_{0:T-1} \tag{30}$$

$$= \int F_0(x_0) \prod_t \frac{F_{t+1}(x_{t+1}) P_B(x_t|x_{t+1})}{C_t(x_t)} dx_{0:T-1} \tag{31}$$

$$\propto \int F_0(x_0) \prod_t \frac{F_{t+1}(x_{t+1}) P_B(x_t|x_{t+1})}{F_t(x_t)} dx_{0:T-1} \tag{32}$$

$$\propto F(x_T) \int \prod_t P_B(x_t|x_{t+1}) dx_{0:T-1} \tag{33}$$

$$\propto F(x_T) \propto R(x_T)^\beta, \tag{34}$$

which proves the validness of the ∇-GFlowNet algorithm. □

B.3 Proof of Proposition 4

When the training objectives equal 0 for all states, we have

$$\nabla_{x_{t+1}} \log \tilde{P}_F(x_{t+1}|x_t) = \nabla_{x_{t+1}} \log \tilde{F}(x_{t+1}) \tag{35}$$

$$\nabla_{x_t} \log \tilde{P}_F(x_{t+1}|x_t) = -\nabla_{x_t} \log \tilde{F}(x_t) \tag{36}$$

$$\nabla_{x_T} \log \tilde{F}(x_T) = \beta \nabla_{x_T} \log R(x_T) \tag{37}$$

for any trajectory $(x_0, \ldots, x_T)$.

Through indefinite integral, these indicate that there exist a function $C_t(x_t)$ satisfies

$$C_t(x_t) \tilde{P}_F(x_{t+1}|x_t) = \tilde{F}(x_{t+1}) \tag{38}$$

$$\tilde{F}(x_t) \tilde{P}_F(x_{t+1}|x_t) = C_{t+1}(x_{t+1}) \tag{39}$$

$$\tilde{F}(x_T) \propto R(x_T)^\beta. \tag{40}$$

Thus we have

$$\frac{C_t(x_t)}{\tilde{F}_t(x_t)} = \frac{\tilde{F}_{t+1}(x_{t+1})}{C_{t+1}(x_{t+1})}, \quad \forall (x_t, x_{t+1}). \tag{41}$$

The right hand side does not depend on x_t, therefore, the left hand side is a constant. So we have

$$C_t(x_t) \propto \tilde{F}_t(x_t), \quad \forall t. \tag{42}$$

The probability of generating a data x_T then equals

$$P_F(x_T) = \int P_0(x_0|\oslash) \prod_t P_F(x_{t+1}|x_t) dx_{0:T-1} \tag{43}$$

$$= \int P_0^\#(x_0|\oslash) \prod_t P_F^\#(x_{t+1}|x_t) \prod_t \tilde{P}_F(x_{t+1}|x_t) dx_{0:T-1} \tag{44}$$

$$= \int P_0^\#(x_0|\oslash) \prod_t P_F^\#(x_{t+1}|x_t) \frac{\tilde{F}_{t+1}(x_{t+1})}{C_t(x_t)} dx_{0:T-1} \tag{45}$$

$$\propto \tilde{F}_T(x_T) \int P_0^\#(x_0|\oslash) \prod_t P_F^\#(x_{t+1}|x_t) dx_{0:T-1} \tag{46}$$

$$\propto R(x_T)^\beta P_F^\#(x_T). \tag{47}$$

This completes the validness proof.

B.4 Relationship between residual DB and Trajectory Balance

Here we illustrate the Remark 5. A different but equivalent condition for GFlowNets states:

Trajectory Balance (TB) [31]. The following TB condition must hold for any transition sequence $(s_0, s_1, ..., s_N)$ where x_0 is the unique starting state in the MDP and s_N is a terminal state, given a GFlowNet with the forward policy $P_F(s'|s)$ and the backward policy $P_B(s|s')$:

$$\log \frac{Z \prod P_F(s'|s)}{R(x_T) \prod P_B(s|s')} = 0 \tag{48}$$

where $Z = F(x_0)$ is the total flow and $R(x_T) = F(x_T)$ the reward. The proof is immediate with a telescoping product of the DB condition.

