BOOT👢: Data-free Distillation of Denoising Diffusion Models with Bootstrapping

Jiatao Gu [1] **Shuangfei Zhai** [1] **Yizhe Zhang** [1] **Lingjie Liu** [2] **Josh Susskind** [1]

Abstract

Diffusion models have demonstrated excellent potential for generating diverse images. However, their performance often suffers from slow generation due to iterative denoising. Existing distillation methods either require significant amounts of offline computation for generating synthetic training data, or need to perform expensive online learning with the help of real data. In this work, we present a novel technique called *BOOT*, that overcomes these limitations with an efficient data-free distillation algorithm. The core idea is to learn a time-conditioned model that predicts the output of a pre-trained diffusion model teacher given any time-step. Such a model can be efficiently trained based on bootstrapping from two consecutive sampled steps. Furthermore, our method can be easily adapted to large-scale text-to-image diffusion models, which are challenging for conventional methods given the fact that the training sets are often large and difficult to access. We demonstrate the effectiveness of our approach on several benchmarks, achieving comparable generation quality while being orders of magnitude faster than the diffusion teacher. The text-to-image results show that BOOT is able to handle highly complex distributions, shedding light on efficient generative modeling. Please check our project page: https://jiataogu.me/boot/ for more results.

1. Introduction

Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Nichol & Dhariwal, 2021; Song et al., 2020b) have become the *de-facto* tools for various generative applications, including images (Dhariwal & Nichol, 2021; Rombach et al., 2021; Ramesh et al., 2022; Saharia et al., 2022), videos (Ho

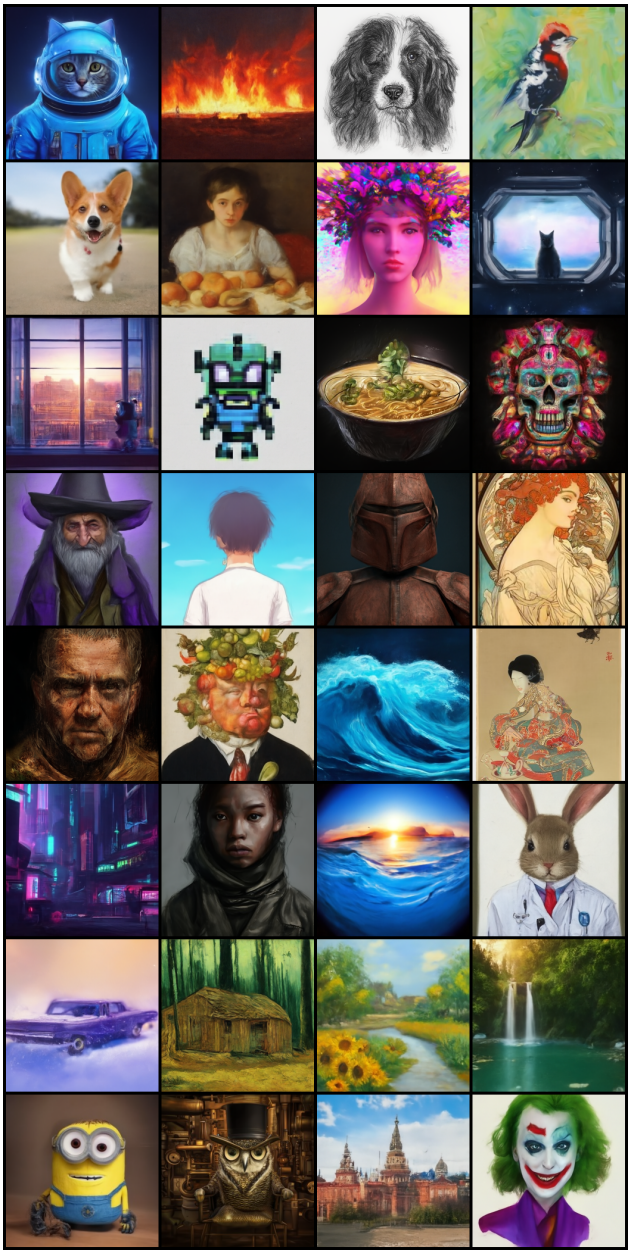


Figure 1. Curated samples of our distilled **single-step** model.

[1]Apple [2]University of Pennsylvania. Correspondence to: Jiatao Gu <jgu32@apple.com>.

Accepted to ICML workshop on Structured Probabilistic Inference & Generative Modeling (SPIGM), Honolulu, Hawaii, USA. 2023.

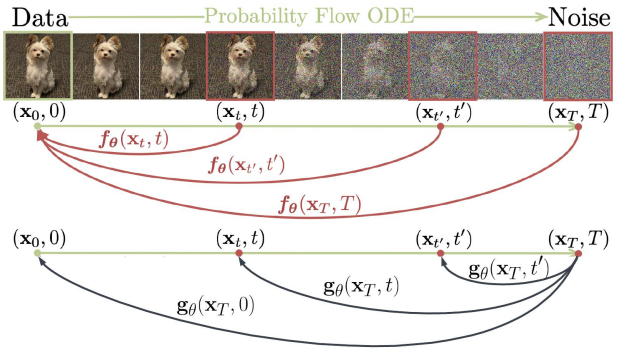


Figure 2. Comparison of Consistency Model (Song et al., 2023) (red ↑) and BOOT (black ↓) highlighting the opposing pathways.

et al., 2022b;a), 3D (Poole et al., 2022; Watson et al., 2022; Gu et al., 2023), audio (Liu et al., 2023) and text (Li et al., 2022) generation. Compared to many alternative generative approaches such as GANs (Goodfellow et al., 2014a; Karras et al., 2021) or VAEs (Kingma & Welling, 2013), diffusion models are arguably much more stable to train without the need of balancing two modules, and thus less prone to issues such as mode collapse or posterior collapse. Despite the empirical success, inference from standard diffusion models are often time-consuming ($50 \sim 1000\times$ slower than single-step models like GANs), causing challenges to deploy on consumer devices. The reason is that diffusion models, by design, employ an iterative sampling process.

Previous studies proposed using knowledge distillation to improve the inference speed (Hinton et al., 2015). Specifically, a faster student model can be trained to replicate the output of a pre-trained diffusion model. In this work, we focus on learning single-step models that require only one neural function evaluation (NFE). However, conventional methods, such as (Luhman & Luhman, 2021), necessitate executing the full teacher sampling to generate synthetic targets for every student update, which is not practical for distilling large diffusion models, such as StableDiffusion (SD, Rombach et al., 2021). Recently, several techniques have been proposed to avoid sampling using the concept of "bootstrap". For example, (Salimans & Ho, 2022) gradually halves the inference steps based on the student of the previous stage, while (Song et al., 2023) and (Berthelot et al., 2023) train single-step denoisers by enforcing self-consistency between adjacent student outputs along the same diffusion trajectory. However, these approaches rely on the availability of real data to simulate the intermediate diffusion states as input. This exposes significant limitations when applying these methods to scenarios where the desired real data is not accessible or very large.

In this paper, we present *BOOT*, a data-free distillation method for denoising diffusion models based on bootstrapping. BOOT is partially motivated by the observation made by (Song et al., 2023) that all points on the same diffusion trajectory (also known as PF-ODE (Song et al., 2020b)) have a deterministic mapping between each other. Different from (Song et al., 2023) that sought self-consistency from any $\boldsymbol{x}_t$ to $\boldsymbol{x}_0$, BOOT predicts all possible $\boldsymbol{x}_t$ given the same noise point $\boldsymbol{x}_T$ and a time-indicator t. Since our model $\boldsymbol{g}_\theta$ always reads pure Gaussian noise, it eliminates the need to sample from real data. Learning all $\boldsymbol{x}_t$ from the same $\boldsymbol{x}_T$ enables bootstrapping: it is easier to predict $\boldsymbol{x}_t$ if the model has already learned to generate $\boldsymbol{x}_{t'}$ where $t' > t$. Formulating the bootstrapping in this way poses additional challenges, such as noisy sample prediction, which is non-trivial for neural networks. To tackle this, we learn the student model from a novel *Signal-ODE* which is derived from the original PF-ODE. We further design the objectives and boundary conditions to enhance the sampling quality. In this way, we enable efficient inference of large diffusion models in scenarios where the original training corpus is inaccessible due to privacy or other concerns.

In the experiments, we first demonstrate the efficacy of BOOT on various challenging image generation benchmarks, including unconditional and class-conditional settings. Next, we show that the proposed method can be easily adopted to distill text-to-image diffusion models.

2. Method

2.1. Signal-ODE

We consider a time-conditioned generative model $\boldsymbol{g}_\theta(\boldsymbol{\epsilon}, t)$, which takes random noise $\boldsymbol{\epsilon}$ as input, and approximates the intermediate output of a pretrained diffusion model: $\boldsymbol{g}_\theta(\boldsymbol{\epsilon}, t) \approx \texttt{ODE-Solver}(\boldsymbol{f}_\phi, \boldsymbol{\epsilon}, T \to t)$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$ and $\boldsymbol{f}_\phi$ is the teacher denoising network. This approach eliminates the need to sample from real data during training. The final sample can be obtained as $\boldsymbol{g}_\theta(\boldsymbol{\epsilon}, 0) \approx \boldsymbol{x}_0$. However, it poses a challenge to train $\boldsymbol{g}_\theta$ effectively, as neural networks struggle to predict partially noisy images (Berthelot et al., 2023), leading to out-of-distribution (OOD) problems and additional complexities in learning $\boldsymbol{g}_\theta$ accurately. To overcome the aforementioned challenge, we propose an alternative approach where we predict $\boldsymbol{y}_t = (\boldsymbol{x}_t - \sigma_t \boldsymbol{\epsilon})/\alpha_t$. In this case, $\boldsymbol{y}_t$ represents the low-frequency "signal" component of $\boldsymbol{x}_t$, which is easier for neural networks to learn. The initial noise for diffusion is denoted by $\boldsymbol{\epsilon}$. This prediction target is reasonable since it aligns with the boundary condition of the teacher model, where $\boldsymbol{y}_0 = \boldsymbol{x}_0$. Furthermore, we can derive an iterative equation from Eq. (6) for consecutive timesteps:

$$\boldsymbol{y}_s = \left(1 - e^{\lambda_s - \lambda_t}\right) \boldsymbol{f}_\phi(\boldsymbol{x}_t, t) + e^{\lambda_s - \lambda_t} \boldsymbol{y}_t, \qquad (1)$$

where $\boldsymbol{x}_t = \alpha_t \boldsymbol{y}_t + \sigma_t \boldsymbol{\epsilon}$, and $\lambda_t = -\log(\alpha_t/\sigma_t)$ represents the "negative half log-SNR." Notably, the noise term $\boldsymbol{\epsilon}$ automatically cancels out in Eq. (1), indicating that the model

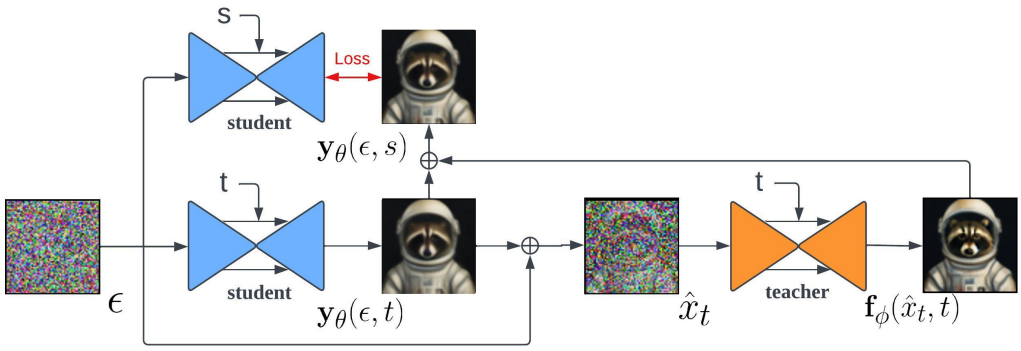


Figure 3. Training pipeline of BOOT. s and t are two consecutive timesteps where $s < t$. From a noise map $\boldsymbol{\epsilon}$, the objective of BOOT minimizes the difference between the output of a student model at timestep s, and the output of stacking the same student model and a teacher model at an earlier time t. **The whole process is data-free**.

always learns from the signal space. We can further obtain a continuous version of Eq. (1) by letting $s \to t^-$ as follows:

$$\frac{\mathrm{d}\boldsymbol{y}_t}{\mathrm{d}t} = -\lambda_t' \cdot (\boldsymbol{f}_\phi(\boldsymbol{x}_t, t) - \boldsymbol{y}_t), \quad \boldsymbol{y}_T \sim p_{\boldsymbol{\epsilon}} \tag{2}$$

where $\lambda_t' = \mathrm{d}\lambda/\mathrm{d}t$, and $p_{\boldsymbol{\epsilon}}$ epresents the boundary distribution of $\boldsymbol{y}_t$. It's important to note that Eq. (2) differs from the PF-ODE, which directly relates to the score function of the data. In our case, the ODE, which we refer to as "Signal-ODE," is specifically defined for signal prediction. At each timestep t, a fixed noise $\boldsymbol{\epsilon}$ is injected and denoised by the diffusion model $\boldsymbol{f}_\phi$. The Signal-ODE implies a "ground-truth" trajectory for sampling new data. For example, one can initialize a reasonable $\boldsymbol{y}_T = \boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$ and solve the Signal-ODE to obtain the final output $\boldsymbol{y}_0$. Although the computational complexity remains the same as conventional DDIM, we will demonstrate in the next section how we can efficiently approximate $\boldsymbol{y}_t$ using bootstrapping objectives.