With an (ideal) pretrained model $P_F^{\#}$ and the satisfication of the finetuning objective of Equation 10, one can prove the conclusion in [55]:

$$\log \frac{Z \prod P_F(s'|s)}{Z^{\#} \prod P_F^{\#}(s'|s)} = \beta \log R(x_T), \tag{49}$$

which is also an immediate result of a telescoping products of the *residual* DB condition (which leads to Equation 20):

$$\log \tilde{P}_F(x_{t+1}|x_t) = \log \tilde{F}(x_{t+1}) - \log \tilde{F}(x_t). \tag{50}$$

While Equation 49 and residual DB are mathematically equivalent, implementation-wise TB in Equation 49 demands the whole sampling sequence be stored in the memory for gradient computation, or one has resort to the time-costly technique of gradient checkpointing. In comparison, with DB-based methods one may amortize the computational cost into flows at different time steps and therefore allow diffusion finetuning with flexible sampling sequence, of which the distribution approximation capacity and generation performance are generally greater.

C Corrections for non-ideal pretrained models

For non-ideal pretrained models in which $P_F^{\#}$ does not match P_B (*i.e.*, the DB condition is violated), the original *residual* ∇-DB objective is apparently biased. We may introduce an additional learnable term $h(x_t, x_{t+1}; \psi)$ to compensate for this error, with which we obtain:

$$L_{\overrightarrow{\nabla}\text{DB-res-v2}}(x_t, x_{t+1}) = \left\| \nabla_{x_{t+1}} \log \tilde{P}_F(x_{t+1}|x_t) - \nabla_{x_{t+1}} \log \tilde{F}(x_{t+1}) + \nabla_{x_{t+1}} h(x_t, x_{t+1}; \psi). \right\|^2 \tag{51}$$

and

$$L_{\overleftarrow{\nabla}\text{DB-res-v2}}(x_t, x_{t+1}) = \left\| \nabla_{x_t} \log \tilde{P}_F(x_{t+1}|x_t) + \nabla_{x_t} \log \tilde{F}(x_{t+1}) + \nabla_{x_t} h(x_t, x_{t+1}; \psi). \right\|^2. \tag{52}$$

D FINETUNING CONVERGENCE IN WALL TIME

We further show the convergence speed measured in relative wall time on a single node with 8 80GB-mem A100 GPUs in Fig. 9.

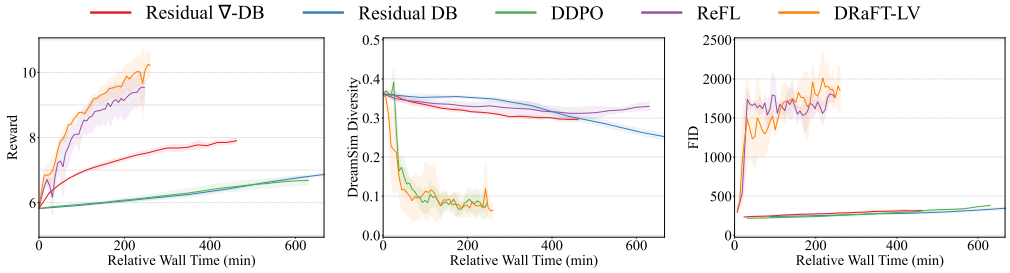

Figure 9: Convergence curves of different metrics for different methods throughout the finetuning process on Aesthetic Score, with the x-axis being the relative wall time. All methods are benchmarked on a single node with 8 80GB-mem A100 GPUs.