2.2. Learning with Bootstrapping

Our objective is to learn $\boldsymbol{y}_\theta(\boldsymbol{\epsilon}, t) \approx \boldsymbol{y}_t$ as a single-step prediction model using neural networks, rather than solving the signal-ODE with Eq. (2). By matching both sides of Eq. (2), we can readily obtain the loss function:

$$\mathcal{L}_\theta^{\mathrm{DE}} = \mathbb{E}_{\boldsymbol{\epsilon},t}\left[\left\|\frac{\mathrm{d}\boldsymbol{y}_\theta(\boldsymbol{\epsilon}, t)}{\mathrm{d}t} + \lambda_t' \cdot (\boldsymbol{f}_\phi(\hat{\boldsymbol{x}}_t, t) - \boldsymbol{y}_\theta(\boldsymbol{\epsilon}, t))\right\|_2^2\right]. \tag{3}$$

In Eq. (3), we use $\boldsymbol{y}_\theta(\boldsymbol{\epsilon}, t)$ to estimate $\boldsymbol{y}_t$, and $\hat{\boldsymbol{x}}_t = \alpha_t \boldsymbol{y}_\theta(\boldsymbol{\epsilon}, t) + \sigma_t \boldsymbol{\epsilon}$ represents the corresponding noisy image. Instead of using forward-mode auto-differentiation, which can be computationally expensive, we can approximate the above equation with finite difference due to the 1-dimensional nature of t:

$$\frac{\mathrm{d}\boldsymbol{y}_\theta(\boldsymbol{\epsilon}, t)}{\mathrm{d}t} \approx \frac{\boldsymbol{y}_\theta(\boldsymbol{\epsilon}, t) - \boldsymbol{y}_\theta(\boldsymbol{\epsilon}, s)}{\delta} \tag{4}$$

where $s = t - \delta$ and δ is the discrete step size. In practise, we apply stop-gradient operator over $\boldsymbol{f}_\phi$ for training stability.

Different from CM-based methods (Song et al., 2023; Berthelot et al., 2023), we do not require an exponential moving average (EMA) copy of the student parameters to avoid collapsing. This avoids potential slow convergence and sub-optimal solutions. The proposed objective is unlikely to degenerate because there is an incremental improvement term in the training target, which is mostly non-zero. In other words, we can consider $\boldsymbol{y}_\theta$ as an exponential moving average of $\boldsymbol{f}_\phi$, with a decaying rate of $1 - \delta\lambda_t'$. This ensures that the student model always receives distinguishable signals for different values of t.

Error Accumulation A critical challenge in learning BOOT is the "error accumulation" issue, where imperfect predictions of $\boldsymbol{y}_\theta$ on large t can propagate to subsequent timesteps. While similar challenges exist in other bootstrapping-based approaches, it becomes more pronounced in our case due to the possibility of out-of-distribution inputs $\hat{\boldsymbol{x}}_t$ for the teacher model, resulting from error accumulation and leading to incorrect learning signals. To mitigate this, we employ two methods: (1) We uniformly sample t throughout the training time, despite the potential slowdown in convergence. (2) We use a higher-order solver (e.g., Heun's method (Ascher & Petzold, 1998)) to compute the bootstrapping target with better estimation.

Boundary Condition In theory, the boundary $\boldsymbol{y}_T$ can have arbitrary values since $\alpha_T = 0$, and the value of $\boldsymbol{y}_T$ does not affect the value $\boldsymbol{x}_T = \boldsymbol{\epsilon}$. However, λ_t' is unbounded at $t = T$, leading to numerical issues in optimization. As a result, the student model must be learned within a truncated range $t \in [t_{\min}, t_{\max}]$. This necessitates additional constraints at the boundaries to ensure that $\alpha_{t_{\max}} \boldsymbol{y}_\theta(\boldsymbol{\epsilon}, t_{\max}) + \sigma_{t_{\max}} \boldsymbol{\epsilon}$ follows the same distribution as the diffusion model. In this work, we address this through an auxiliary boundary loss:

$$\mathcal{L}_\theta^{\mathrm{BC}} = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,I)}\left[\|\boldsymbol{f}_\phi(\boldsymbol{\epsilon}, t_{\max}) - \boldsymbol{y}_\theta(\boldsymbol{\epsilon}, t_{\max})\|_2^2\right]. \tag{5}$$

Here, we enforce the student model to match the initial denoising output. In our early exploration, we found that

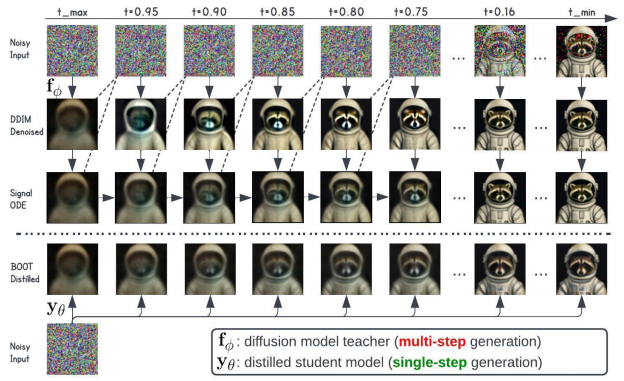


Figure 4. Comparison between the outputs of DDIM/Signal-ODE and our distilled model given the same prompt and initial noise input. By definition, signal-ODE converges to the same final sample as the original DDIM, while the distilled single-step model does not necessarily follow.

the boundary condition is crucial for the single-step student to fully capture the modeling space of the teacher, especially in text-to-image scenarios. Failure to learn the boundaries tends to result in severe mode collapse problems.

2.3. Distillation of Text-to-Image Models

Distillation with Guidance Our approach can be readily applied for distilling conditional diffusion models, such as text-to-image generation (Ramesh et al., 2022; Rombach et al., 2021; Balaji et al., 2022), where a conditional denoiser $\boldsymbol{f}_\phi(\boldsymbol{x}_t, t, \boldsymbol{c})$ is learned. Inference of these models requires necessary post-processing steps. For instance, one can perform classifier-free guidance (CFG, Ho & Salimans, 2022) to amplify the conditioning. We directly use the CFG modified output to replace the original $\boldsymbol{f}_\phi$ in the training objectives. Optionally, similar to (Meng et al., 2022), we can also learn student model condition on both t and w (CFG weight) to reflect different guidance strength.

Pixel or Latent Our method can be easily adopted in either pixel (Saharia et al., 2022) or latent space (Rombach et al., 2021) models without specific code change. Pixel-space models (Saharia et al., 2022) typically involve learning cascaded models (one base model + a few super-resolution (SR) models) to increase the output resolutions progressively. We can also distill the SR models with BOOT into one step by conditioning both the SR teacher and the student with the output of the distilled base model.

3. Experiments

3.1. Experimental Setups

Implementation Details We first validate the results of the proposed BOOT on diffusion models trained on standard image generation benchmarks: FFHQ 64×64 (Karras et al., 2017), class-conditional ImageNet 64×64 (Deng et al., 2009) and LSUN Bedroom 256×256 (Yu et al., 2015). For controlled comparison, we train all teacher diffusion models by ourselves separately on each dataset with the signal prediction objective. Moreover, for ImageNet, we also test the performance of CFG where the student models are trained with random conditioning on $w \in [1, 5]$.

For text-to-image generation scenarios, we directly apply BOOT on open-sourced diffusion models in both pixel-space (IF, Saharia et al., 2022) and latents space (SD, Rombach et al., 2021). Thanks to the data-free nature, we don't need to access the original training set. Instead, we only need the prompt conditions to distill both models. In this work, we consider general purpose prompts generated by users. Specifically, we adopt the text prompts from diffusiondb (Wang et al., 2022), a large-scale prompt dataset.

Following Song et al. (2023), student models have similar architecture as the teacher, with nearly identical number of parameters. We include additional details in the appendix.

	Steps	FFHQ FID	FFHQ fps	LSUN FID	LSUN fps	ImageNet FID	ImageNet fps
DDPM	250	5.4	0.2	8.2	0.1	11.0	0.1
DDIM	50	7.6	1.2	13.5	0.6	13.7	0.6
	10	18.3	5.3	31.0	3.1	18.3	3.3
	1	225	54	308	31	237	34
Ours	1	9.0	54	23.4	32	16.3	34

Table 1. Comparison for image generation benchmarks on FFHQ, LSUN and class-conditioned ImageNet. For ImageNet, numbers are reported without using CFG ($w = 1$).

Evaluation Metrics For image generation, results are compared according to Fréchet Inception Distance (FID, (Heusel et al., 2017), lower is better), over $50,000$ real samples from the corresponding datasets. For text-to-image tasks, we measure the zero-shot CLIP score (Radford et al., 2021) for measuring the faithfulness of generation given 5000 randomly sampled captions from COCO2017 (Lin et al., 2014) validation set. In addition, we also report the inference speed in `fps` on single A100 GPU.

3.2. Results

Quantitative Results We first evaluate the proposed method on standard image generation benchmarks. The quantitative comparison with the standard diffusion inference methods like DDPM (Ho et al., 2020) and the deterministic DDIM (Song et al., 2020a) are shown in Table 1. Despite lagging behind the 50-step DDIM inference, BOOT significantly improves the performance 1-step inference, and achieves better performance against DDIM with around

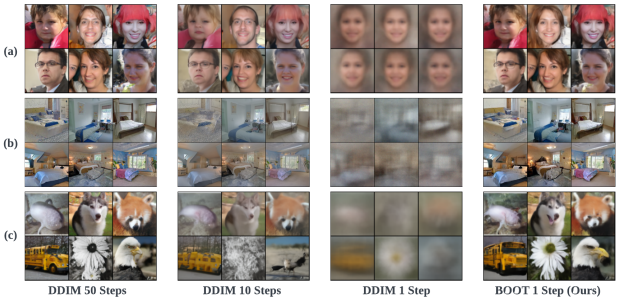


Figure 5. Uncurated samples of {50, 10, 1} DDIM sampling steps and the proposed BOOT from (a) FFHQ (b) LSUN (c) ImageNet benchmarks, respectively, given the same set of initial noise input.

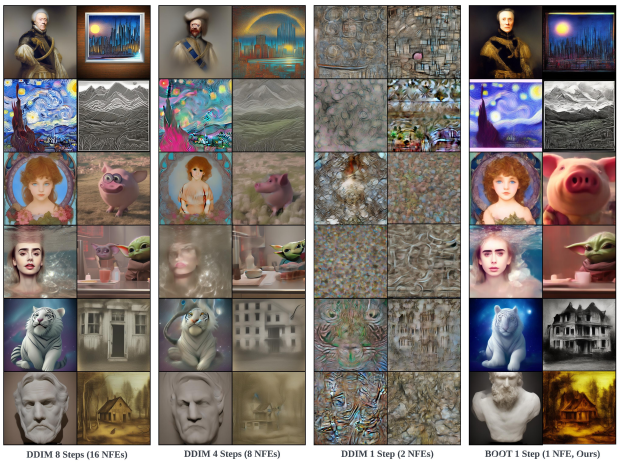


Figure 6. Uncurated samples of {50, 10, 1} DDIM sampling steps and the proposed BOOT from the SD teacher, given the same set of initial noise input and random prompts.

10 denoising steps, while maintaining $\times 10$ speed-up. Note that, the speed advantage doubles if CFG is considered.

We also conduct quantitative evaluation on text-to-image tasks. Using the SD teacher, we obtain a CLIP-score of 0.254 on COCO2017, a slight degradation compared to the 50-step DDIM results (0.262), while it generates 2 orders of magnitude faster, rendering real-time applications.

Visual Results We show the qualitative comparison in Figures 5 and 6 for image generation and text-to-image, respectively. For both cases, navïe 1-step inference fails completely, and the diffusion generally outputs grey and ill-structured images with fewer than 10 NFEs. In contrast, BOOT is able to synthesize high-quality images that are visually close (Figure 5) or semantically similar (Figure 6) to teacher's results with much more steps. Unlike the standard benchmarks, distilling text-to-image models (e.g., SD) typically leads to noticeably different generation from the original diffusion model, even starting with the same initial noise. We hypothesize it is a combined effect of highly complex underlying distribution and CFG. We show more results including pixel-space models in the appendix.

Figure 7. Ablation Study. (a) vs. (b): The additional boundary loss in § 2.2 alleviates the mode collapsing issue and prompts diversity in generation. (c) vs. (d): Uniform time training yields better generation compared with progressive time training.

3.3. Analysis

Importance of Boundary Condition We demonstrate the necessity of incorporating the boundary loss in Figure 7 (a) (b) where given the same noise inputs, we show the student outputs based on various target timesteps. As $\boldsymbol{y}_\theta(\boldsymbol{\epsilon}, t)$ tracks the signal-ODE output which gets more averaged results for t approaching 1. Yet, without proper boundary constraints, the results get the same sharpness across timesteps, producing over-saturated and non-realistic images. This indicates the learned student model has completely failed to capture the teacher distribution, yielding serious mode collapse.