E FIGURES FOR ABLATION RESULTS

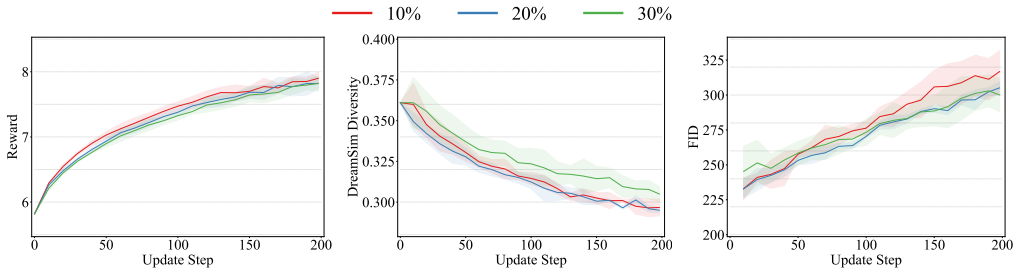

Figure 10: Ablation study on the effect of subsampling rate on the collected trajectories for computing the *residual* ∇-DB loss.

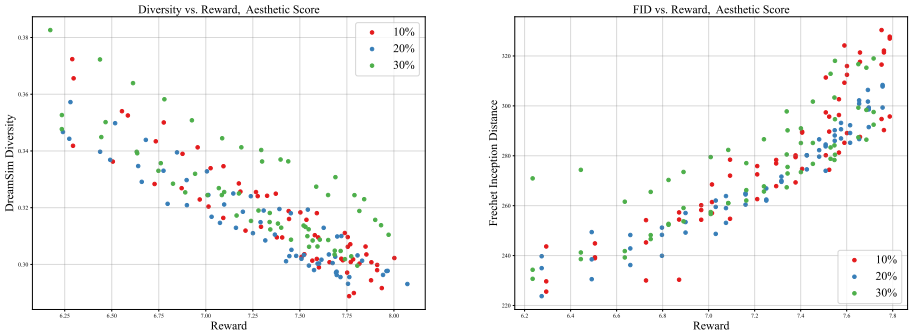

Figure 11: Pareto frontiers for reward, diversity and prior-following (measured by FID) of models trained with different subsampling rate. In expectation, higher subsampling rates seem to slightly help in increasing diversity.

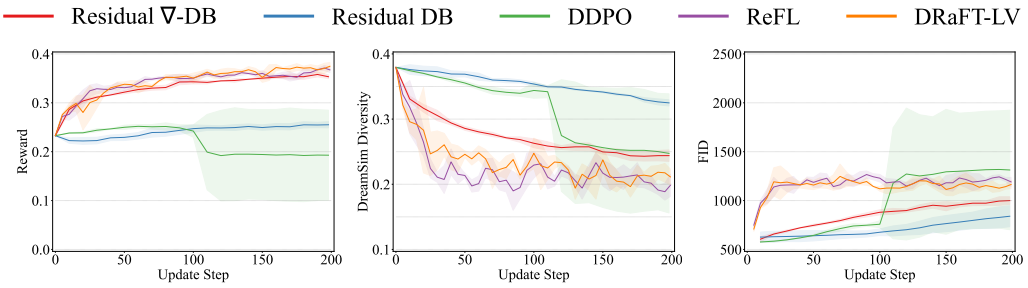

Figure 12: Convergence curve of metrics of different methods throughout the finetuning process on the HPSv2 reward model.

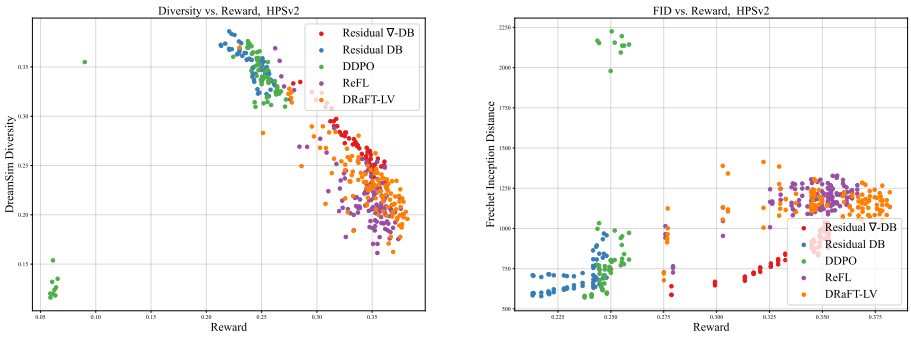

Figure 13: Pareto frontiers for reward, diversity and prior-following (measured by FID) on HPSv2.