Progressive v.s. Uniform Time Training We also show comparison of training strategies in Figure 7 (c) (d). Compared to the proposed way of sampling t uniformly, one may gain additional efficiency with a fixed schedule – progressively decreasing t while the training proceeds. While such method is reasonable given the fact that the student is always initialized from $t_{\max}$, and gradually learns to predicts the clean signals (small t) during training. However, progressive training tends to produce more artifacts than the navïve sampling approach (see visual comparison in Figure 7). We conjecture that it is more likely to accumulate unfixable errors in progressive training.

4. Conclusion

In summary, this paper introduced a novel technique *BOOT* to distill diffusion models into single step. The method did not require the presence of any real or synthetic data by learning a time-conditioned student model with bootstrapping objectives. The proposed approach achieved comparable generation quality while being significantly faster than the diffusion teacher, and was also applicable to large-scale text-to-image generation, showcasing its versatility.

References

Aguinaldo, A., Chiang, P.-Y., Gain, A., Patil, A., Pearson, K., and Feizi, S. Compressing gans using knowledge distillation. *arXiv preprint arXiv:1902.00159*, 2019.

Ascher, U. M. and Petzold, L. R. *Computer methods for ordinary differential equations and differential-algebraic equations*, volume 61. Siam, 1998.

Balaji, Y., Nah, S., Huang, X., Vahdat, A., Song, J., Kreis, K., Aittala, M., Aila, T., Laine, S., Catanzaro, B., et al. ediffi: Text-to-image diffusion models with an ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*, 2022.

Berthelot, D., Autef, A., Lin, J., Yap, D. A., Zhai, S., Hu, S., Zheng, D., Talbot, W., and Gu, E. Tract: Denoising diffusion models with transitive closure time-distillation. *arXiv preprint arXiv:2303.04248*, 2023.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A Large-scale Hierarchical Image Database. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

Dhariwal, P. and Nichol, A. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

Fu, Y., Chen, W., Wang, H., Li, H., Lin, Y., and Wang, Z. Autogan-distiller: Searching to compress generative adversarial networks. *arXiv preprint arXiv:2006.08198*, 2020.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems*, volume 27, pp. 2672–2680. Curran Associates, Inc., 2014a. URL https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *NeurIPS*, 2014b.

Gu, J., Bradbury, J., Xiong, C., Li, V. O., and Socher, R. Non-autoregressive neural machine translation. *arXiv preprint arXiv:1711.02281*, 2017.

Gu, J., Zhai, S., Zhang, Y., Bautista, M. A., and Susskind, J. f-dm: A multi-stage diffusion model via progressive signal transformation. *arXiv preprint arXiv:2210.04955*, 2022.

Gu, J., Trevithick, A., Lin, K.-E., Susskind, J., Theobalt, C., Liu, L., and Ramamoorthi, R. Nerfdiff: Single-image view synthesis with nerf-guided distillation from 3d-aware diffusion. *arXiv preprint arXiv:2302.10109*, 2023.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Ho, J. and Salimans, T. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Ho, J., Chan, W., Saharia, C., Whang, J., Gao, R., Gritsenko, A., Kingma, D. P., Poole, B., Norouzi, M., Fleet, D. J., et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022a.

Ho, J., Salimans, T., Gritsenko, A. A., Chan, W., Norouzi, M., and Fleet, D. J. Video diffusion models. In *ICLR Workshop on Deep Generative Models for Highly Structured Data*, 2022b.

Hsieh, C.-Y., Li, C.-L., Yeh, C.-K., Nakhost, H., Fujii, Y., Ratner, A., Krishna, R., Lee, C.-Y., and Pfister, T. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes, 2023.

Jing, B., Corso, G., Berlinghieri, R., and Jaakkola, T. Subspace diffusion generative models. *arXiv preprint arXiv:2205.01490*, 2022.

Jolicoeur-Martineau, A., Li, K., Piché-Taillefer, R., Kachman, T., and Mitliagkas, I. Gotta go fast when generating data with score-based models. *arXiv preprint arXiv:2105.14080*, 2021.

Karras, T., Aila, T., Laine, S., and Lehtinen, J. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

Karras, T., Aittala, M., Laine, S., Härkönen, E., Hellsten, J., Lehtinen, J., and Aila, T. Alias-free generative adversarial networks. *arXiv preprint arXiv:2106.12423*, 2021.

Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. *arXiv preprint arXiv:2206.00364*, 2022.

Kim, Y. and Rush, A. M. Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947*, 2016.

Kingma, D., Salimans, T., Poole, B., and Ho, J. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Li, X., Thickstun, J., Gulrajani, I., Liang, P. S., and Hashimoto, T. B. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35:4328–4343, 2022.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft COCO: Common Objects in Context. *European Conference on Computer Vision*, pp. 740–755, 2014.

Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Liu, H., Chen, Z., Yuan, Y., Mei, X., Liu, X., Mandic, D., Wang, W., and Plumbley, M. D. Audioldm: Text-to-audio generation with latent diffusion models. *arXiv preprint arXiv:2301.12503*, 2023.

Liu, X., Gong, C., and Liu, Q. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *arXiv preprint arXiv:2206.00927*, 2022.

Luhman, E. and Luhman, T. Knowledge distillation in iterative generative models for improved sampling speed. *arXiv preprint arXiv:2101.02388*, 2021.

Meng, C., Gao, R., Kingma, D. P., Ermon, S., Ho, J., and Salimans, T. On distillation of guided diffusion models. *arXiv preprint arXiv:2210.03142*, 2022.

Nichol, A. Q. and Dhariwal, P. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pp. 8162–8171. PMLR, 2021.

Oord, A., Li, Y., Babuschkin, I., Simonyan, K., Vinyals, O., Kavukcuoglu, K., Driessche, G., Lockhart, E., Cobo, L., Stimberg, F., et al. Parallel wavenet: Fast high-fidelity speech synthesis. In *International conference on machine learning*, pp. 3918–3926. PMLR, 2018.

Poole, B., Jain, A., Barron, J. T., and Mildenhall, B. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models, 2021.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S. K. S., Ayan, B. K., Mahdavi, S. S., Lopes, R. G., et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022.

Salimans, T. and Ho, J. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pp. 2256–2265. PMLR, 2015.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020a.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020b.

Song, Y., Dhariwal, P., Chen, M., and Sutskever, I. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

Süli, E. and Mayers, D. F. *An introduction to numerical analysis*. Cambridge university press, 2003.

Vahdat, A., Kreis, K., and Kautz, J. Score-based generative modeling in latent space. In *Neural Information Processing Systems (NeurIPS)*, 2021.

Wang, Z. J., Montoya, E., Munechika, D., Yang, H., Hoover, B., and Chau, D. H. DiffusionDB: A large-scale prompt gallery dataset for text-to-image generative models. *arXiv:2210.14896 [cs]*, 2022. URL https://arxiv.org/abs/2210.14896.

Watson, D., Chan, W., Martin-Brualla, R., Ho, J., Tagliasacchi, A., and Norouzi, M. Novel view synthesis with diffusion models. *arXiv preprint arXiv:2210.04628*, 2022.

Yu, F., Seff, A., Zhang, Y., Song, S., Funkhouser, T., and Xiao, J. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

Zhou, C., Neubig, G., and Gu, J. Understanding knowledge distillation in non-autoregressive machine translation. *arXiv preprint arXiv:1911.02727*, 2019.

Appendices

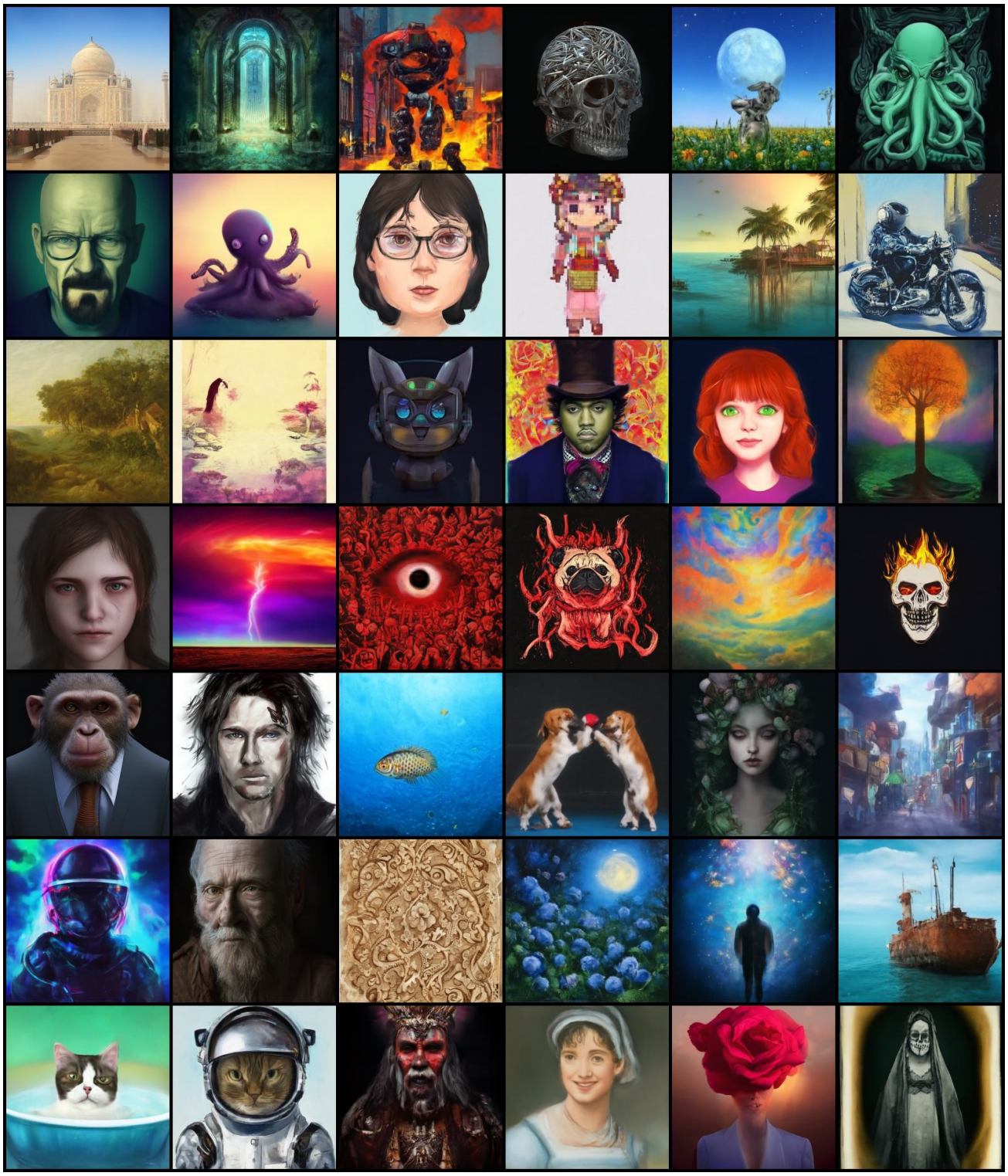


Figure 8. Curated samples of our distilled single-step model from DeepFloyd IF with the prompts from *diffusiondb* (Wang et al., 2022).

Appendices

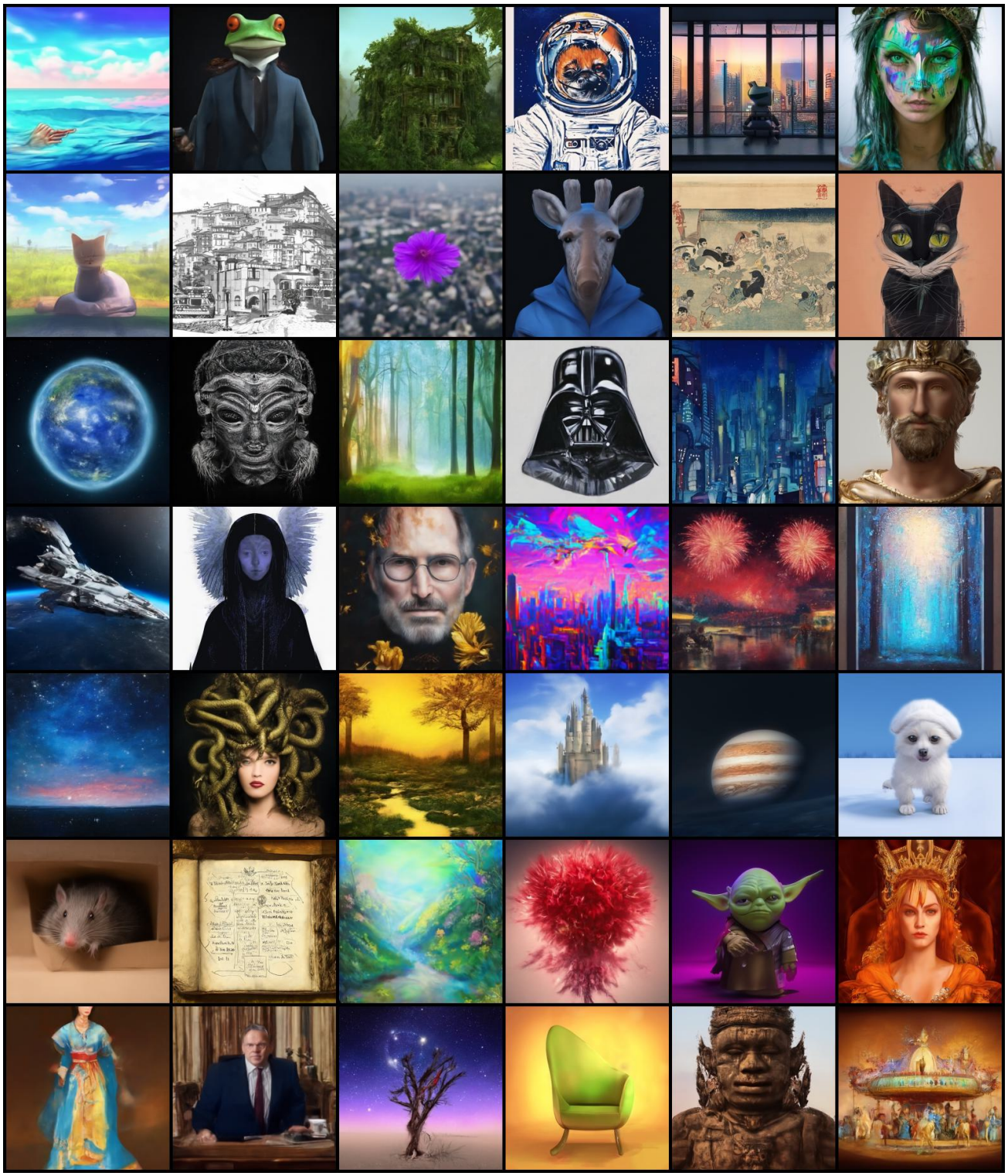


Figure 9. Curated samples of our distilled single-step model from DeepFloyd IF with the prompts from *diffusiondb* (Wang et al., 2022).

Appendices

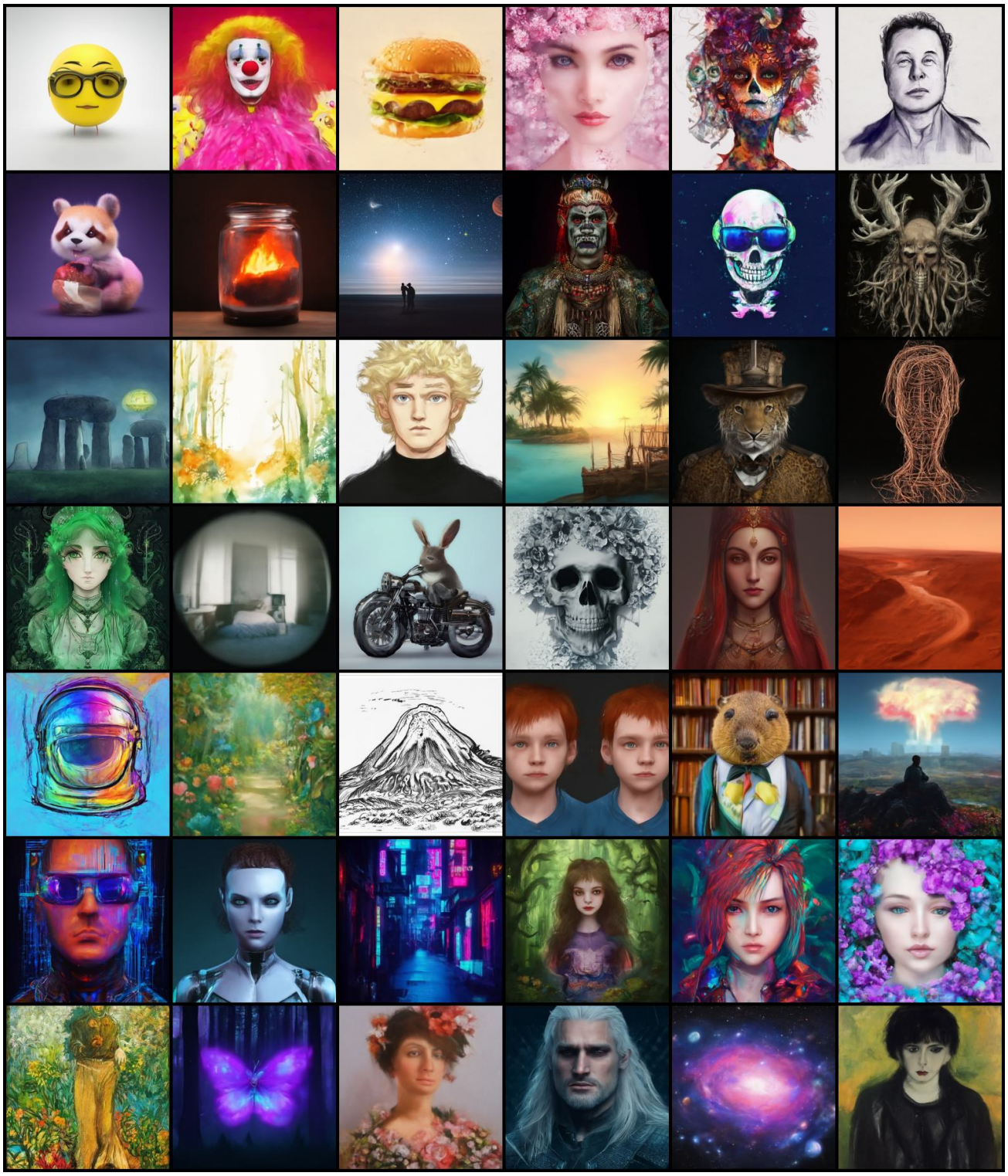


Figure 10. Curated samples of our distilled single-step model from DeepFloyd IF with the prompts from *diffusiondb* (Wang et al., 2022).

A. Preliminaries

A.1. Diffusion Models

Diffusion models (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020) are a class of deep generative models that generate data by progressively removing noise from the initial input. In this work, we consider continuous-time diffusion models (Song et al., 2020b; Kingma et al., 2021; Karras et al., 2022) in the variance preserving formulation (Salimans & Ho, 2022). One can perform ancestral sampling (Ho et al., 2020) to synthesize new data from the learned model. While the conventional method is stochastic, (Song et al., 2020a) shows that one can follow a deterministic DDIM sampler to generate the final sample $\boldsymbol{x}_0$, which follows the update rule:

$$\boldsymbol{x}_s = (\sigma_s/\sigma_t)\,\boldsymbol{x}_t + (\alpha_s - \alpha_t\sigma_s/\sigma_t)\,\boldsymbol{f}_\phi(\boldsymbol{x}_t, t), \quad s < t. \tag{6}$$

As noted in previous studies (Lu et al., 2022), Eq. (6) is equivalent to the first-order ODE solver for the underlying probability-flow (PF) ODE (Song et al., 2020b). Therefore, the step size $\delta = t - s$ has to be small to mitigate the error accumulation. Additionally, using higher-order solvers such as Runge-Kutta (Süli & Mayers, 2003), Heun (Ascher & Petzold, 1998) and other solvers (Lu et al., 2022; Jolicoeur-Martineau et al., 2021) are able further to reduce the number of function evaluations (NFEs). Yet, these approaches are unable to work in single-step scenarios.

A.2. Distillation of Diffusion Models

Orthogonal to the development of ODE solvers, distillation-based techniques have been proposed to learn faster student models from the pre-trained diffusion teacher. The most straightforward way is to perform **direct distillation** (Luhman & Luhman, 2021) where a student model $\boldsymbol{g}_\theta$ is forced to learn from the diffusion model output, which itself is computationally expensive:

$$\mathcal{L}_\theta^{\text{Direct}} = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,I)}\|\boldsymbol{g}_\theta(\boldsymbol{\epsilon}) - \texttt{ODE-Solver}(\boldsymbol{f}_\phi, \boldsymbol{\epsilon}, T \to 0)\|_2^2, \tag{7}$$

where `ODE-solver` stands for any solvers like DDIM as mentioned above. While this naive approach shows promising results, it usually requires over 50 steps of evaluations to get reasonable distillation targets, which itself becomes a bottleneck in learning large-scale models.

Alternatively, recent studies (Salimans & Ho, 2022; Song et al., 2023; Berthelot et al., 2023) have been proposed to avoid running the full diffusion path during distillation. Take the consistency model (CM, Song et al., 2023) as an example. A time-conditioned student model $\boldsymbol{g}_\theta(\boldsymbol{x}_t, t)$ is trained to predict self-consistent outputs along the diffusion trajectory in a bootstrap fashion:

$$\mathcal{L}_\theta^{\text{CM}} = \mathbb{E}_{\boldsymbol{x}_t\sim q(\boldsymbol{x}_t|\boldsymbol{x}),s,t\sim[0,T],s<t}\|\boldsymbol{g}_\theta(\boldsymbol{x}_t, t) - \boldsymbol{g}_{\theta^-}(\hat{\boldsymbol{x}}_s, s)\|_2^2, \tag{8}$$

where $\hat{\boldsymbol{x}}_s = \texttt{ODE-Solver}(\boldsymbol{f}_\phi, \boldsymbol{x}_t, t \to s)$, typically with single step evaluation using Eq. (6).

Here, θ^- is typically an exponential moving average (EMA) of the student parameters θ, which is important for these approaches, as the self-consistency objectives are likely to collapse to trivial solutions by always predicting similar outputs. After the training is done, one can generate samples by executing $\boldsymbol{g}_\theta(\boldsymbol{x}_T, T)$ with a single NFE. Note that, Eq. (8) requires sampling $\boldsymbol{x}_t$ from the real data sample $\boldsymbol{x}$, which is the core of bootstrapping: the model learns to denoise more and more noisy input until $\boldsymbol{x}_T$. However, the original training data $\boldsymbol{x}$ for distillation is inaccessible in many tasks, for example, text-to-image generation models that require billions of paired data for training. A possible solution is using a different dataset for distillation; however, the mismatch in the distributions of the two datasets would result in suboptimal distillation performance.

B. Algorithm Details

B.1. Notations

In this paper, we use $\boldsymbol{f}_\phi(\boldsymbol{x}, t)$ to represent the diffusion model that denoises the noisy sample $\boldsymbol{x}$ into its clean version, and we derive the DDIM sampler (Eq. (6)) following the definition of Song et al. (2020a): we deterministically synthesize $\boldsymbol{x}_s$

based on the following update rule:

$$\begin{aligned}\boldsymbol{x}_s &= \texttt{ODE-Solver}(\boldsymbol{f}_\phi, \boldsymbol{\epsilon}, T \to s) \\ &= \alpha_s \boldsymbol{f}_\phi(\boldsymbol{x}_t, t) + \sigma_s \left(\frac{\boldsymbol{x}_t - \alpha_t \boldsymbol{f}_\phi(\boldsymbol{x}_t, t)}{\sigma_t} \right) \\ &= \frac{\sigma_s}{\sigma_t} \boldsymbol{x}_t + \left(\alpha_s - \alpha_t \frac{\sigma_s}{\sigma_t} \right) \boldsymbol{f}_\phi(\boldsymbol{x}_t, t)\end{aligned} \tag{9}$$

where $0 \leq s < t \leq T$. Here we use $\texttt{ODE-Solver}$ to represent the DDIM sampling from a random noise $\boldsymbol{x}_T = \boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$, and iteratively obtain the sample at step s. In practice, we can generalize to higher-order ODE-solvers for better efficiency.

For distillation, we define the student model with $\boldsymbol{g}_\theta(\boldsymbol{\epsilon}, t)$ which approximates $\boldsymbol{x}_t$ along the diffusion trajectory above. To avoid directly predicting the noisy samples $\boldsymbol{x}_t$ with neural networks, we re-parameterize $\boldsymbol{g}_\theta(\boldsymbol{\epsilon}, t) = \alpha_t \boldsymbol{y}_\theta(\boldsymbol{\epsilon}, t) + \sigma_t \boldsymbol{\epsilon}$ where the noise part is constant throughout t except the scale factor σ_t. In this way, the learning goal $\boldsymbol{y}_\theta(\boldsymbol{\epsilon}, t)$ is to predict a new variable $\boldsymbol{y}_t$: the "signal" part of the original variable $\boldsymbol{y}_t = (\boldsymbol{x}_t - \sigma_t \boldsymbol{\epsilon})/\alpha_t$.