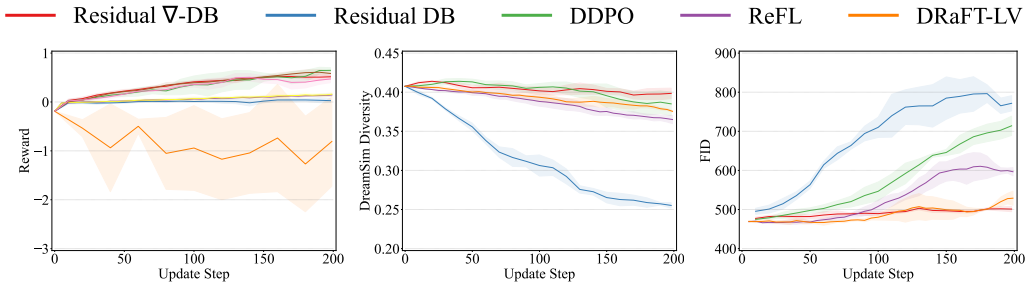

Figure 14: Convergence curve of metrics of different methods throughout the finetuning process on the ImageReward reward model.

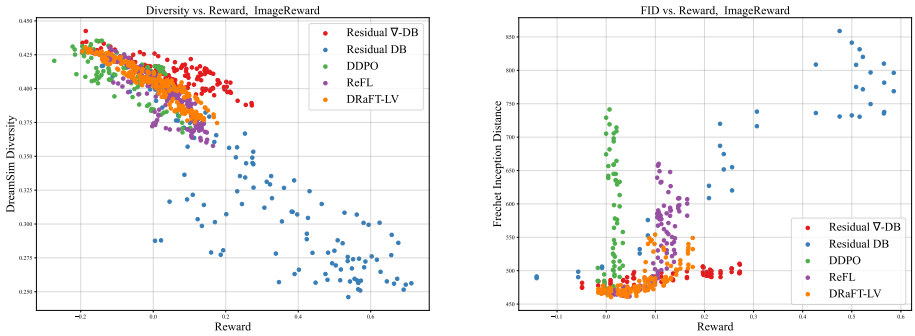

Figure 15: Pareto frontiers for reward, diversity and prior-following (measured by FID) on ImageReward.

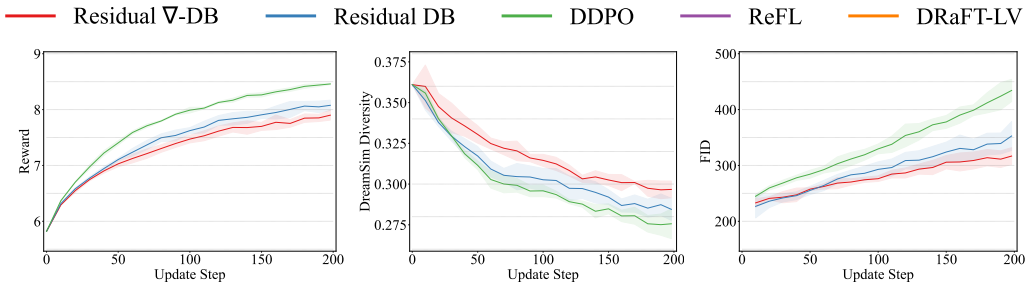

Figure 16: Convergence curve of metrics of different methods throughout the finetuning process on Aesthetic Score with different strength of prior η.

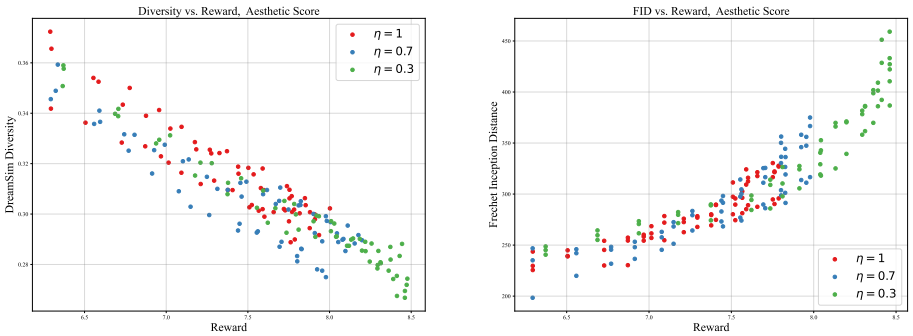

Figure 17: Pareto frontiers for reward, diversity and prior-following (measured by FID) on Aesthetic Score with different strength of prior η.

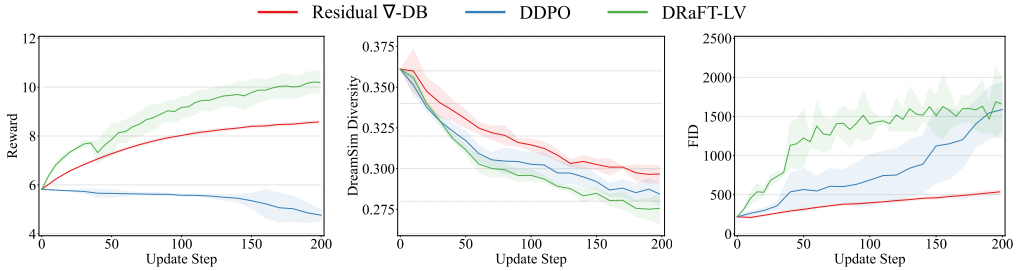

Figure 18: Convergence curve of metrics of different methods throughout the finetuning process on Aesthetic Score with the MDP constructed by SDE-DPM-Solver++ (with 20 inference steps).

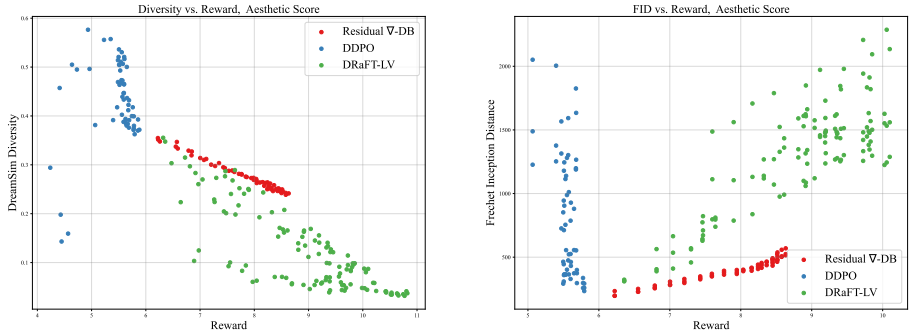

Figure 19: Pareto frontiers for reward, diversity and prior-following (measured by FID) on Aesthetic Score with the MDP constructed by SDE-DPM-Solver++ (with 20 inference steps).

F ADDITIONAL ABLATION EXPERIMENTS

Effect of attenuating scaling on predicted reward. In Fig. 20 and 21, we show the comparison between models with and without time-dependent attenuation of reward signals, where both models are trained with $\beta = 10000$. While setting $\gamma_t = 1$ for all t can slightly increase the convergence speed, it comes at the cost of worse diversity and prior-following capability.'

Effect of 2nd-order gradients in finetuning. In Fig. 22 and 23, we show the comparison between models with and without 2nd-order gradients, where both models are trained with $\beta = 10000$. Empirically, 2nd-order gradients hurts the trade-off between reward convergence, diversity preservation and prior following.