B.2. Derivation of Signal-ODE

Based on the definition of $\boldsymbol{y}_t = (\boldsymbol{x}_t - \sigma_t \boldsymbol{\epsilon})/\alpha_t$, we can derive the following equations from Eq. (9):

$$\begin{aligned}\boldsymbol{x}_s &= \frac{\sigma_s}{\sigma_t} \boldsymbol{x}_t + \left(\alpha_s - \alpha_t \frac{\sigma_s}{\sigma_t} \right) \boldsymbol{f}_\phi(\boldsymbol{x}_t, t) \\ \Rightarrow \alpha_s \boldsymbol{y}_s + \sigma_s \boldsymbol{\epsilon} &= \frac{\sigma_s}{\sigma_t} (\alpha_t \boldsymbol{y}_t + \sigma_t \boldsymbol{\epsilon}) + \left(\alpha_s - \alpha_t \frac{\sigma_s}{\sigma_t} \right) \boldsymbol{f}_\phi(\boldsymbol{x}_t, t) \\ \Rightarrow \alpha_s \boldsymbol{y}_s + \cancel{\sigma_s \boldsymbol{\epsilon}} &= \alpha_t \frac{\sigma_s}{\sigma_t} \boldsymbol{y}_t + \cancel{\sigma_s \boldsymbol{\epsilon}} + \left(\alpha_s - \alpha_t \frac{\sigma_s}{\sigma_t} \right) \boldsymbol{f}_\phi(\boldsymbol{x}_t, t) \\ \Rightarrow \boldsymbol{y}_s &= \frac{\alpha_t \sigma_s}{\sigma_t \alpha_s} \boldsymbol{y}_t + \left(1 - \frac{\alpha_t \sigma_s}{\sigma_t \alpha_s} \right) \boldsymbol{f}_\phi(\boldsymbol{x}_t, t) \\ &= \left(1 - e^{\lambda_s - \lambda_t} \right) \boldsymbol{f}_\phi(\boldsymbol{x}_t, t) + e^{\lambda_s - \lambda_t} \boldsymbol{y}_t,\end{aligned} \tag{10}$$

where we use the auxiliary variable $\lambda_t = -\log(\alpha_t / \sigma_t)$ for simplifying the equations. As mentioned in § 2.1, we can further obtain the continuous form of Eq. (10) by assigning $t - s \to 0$. That is, Eq. (10) is equivalent to that shown in the following:

$$\begin{aligned}\boldsymbol{y}_s &= \left(1 - e^{\lambda_s - \lambda_t} \right) \boldsymbol{f}_\phi(\boldsymbol{x}_t, t) + e^{\lambda_s - \lambda_t} \boldsymbol{y}_t \\ \Rightarrow \boldsymbol{y}_t - \boldsymbol{y}_s &= -\left(1 - e^{\lambda_s - \lambda_t} \right) \left(\boldsymbol{f}_\phi(\boldsymbol{x}_t, t) - \boldsymbol{y}_t \right) \\ \Rightarrow \frac{\boldsymbol{y}_t - \boldsymbol{y}_s}{t - s} &= -\frac{e^{\lambda_t} - e^{\lambda_s}}{t - s} \cdot e^{-\lambda_t} \left(\boldsymbol{f}_\phi(\boldsymbol{x}_t, t) - \boldsymbol{y}_t \right) \\ \Rightarrow \frac{\mathrm{d}\boldsymbol{y}_t}{\mathrm{d}t} &= -\cancel{e^{\lambda_t}} \cdot \lambda_t' \cdot \cancel{e^{-\lambda_t}} \left(\boldsymbol{f}_\phi(\boldsymbol{x}_t, t) - \boldsymbol{y}_t \right)\end{aligned} \tag{11}$$

where $\lambda_t' = \mathrm{d}\lambda_t / \mathrm{d}t$. Given a fixed noise input $\boldsymbol{\epsilon}$, Eq. (11) defines an ODE over $\boldsymbol{y}_\theta$ w.r.t t, which we call *Signal-ODE*, as both sides of the equation only operate in "low-frequency" signal space.

B.3. Bootstrapping Objectives

The bootstrapping objectives in Eq. (3) can be easily derived by taking the finite difference of Eq. (8). Here we use $\boldsymbol{y}_\theta(\boldsymbol{\epsilon}, t)$ to estimate $\boldsymbol{y}_t$, and use $\hat{\boldsymbol{x}}_t$ to represent the noisy image obtained from $\boldsymbol{y}_\theta(\boldsymbol{\epsilon}, t)$.

$$\begin{aligned}
\mathcal{L}_\theta &= \mathbb{E}_{\boldsymbol{\epsilon},t}\left[\tilde{\omega}_t \left|\left| \frac{\mathrm{d}\boldsymbol{y}_\theta(\boldsymbol{\epsilon},t)}{\mathrm{d}t} + \lambda_t' \cdot \left(\boldsymbol{f}_\phi(\hat{\boldsymbol{x}}_t, t) - \boldsymbol{y}_\theta(\boldsymbol{\epsilon},t)\right)\right|\right|_2^2\right] \\
&\approx \mathbb{E}_{\boldsymbol{\epsilon},t}\left[\tilde{\omega}_t \| \frac{\boldsymbol{y}_\theta(\boldsymbol{\epsilon},s) - \boldsymbol{y}_\theta(\boldsymbol{\epsilon},t)}{\delta} - \lambda_t' \left(\boldsymbol{f}_\phi(\hat{\boldsymbol{x}}_t, t) - \boldsymbol{y}_\theta(\boldsymbol{\epsilon},t)\right) \|_2^2\right] \\
&= \mathbb{E}_{\boldsymbol{\epsilon},t}\left[\frac{\tilde{\omega}_t}{\delta^2} \| \boldsymbol{y}_\theta(\boldsymbol{\epsilon},s) - \left[\boldsymbol{y}_\theta(\boldsymbol{\epsilon},t) + \delta\lambda_t' \left(\boldsymbol{f}_\phi(\hat{\boldsymbol{x}}_t, t) - \boldsymbol{y}_\theta(\boldsymbol{\epsilon},t)\right)\right] \|_2^2\right] \\
&= \mathbb{E}_{\boldsymbol{\epsilon},t}\left[\frac{\tilde{\omega}_t}{\delta^2} \| \boldsymbol{y}_\theta(\boldsymbol{\epsilon},s) - \hat{\boldsymbol{y}}_\theta(\boldsymbol{\epsilon},s)\|_2^2\right],
\end{aligned} \tag{12}$$

where $s = t - \delta$, and $\hat{\boldsymbol{y}}_\theta(\boldsymbol{\epsilon}, s)$ is the approximated target. $\tilde{\omega}_t$ is the additional weight, where by default $\tilde{\omega}_t = 1$. To stabilize training, a stop-gradient operation $\mathtt{SG}(.)$ is typically included:

$$\mathcal{L}_\theta = \mathbb{E}_{\boldsymbol{\epsilon},t}\left[\frac{\tilde{\omega}_t}{\delta^2} \| \boldsymbol{y}_\theta(\boldsymbol{\epsilon},s) - \mathtt{SG}(\hat{\boldsymbol{y}}_\theta(\boldsymbol{\epsilon},s))\|_2^2\right]. \tag{13}$$

In our experiments, we also find that it helps use $\tilde{\omega}_t = 1/\lambda_t'^2$ for text-to-image generation.

We can take advantage of higher-order solvers for a more accurate target that reduces the discretization error. For example, one can use Heun's method (Ascher & Petzold, 1998) to first calculate the intermediate value $\tilde{\boldsymbol{y}}_\theta(\boldsymbol{\epsilon}, s)$, and then the final approximation $\hat{\boldsymbol{y}}_\theta(\boldsymbol{\epsilon}, s)$:

$$\begin{aligned}
\tilde{\boldsymbol{y}}_\theta(\boldsymbol{\epsilon},s) &= \boldsymbol{y}_\theta(\boldsymbol{\epsilon},t) + \delta\lambda_t' \left(\boldsymbol{f}_\phi(\hat{\boldsymbol{x}}_t, t) - \boldsymbol{y}_\theta(\boldsymbol{\epsilon},t)\right), \quad \tilde{\boldsymbol{x}}_s = \alpha_s \tilde{\boldsymbol{y}}_\theta(\boldsymbol{\epsilon},s) + \sigma_s\boldsymbol{\epsilon} \\
\hat{\boldsymbol{y}}_\theta(\boldsymbol{\epsilon},s) &= \boldsymbol{y}_\theta(\boldsymbol{\epsilon},t) + \frac{\delta\lambda_t'}{2}\left[\left(\boldsymbol{f}_\phi(\hat{\boldsymbol{x}}_t, t) - \boldsymbol{y}_\theta(\boldsymbol{\epsilon},t)\right) + \left(\boldsymbol{f}_\phi(\tilde{\boldsymbol{x}}_s, s) - \tilde{\boldsymbol{y}}_\theta(\boldsymbol{\epsilon},s)\right)\right].
\end{aligned} \tag{14}$$

Using Heun's method essentially doubles the evaluations of the teacher model during training, while the add-on overheads are manageable as we stop the gradients to the teacher model.

B.4. Training Algorithm

We summarize the training algorithm of BOOT in Algorithm 1, where by default we assume conditional diffusion model with classifier-free guidance and DDIM solver. Here, for simplicity, we write $\lambda_t' \approx (1 - \frac{\alpha_t\sigma_s}{\sigma_t\alpha_s})/\delta$. For unconditional models, we can simply remove the context sampling part.

C. Connections to Existing Literature

C.1. Physics Informed Neural Networks (PINNs)

Physics-Informed Neural Networks (PINNs, Raissi et al., 2019) are powerful approaches that combine the strengths of neural networks and physical laws to solve ODEs. Unlike traditional numerical methods, which rely on discretization and iterative solvers, PINNs employ machine learning techniques to approximate the solution of ODEs. The key idea behind PINNs is to incorporate physics-based constraints directly into the training process of neural networks. By embedding the governing equations and available boundary or initial conditions as loss terms, PINNs can effectively learn the underlying physics while simultaneously discovering the solution. This ability makes PINNs highly versatile in solving a wide range of ODEs, including those arising in fluid dynamics, solid mechanics, and other scientific domains.

Despite being motivated from different perspectives, BOOT is similar to PINNs at high level which shares the same goals of learning ODE/PDE solvers directly through neural networks. In the PINNs domain, solving ODEs can also be simplified into two objectives, i.e. the differential equation (DE) loss (Eq. (3)) and the boundary condition (BC) loss (Eq. (5)). The major difference is that, PINNs are mainly focusing on learning complex ODEs/PDEs for single problems where neural networks are used as the universal approximator to ease the discretization difficulties of traditional solvers, while the data space is

Algorithm 1 Distillation using BOOT for Conditional Diffusion Models.

Require: pretrained diffusion model $\boldsymbol{f}_\phi$, initial student parameter from the teacher $\theta \leftarrow \phi$, step size δ, learning rate η, CFG weight w, context dataset $\mathcal{D}$, negative condition $\boldsymbol{n} = \emptyset$, $t_{\min}, t_{\max}$, β.

while not converged **do**
  Sample noise input $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$
  Sample context input $\boldsymbol{c} \sim \mathcal{D}$
  Sample $t \sim (t_{\min}, t_{\max}), s = \min(t - \delta, t_{\min}))$
  Compute noise schedule $\alpha_t, \sigma_t, \alpha_s, \sigma_s$
  Compute $\lambda'_t \approx (1 - \frac{\alpha_t \sigma_s}{\sigma_t \alpha_s})/\delta$
  Generate the model predictions:
    $\boldsymbol{y}_t = \boldsymbol{y}_\theta(\boldsymbol{\epsilon}, t, \boldsymbol{c}), \quad \boldsymbol{y}_s = \boldsymbol{y}_\theta(\boldsymbol{\epsilon}, s, \boldsymbol{c}), \quad \boldsymbol{y}_{t_{\max}} = \boldsymbol{y}_\theta(\boldsymbol{\epsilon}, t_{\max}, \boldsymbol{c})$
  Generate the noisy sample $\hat{\boldsymbol{x}}_t = \alpha_t \boldsymbol{y}_t + \sigma_t \boldsymbol{\epsilon}$
  Compute the denoised target:
    $\tilde{\boldsymbol{f}}_t = \boldsymbol{f}_\phi(\hat{\boldsymbol{x}}_t, t, \boldsymbol{n}) + w \cdot (\boldsymbol{f}_\phi(\hat{\boldsymbol{x}}_t, t, \boldsymbol{c}) - \boldsymbol{f}_\phi(\hat{\boldsymbol{x}}_t, t, \boldsymbol{n}))$
    $\tilde{\boldsymbol{f}}_{t_{\max}} = \boldsymbol{f}_\phi(\boldsymbol{\epsilon}, t_{\max}, \boldsymbol{n}) + w \cdot (\boldsymbol{f}_\phi(\boldsymbol{\epsilon}, t_{\max}, \boldsymbol{c}) - \boldsymbol{f}_\phi(\boldsymbol{\epsilon}, t_{\max}, \boldsymbol{n}))$
  Compute the bootstrapping loss $\mathcal{L}_\theta^{\text{BS}} = \frac{1}{(\delta \lambda'_t)^2} \|\boldsymbol{y}_s - \texttt{SG}(\boldsymbol{y}_t + \delta \lambda'_t (\tilde{\boldsymbol{f}}_t - \boldsymbol{y}_t))\|_2^2$
  Compute the boundary loss $\mathcal{L}_\theta^{\text{BC}} = \|\boldsymbol{y}_{t_{\max}} - \tilde{\boldsymbol{f}}_{t_{\max}}\|_2^2$
  Update model parameters $\theta \leftarrow \theta - \eta \cdot \nabla_\theta \left(\mathcal{L}_\theta^{\text{BS}} + \beta \mathcal{L}_\theta^{\text{BC}}\right)$
end while
return Trained model parameters θ

relatively low-dimensional. In contrast, in BOOT, we are trying to learn single-step generative models that can synthesize data in high-dimensional space (e.g., millions of pixels) from random noise input and conditions (e.g., labels, prompts), which, to the best of our knowledge, no existing work has applied similar methods in generative modeling. Besides, standard PINNs usually compute the derivatives (Eq. (3)) directly with auto-differentiation, while in this paper, we take the finite difference method and propose a bootstrapping-based algorithm.

C.2. Consistency Models / TRACT

The most related previous works to our work are Consistency Models (Song et al., 2023), and concurrently TRACT (Berthelot et al., 2023), which propose bootstrapping-style algorithms that distill the diffusion models to map an intermediate noisy training example at time step t to the teacher's t-step denoising outputs following the DDIM inference procedure. The student's training target is constructed by first running the teacher model with one step, followed by the self-teacher with $t - 1$ steps. As illustrated in Figure 2, BOOT adopts a different direction of performing bootstrapping, where it starts from the Gaussian noise prior and maps it to an intermediate step t in one shot. This change makes a significant difference in the modeling implications, as one does not require the presence of any training data and can achieve data-free distillation, which none of the prior works are capable of.

C.3. Single-step Generative Models

BOOT is also related to other single-step generative models including VAEs (Kingma & Welling, 2013) and GANs (Goodfellow et al., 2014b), which aims to synthesize data in a single forward pass. Compared to VAEs, BOOT does not require an encoder network. Thanks to the power of the underlying diffusion model, BOOT can produce higher-contrast and more realistic samples. Compared to GANs, BOOT does not require a discriminator or critic network. In addition, the distillation process of BOOT enables better-controlled exploration of the text-image joint space, which is explored by the pretrained diffusion models, resulting in more coherent and realistic samples in text-guided generation. BOOT is also more stable to learn. By contrast, GANs are hard to train due to their adversarial nature.

C.4. Improving Efficiency of Diffusion Models

Speeding up inference of diffusion models is a broad area. Recent works and also our work (Luhman & Luhman, 2021; Salimans & Ho, 2022; Meng et al., 2022; Song et al., 2023; Berthelot et al., 2023) aim at reducing the number of diffusion

model inference steps via distillation. Aside from distillation methods, other representative approaches include advanced ODE solvers (Karras et al., 2022; Lu et al., 2022), low-dimension space diffusion (Rombach et al., 2021; Vahdat et al., 2021; Jing et al., 2022; Gu et al., 2022), and improved diffusion targets (Lipman et al., 2023; Liu et al., 2022). BOOT is orthogonal and complementary to these approaches, and can theoretically benefit from improvements made in all these aspects.

C.5. Knowledge Distillation for Generative Models

Knowledge distillation (Hinton et al., 2015) has seen successful applications in learning efficient generative models, including model compression (Kim & Rush, 2016; Aguinaldo et al., 2019; Fu et al., 2020; Hsieh et al., 2023) and non-autoregressive sequence generation (Gu et al., 2017; Oord et al., 2018; Zhou et al., 2019). We believe that BOOT could inspire a new paradigm of distilling powerful generative models without requiring access to the training data.

D. Additional Experimental Settings

D.1. Datasets

While the proposed method is data-free, we list the additional dataset information that used to train our teacher diffusion models:

FFHQ (https://github.com/NVlabs/ffhq-dataset) contains 70k images of real human faces in resolution of 1024×1024. In most of our experiments, we resize the images to a low resolution at 64×64 for early-stage benchmarking.

LSUN (https://www.yf.io/p/lsun) is a collection of large-scale image dataset containing 10 scenes and 20 object categories. Following previous works (Song et al., 2023), we choose the category Bedroom (3M images), and train an unconditional diffusion teacher. All images are resized to 256×256 with center-crop. We use LSUN to validate the ability of learning in relative high-resolution scenarios.

ImageNet-1K (https://image-net.org/download.php) contains 1.28M images across 1000 classes. We directly merge all the training images with class labels and train a class-conditioned diffusion teacher. All images are resized to 64×64 with center-crop. To support test-time classifier-free guidance, the teacher model is trained with 0.2 unconditional probability.

As we do not need to train our own teacher models for text-to-image experiments, no additional text-image pairs are required in this paper. However, our distillation still requires the text conditions for querying the teacher diffusion. To better capture and generalize the real user preference of such diffusion models, we choose to adopt the collected prompt datasets:

DiffusionDB (https://poloclub.github.io/diffusiondb/) contains 14M images generated by Stable Diffusion using prompts and hyperparameters specified by users. For the purpose of our experiments, we only keep the text prompts and discard all model-generated images as well as meta-data and hyperparameters so that they can be used for different teacher models. We use the same prompts for both latent and pixel space models.

D.2. Text-to-Image Teachers

We directly choose the recently open-sourced large-scale diffusion models as our teacher models. More specifically, we looked into the following models:

StableDiffusion (SD) (https://github.com/Stability-AI/stablediffusion) is an open-source text-to-image latent diffusion model (Rombach et al., 2021) conditioned on the penultimate text embeddings of a CLIP ViT-H/14 (Radford et al., 2021) text encoder. Different standard diffusion models, SD performs diffusion purely in the latent space. In this work, we use the checkpoint of **SD v2.1-Base** (https://huggingface.co/stabilityai/stable-diffusion-2-1-base) as our teacher which first generates in 64×64 latent space, and then directly upscaled to 512×512 resolution with the pre-trained VAE decoder. The teacher model was trained on subsets of LAION-5B (Schuhmann et al., 2022) with noise prediction objective.

DeepFloyd IF (IF) (https://github.com/deep-floyd/IF) is a recently open-source text-to-image model with a high degree of photorealism and language understanding. IF is a modular composed of a frozen text encoder and three cascaded pixel diffusion modules, similar to Imagen (Saharia et al., 2022): a base model that generates 64×64 image based on text prompt and two super-resolution models ($256 \times 256, 1024 \times 1024$). All stages of the model utilize a frozen text

Hyperparameter	Image Generation			Text-to-Image		
	FFHQ	LSUN	ImageNet	SD-Base	IF-I-L	IF-II-M
Architecture						
Denosing resolution	64×64	256×256	64×64	64×64	64×64	256×256
Base channels	128	128	192			
Multipliers	1,2,3,4	1,1,2,2,4,4	1,2,3,4			
# of Resblocks	1	1	2			
Attention resolutions	8,16	8,16	8,16		– Default –	
Noise schedule	cosine	cosine	cosine			
Model Prediction	signal	signal	signal			
Text Encoder	-	-	-	CLIP	T5	T5
Training						
Loss weighting	uniform	uniform	uniform	$\lambda_t'^{-2}$	$\lambda_t'^{-2}$	$\lambda_t'^{-2}$
Bootstrapping step size	0.04	0.04	0.04	0.01	0.04	0.04
CFG weight	-	-	$1 \sim 5$	7.5	7.0	4.0
Learning rate	1e-4	1e-4	3e-4	2e-5	2e-5	2e-5
Batch size	128	128	1024	64	64	32
EMA decay rate	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
Training iterations	500k	500k	300k	500k	500k	100k

Table 2. Hyperparameters used for training BOOT. The CFG weights for text-to-image models are determined based on the default value of the open-source codebase.

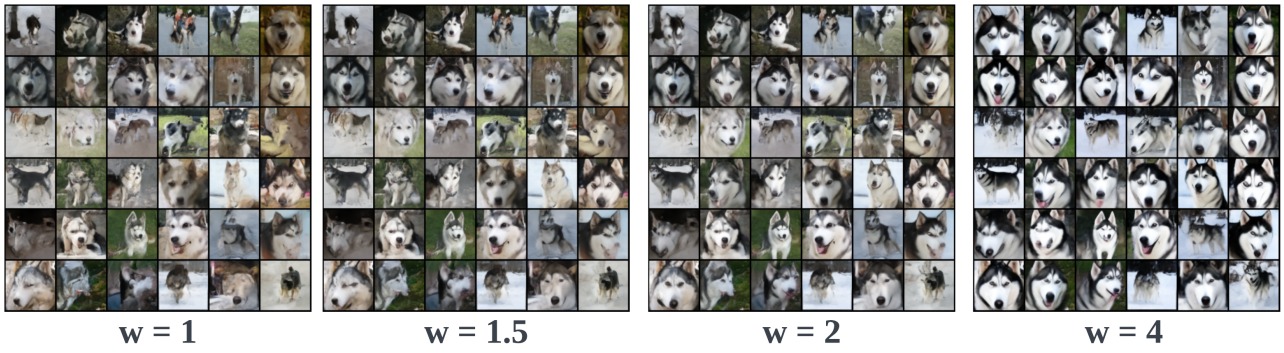


Figure 11. The distilled student is able to trade generation quality with diversity based on CFG weights.

encoder based on the T5 (Raffel et al., 2020) to extract text embeddings, which are then fed into a UNet architecture enhanced with cross-attention and attention pooling. Models were trained on 1.2B text-image pairs (based on LAION (Schuhmann et al., 2022) and few additional internal datasets) with noise prediction objective. In this paper, we conduct experiments on the first two resolutions ($64 \times 64, 256 \times 256$) with the checkpoints of **IF-I-L-v1.0** (`https://huggingface.co/DeepFloyd/IF-I-L-v1.0`) and **IF-II-M-v1.0** (`https://huggingface.co/DeepFloyd/IF-II-M-v1.0`).

D.3. Model Architectures

We follow the standard U-Net architecture (Nichol & Dhariwal, 2021) for image generation benchmarks and adopt the hyperparameters similar in f-DM (Gu et al., 2022). For text-to-image applications, we keep the default architecture setups from the teacher models unchanged. As mentioned in the main paper, we initialize the weights of the student models directly from the pretrained checkpoints and use *zero* initialization for the newly added modules, such as target time and CFG weight embeddings. We include additional architecture details in the Table 2.

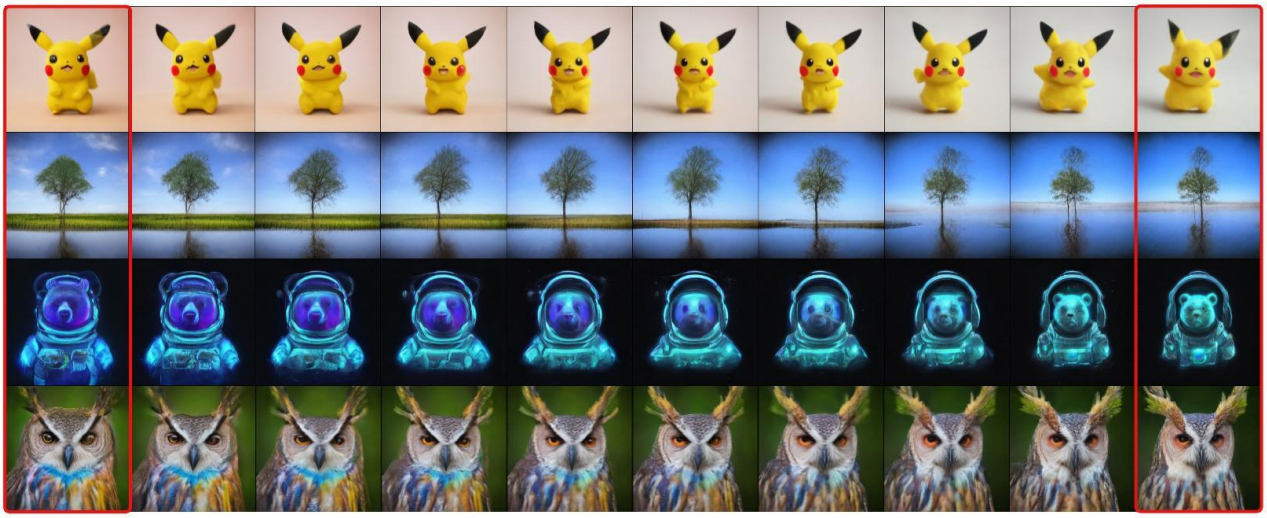


Figure 12. Latent space interpolation of the student model distilled from the IF teacher. We randomly sample two noises to generate images (shown in red boxes) given the same text prompts, and then linearly interpolate the noises to synthesize images shown in the middle. All images are generated at 64×64 in **a single step**, and upscaled to 256×256 in **a single step** given the input noise.

D.4. Training Details

All models for all the tasks are trained on the same resources of 8 NVIDIA A100 GPUs for 500K updates. Training roughly takes $3 \sim 7$ days to converge depending on the model sizes. We train all our models with the AdamW (Loshchilov & Hutter, 2017) optimizer, with no learning rate decay or warm-up, and no weight decay. Standard EMA to the weights is also applied for student models. Since our methods are data-free, there is no additional overhead on data storage and loading except for the text prompts, which are much smaller and can be efficiently loaded into memory.

Learning the boundary loss requires additional NFEs during each training step. In practice, we apply the boundary loss less frequently (e.g., computing the boundary condition every 4 iterations and setting the loss to be 0 otherwise) to improve the overall training efficiency. Note that distilling from the class-conditioned / text-to-image teachers requires multiple forward passes due to CFG, which relatively slows down the training compared to unconditional models.

Distilling from the DeepFloyd IF teacher requires learning from two stages. In this paper, we can easily achieve that by first distilling the first-stage model into single-step with BOOT, and then distilling the upscaler model based on the output of the first-stage student. Following the original paper (Saharia et al., 2022), noise augmentation is also applied on the first-stage output where we set the noise-level as 250 [*]. For more training hyperparameters, please refer to Table 2.

E. Additional Analysis

E.1. Comparison of Learned Trajectory

We show a comparison of the original diffusion path, the proposed signal ODE and the output of our single-step distilled model with different target times in Figure 4.