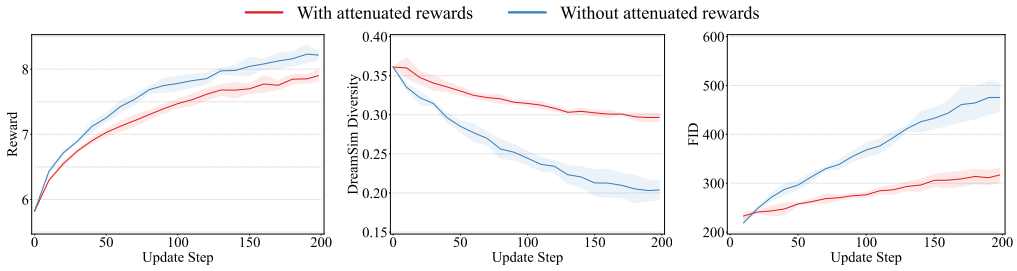

Figure 20: Convergence curve of metrics of different methods throughout the finetuning process on Aesthetic Score with time-dependent attenuation of predicted rewards. Both models are trained with $\beta = 10000$. With decayed predicted rewards, the convergence speed is slower but due to less aggressive prediction on reward signal, the model with reward attenuation achives better diversity and prior-following results.

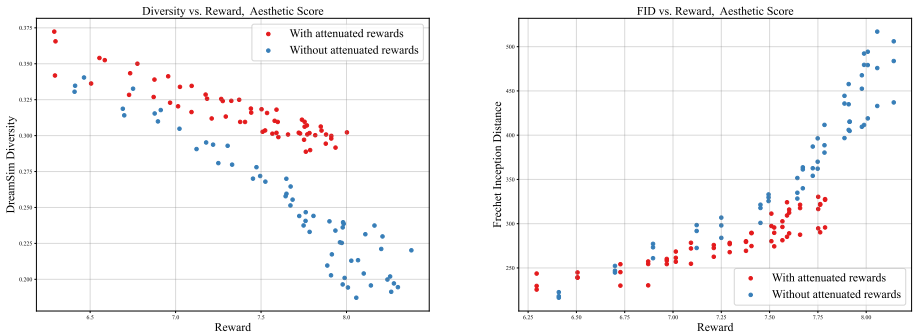

Figure 21: Pareto frontiers for reward, diversity and prior-following (measured by FID) on Aesthetic Score with time-dependent scaling of predicted rewards.

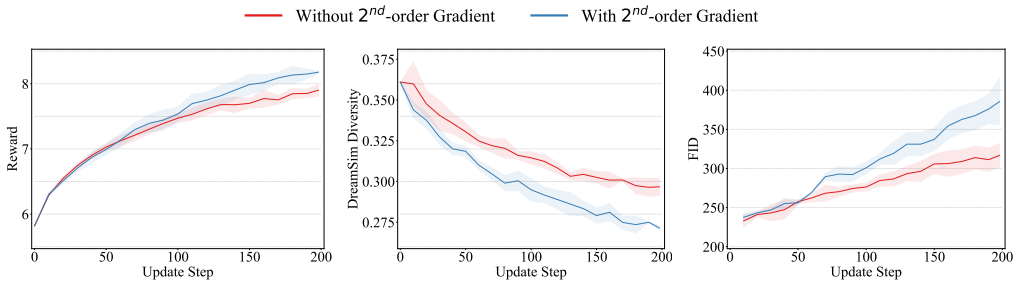

Figure 22: Convergence curve of metrics of different methods throughout the finetuning process on Aesthetic Score with and without 2nd-order gradients.

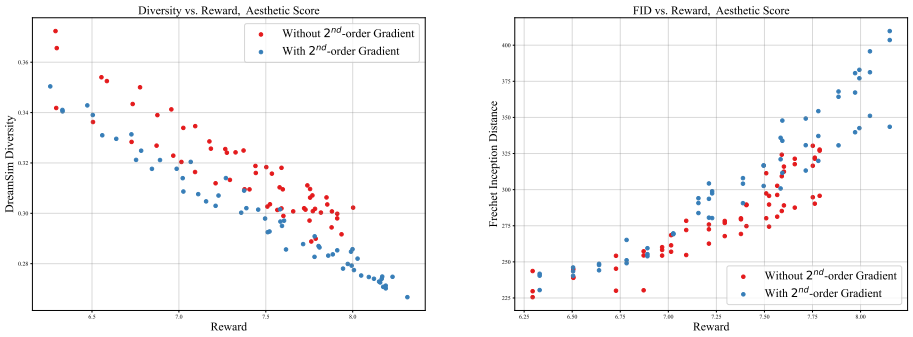

Figure 23: Pareto frontiers for reward, diversity and prior-following (measured by FID) on Aesthetic Score with and without 2nd-order gradients.