E.2. Effect of CFG Weights

In Figure 11, we also show an example of the learned model with various CFG weight as inputs. Similar to the findings in (Meng et al., 2022), the BOOT student can easily learn to adapt the teacher distribution based on the guidance strength (w).

[*]https://github.com/huggingface/diffusers/blob/main/src/diffusers/pipelines/deepfloyd_if/pipeline_if_superresolution.py#L715

Figure 13. With fixed noise, we can perform controllable generation by swapping the keywords from the prompts. The prompts are chosen from the combination of *portrait of a {owl, raccoon, tiger, fox, llama, gorilla, panda} wearing { a t-shirt, a jacket, glasses, a crown} { drinking a latte, eating a pizza, reading a book, holding a cake} cinematic, hdr.* All images are generated from the student distilled from IF teacher given the same noise input at 64×64 in **a single step**, and upscaled to 256×256 in **a single step** given the same noise input.

E.3. Latent Space Interpolation

We show a visualization of latent space interpolation results in Figure 12, where the student model is distilled from the pretrained IF teacher. The smooth transition of the results demonstrates that the distilled student model has learned a smooth and continuous latent space, enabling potential applications such as video generation and semantic-level editing.

E.4. Text-controlled Generation

In Figure 13, we show an example of controllable generation by fixing the noise input and only edit on the prompts. Similar to the original diffusion teacher model, the BOOT distilled student retains all abilities of disentangled representation which supports fine-grained control with consistent styles.

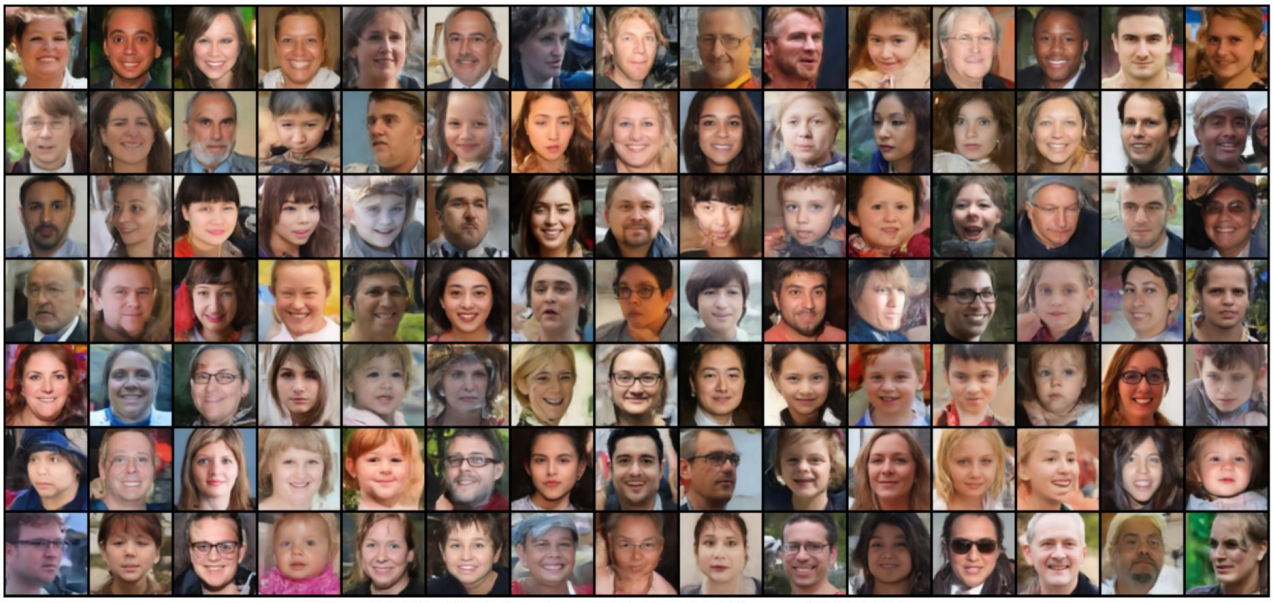
F. Limitations and Future Work

BOOT is a knowledge distillation algorithm, which by nature requires a pre-trained teacher model. Also by design, the sampling quality of BOOT is upper bounded by that of the teacher. Besides, BOOT may produce lower quality samples compared to other distillation methods (Song et al., 2023; Berthelot et al., 2023) in settings where ground-truth data are easy to use, which can potentially be remedied by combining methods. In future work, we would like to explore jointly training the teacher and the student such that diffusion is "distillation aware". Beyond that, it would be interesting to consider training a single-step diffusion model from scratch with a similar form of BOOT.

G. Additional Samples from BOOT

Finally, we provide additional qualitative comparisons for the unconditional models of FFHQ 64×64 (Figure 14), LSUN 256×256 (Figure 15), the class-conditional model of ImageNet 64×64 (Figure 16), and comparisons for text-to-image generation based on DeepFloyd-IF (64×64 in Figures 17 and 20, 256×256 in Figure 1) and StableDiffusion (512×512

(a) Diffusion Model Teacher (50 steps / 1.2 FPS)



(b) Distilled Student with BOOT (1 step / 54 FPS)

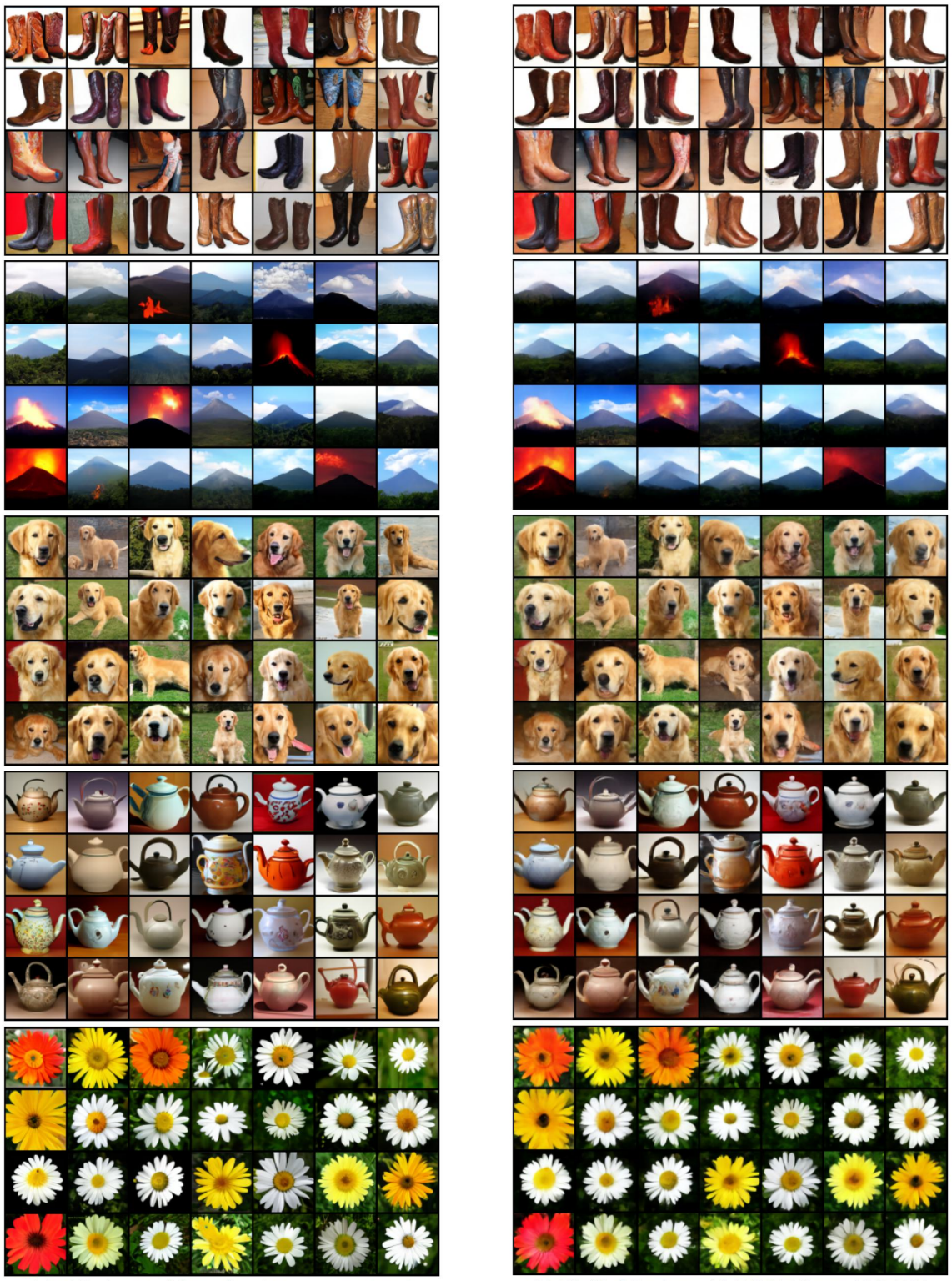
Figure 14. Uncurated samples from FFHQ 64×64. All corresponding samples use the same initial noise for the DDIM teacher and the single-step BOOT student.

in Figures 19 and 21).

(a) Diffusion Model Teacher (50 steps / 0.6 FPS)

(b) Distilled Student with BOOT (1 step / 32 FPS)

Figure 15. Uncurated samples from LSUN Bedroom 256×256. All corresponding samples use the same initial noise for the DDIM teacher and the single-step BOOT student.



(a) Diffusion Model Teacher (50 steps / 0.3 FPS)

(b) Distilled Student with BOOT (1 step / 34 FPS)

Figure 16. Uncurated class-conditioned samples from ImageNet 64×64. All corresponding samples use the same initial noise for the DDIM teacher and the single-step BOOT student. Classes from top to bottom: *cowboy boot, volcano, golden retriever, teapot, daisy*. The diffusion model uses CFG with $w = 3$, and our student model conditions on the same weight.

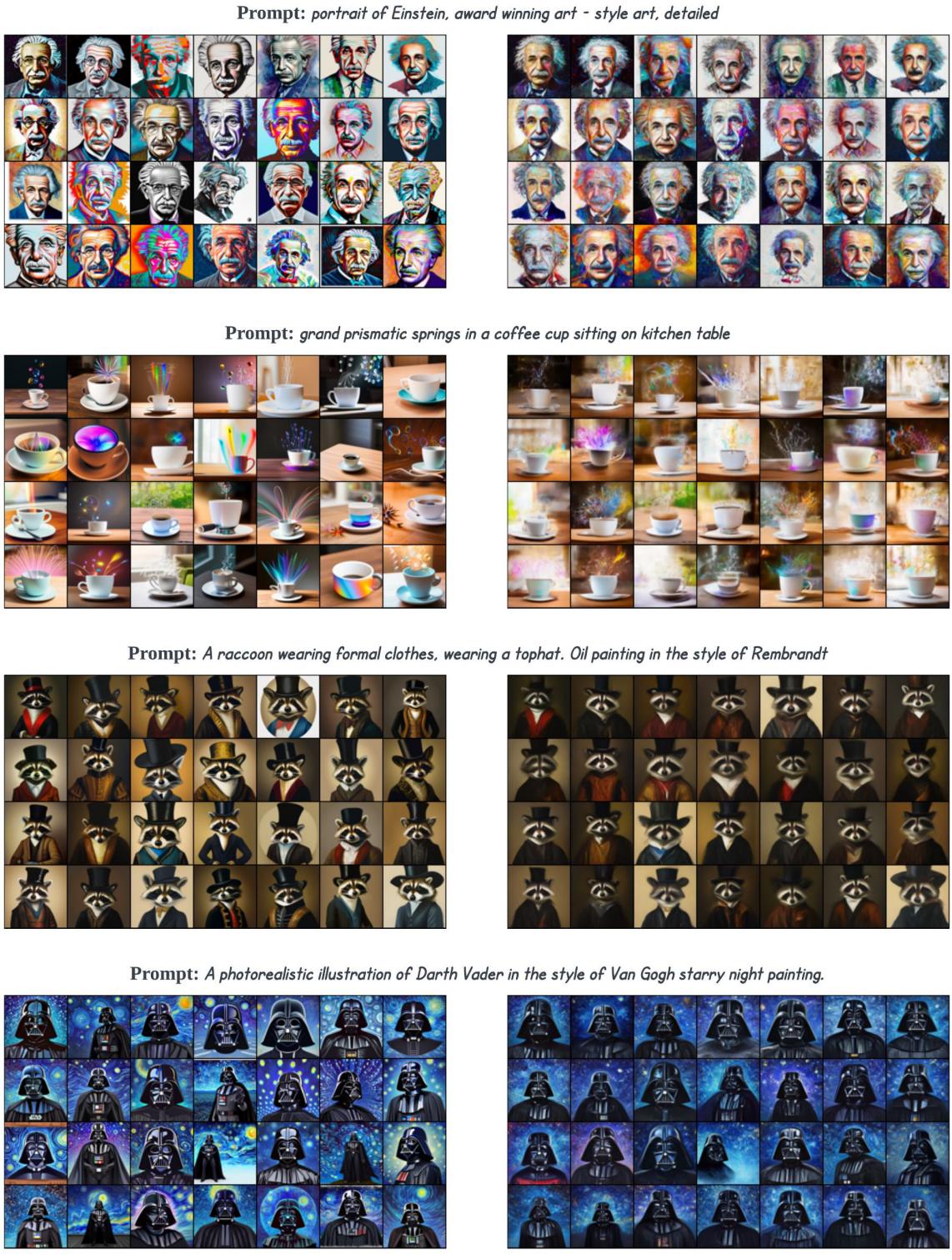


Figure 17. Uncurated text-conditioned image generation distilled from DeepFloyd IF (the first stage model, images are at 64×64). All corresponding samples use the same initial noise for the DDIM teacher and the single-step BOOT student. The specific prompts are shown above the images.