G MORE SAMPLES (AESTHETIC SCORE)

Figure 24: Additional uncurated samples from the model finetuned with *residual* ∇-DB on the reward model of Aesthetic Score.

H MORE SAMPLES (HPSV2)

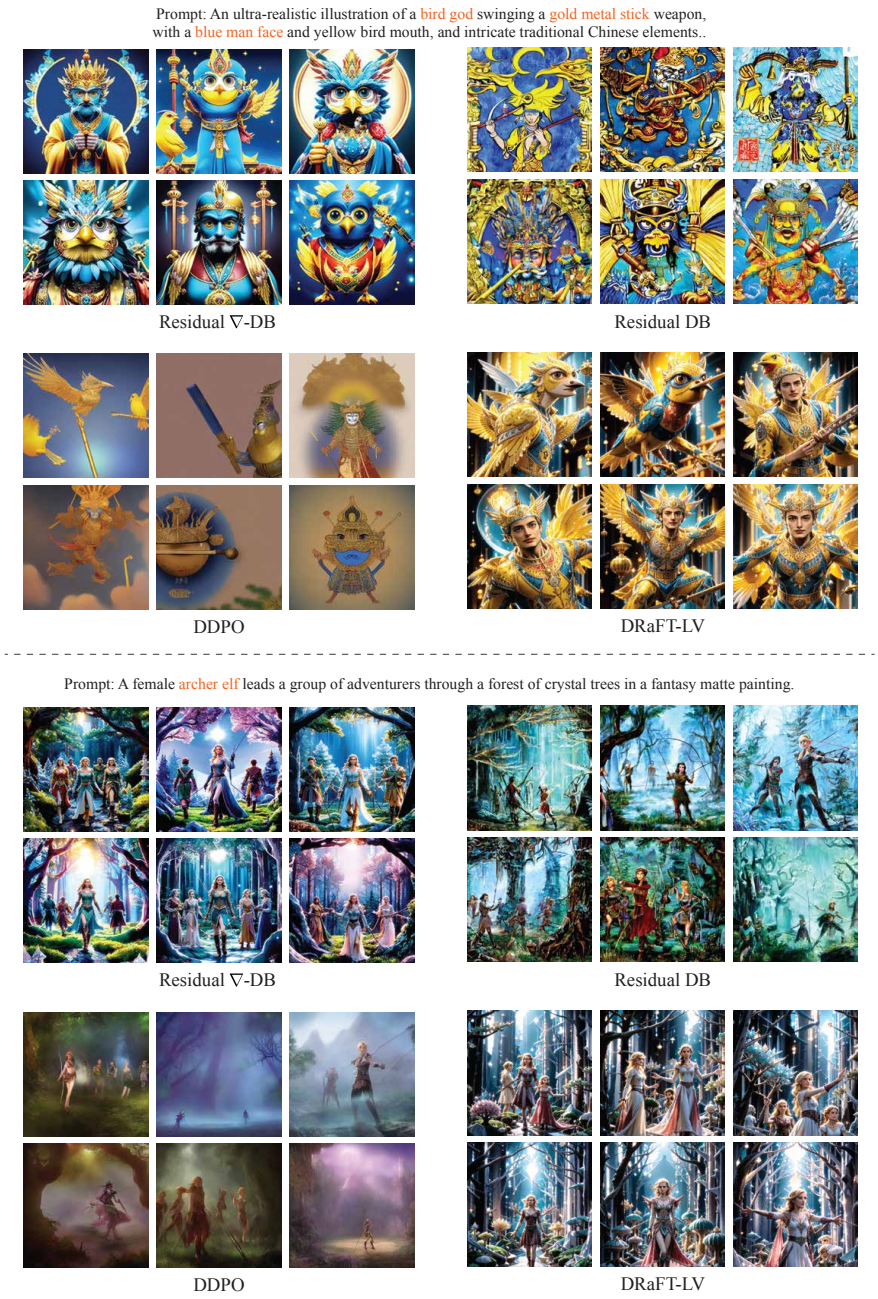

Figure 25: More comparison between samples generated by *residual* ∇-DB and the baseline methods. The model finetuned with *residual* ∇-DB is capable of following the instructions while generating diverse samples.

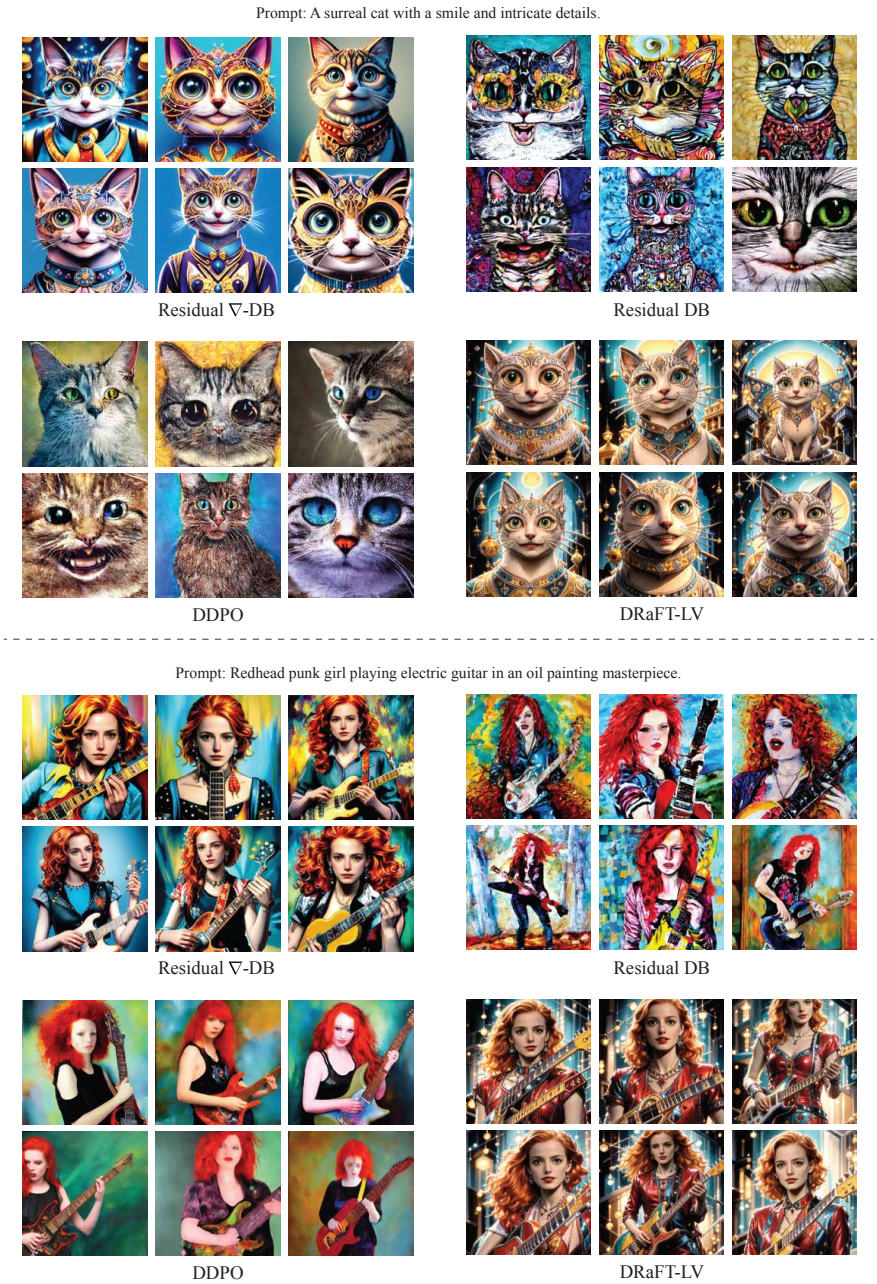

Figure 26: HPSv2 samples, Continued.