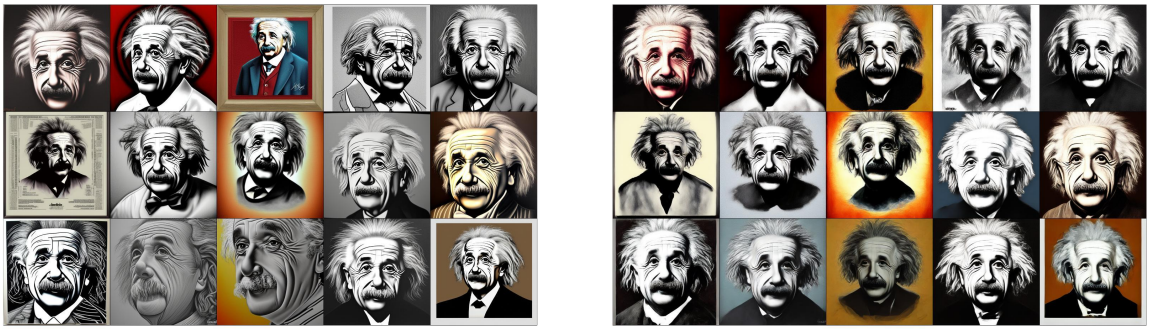
(a) First-stage Distilled Student Output + Bilinear Upsample + Noise Augment

(b) Diffusion Model Teacher (50 Steps + CFG)

(c) Distilled Student with BOOT (1 Step)

Figure 18. Given the 64×64 outputs from Figure 17, we also show comparison for the second-stage models which upscale the images to 256×256. All corresponding samples use the same initial noise for the DDIM teacher and the single-step BOOT student.

Prompt: *portrait of Einstein, award winning art - style art, detailed*



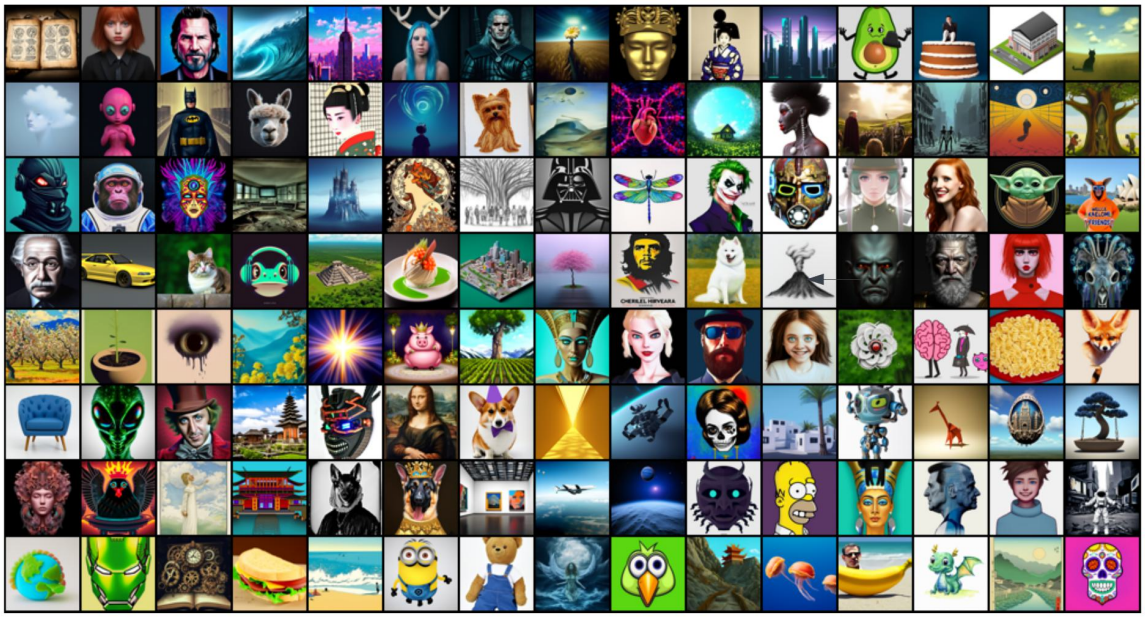
Prompt: *grand prismatic springs in a coffee cup sitting on kitchen table*

Prompt: *A raccoon wearing formal clothes, wearing a tophat. Oil painting in the style of Rembrandt*

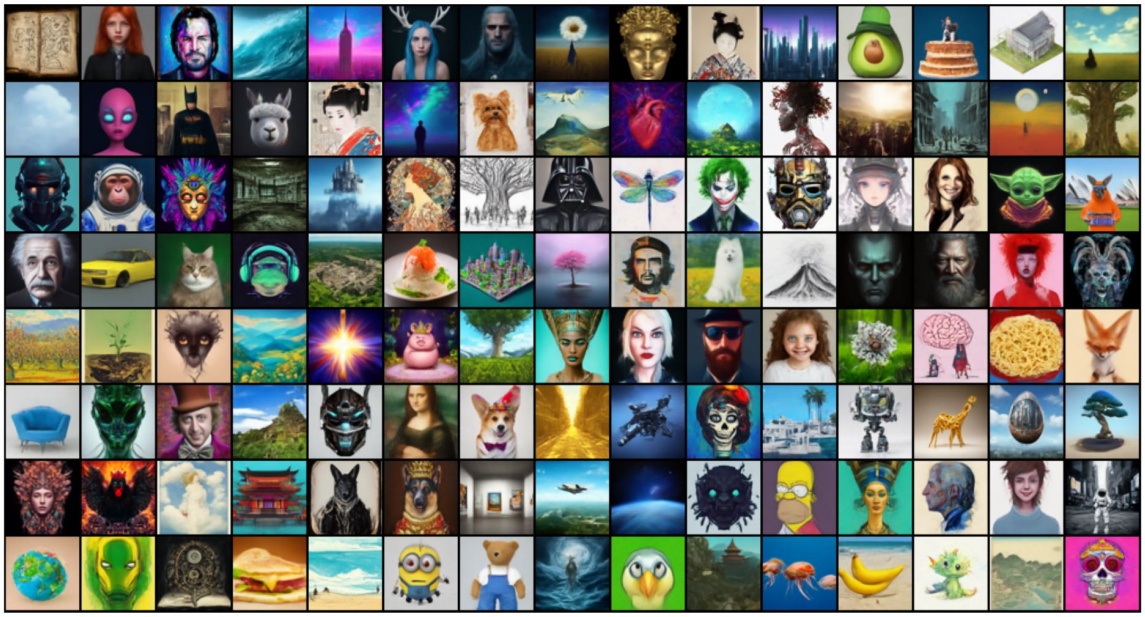
Prompt: *A photorealistic illustration of Darth Vader in the style of Van Gogh starry night painting.*

(a) Diffusion Model Teacher (50 steps) **(b) Distilled Student with BOOT (1 step)**

Figure 19. Uncurated text-conditioned image generation distilled from StableDiffusion (latent diffusion in 64×64, images are upscaled to 512×512 with the pre-trained VAE decoder). All corresponding samples use the same initial noise for the DDIM teacher and the single-step BOOT student. We use the same prompts as in Figure 17 for better comparison.

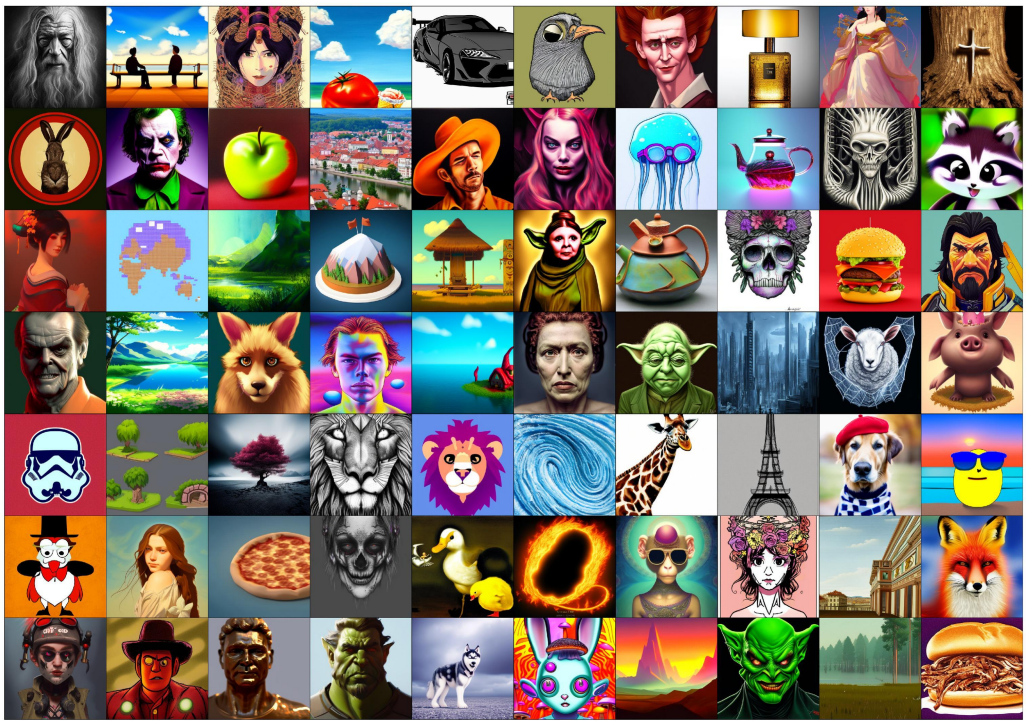


(a) Random Samples from Diffusion Model Teacher (50 steps with CFG)

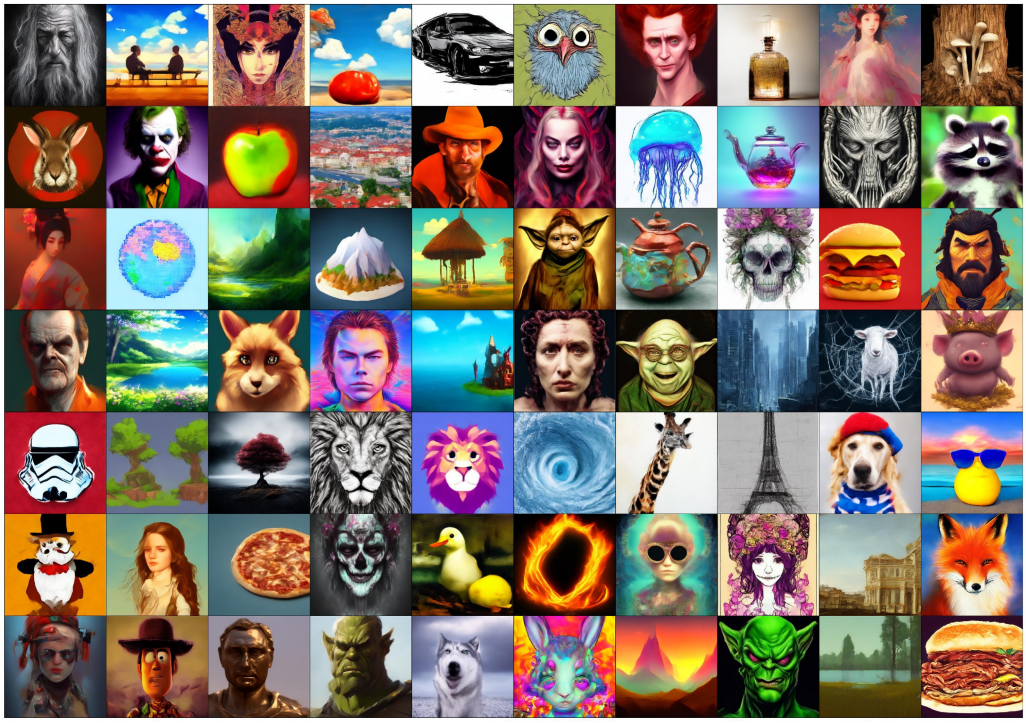


(b) Random Samples from Distilled Student with BOOT (1 step)

Figure 20. Uncurated text-conditioned image generation distilled from DeepFloyd IF (the first stage model, images are at 64×64) given sampled text prompts from *diffusiondb* (Wang et al., 2022) randomly. All corresponding samples use the same initial noise for the DDIM teacher and the single-step student. Besides, we also show curated examples from the two-stage distilled model at 256×256 in Figure 1.



(a) Random Samples from Diffusion Model Teacher (50 steps with CFG)



(b) Random Samples from Distilled Student with BOOT (1 step)

Figure 21. Uncurated text-conditioned image generation distilled from StableDiffusion (latent diffusion in 64×64, images are upscaled to 512×512 with the pre-trained VAE decoder) given sampled text prompts from *diffusiondb* (Wang et al., 2022) randomly. All corresponding samples use the same initial noise for the DDIM teacher and the single-step student.