Improved Distribution Matching Distillation for Fast Image Synthesis

Tianwei Yin[1] **Michaël Gharbi**[2] **Taesung Park**[2] **Richard Zhang**[2]
Eli Shechtman[2] **Frédo Durand**[1] **William T. Freeman**[1]

[1]Massachusetts Institute of Technology [2]Adobe Research

https://tianweiy.github.io/dmd2/

Abstract

Recent approaches have shown promises distilling expensive diffusion models into efficient one-step generators. Amongst them, Distribution Matching Distillation (DMD) produces one-step generators that match their teacher in *distribution*, i.e., the distillation process does not enforce a one-to-one correspondence with the sampling trajectories of their teachers. However, to ensure stable training in practice, DMD requires an additional regression loss computed using a large set of noise–image pairs, generated by the teacher with many steps of a deterministic sampler. This is not only computationally expensive for large-scale text-to-image synthesis, but it also limits the student's quality, tying it too closely to the teacher's original sampling paths. We introduce DMD2, a set of techniques that lift this limitation and improve DMD training. First, we eliminate the regression loss and the need for expensive dataset construction. We show that the resulting instability is due to the "fake" critic not estimating the distribution of generated samples with sufficient accuracy and propose a two time-scale update rule as a remedy. Second, we integrate a GAN loss into the distillation procedure, discriminating between generated samples and real images. This lets us train the student model on *real* data, thus mitigating the imperfect "real" score estimation from the teacher model, and thereby enhancing quality. Third, we introduce a new training procedure that enables multi-step sampling in the student, and addresses the training–inference input mismatch of previous work, by simulating inference-time generator samples during training. Taken together, our improvements set new benchmarks in one-step image generation, with FID scores of 1.28 on ImageNet-64×64 and 8.35 on zero-shot COCO 2014, surpassing the original teacher despite a $500\times$ reduction in inference cost. Further, we show our approach can generate megapixel images by distilling SDXL, demonstrating exceptional visual quality among few-step methods, and surpassing the teacher. We release our code and pretrained models.

1 Introduction

Diffusion models have achieved unprecedented quality in visual generation tasks [1–8]. But their sampling procedure typically requires dozens of iterative denoising steps, each of which is a forward pass through a neural network. This makes high resolution text-to-image synthesis slow and expensive. To address this issue, numerous distillation methods have been developed to convert a teacher diffusion model into an efficient, few-step student generator [9–20]. However, they often result in degraded quality, as the student model is typically trained with a loss to learn the pairwise noise-to-image mapping of the teacher, but struggles to perfectly mimic its behavior.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-1505

Figure 1: 1024×1024 samples produced by our 4-step generator distilled from SDXL. Please zoom in for details.

Nevertheless, it should be noted that loss functions aimed at matching distributions, such as the GAN [21] or the DMD [22] loss, are not burdened with the complexity of precisely learning the specific paths from noise to image because their goal is to align with the teacher model in terms of *distribution*—by minimizing either a Jensen-Shannon (JS) or an approximate Kullback-Leibler (KL) divergence between the student and teacher output distributions.

In particular, DMD [22] has demonstrated state-of-the-art results in distilling Stable Diffusion 1.5, yet it remains less investigated than GAN-based methods [23–29]. A likely reason is that DMD still requires an additional regression loss to ensure stable training. In turn, this necessitates creating millions of noise-image pairs by running the full sampling steps of the teacher model, which is particularly costly for text-to-image synthesis. The regression loss also negates the key benefit of DMD's unpaired distribution matching objective, because it causes the student's quality to be upper-bounded by the teacher's.

In this paper, we show how to do away with DMD's regression loss, without compromising training stability. We then push the limits of distribution matching by integrating the GAN framework into DMD, and enable few-steps sampling with a novel training procedure, which we termed 'backward simulation'. Taken together, our contributions lead to state-of-the-art fast generative models that outperform their teacher, using as few as 4 sampling steps. Our method, which we call DMD2,

achieves state-of-the-art results in one-step image generation, setting a new benchmark with FID scores of 1.28 on ImageNet-64×64 and 8.35 on zero-shot COCO 2014. We demonstrate our approach's scalability by distilling from SDXL to produce high-quality megapixel images, establishing new standards among few-step methods.

In short, our contributions are as follows:

- We propose a new distribution matching distillation technique that does not require a regression loss for stable training, thereby eliminating the need for costly data collection, and allowing for more flexible and scalable training.
- We show that training instability in DMD [22] without regression loss stems from an insufficiently trained *fake diffusion critic*, and implement a two time-scale update rule to address this issue.
- We integrate a GAN objective into the DMD framework, where the discriminator is trained to distinguish samples from the student generator vs. *real* images. This additional supervision operates at the *distribution* level, which better aligns with DMD's distribution-matching philosophy than the original regression loss. It mitigates approximation errors in the teacher diffusion model and enhances image quality.
- While the original DMD only supports one-step students, we introduce a technique to support multi-step generators. Unlike previous multi-step distillation methods, we avoid the domain mismatch between training and inference by simulating inference-time generator inputs during training, thus improving overall performance.

2 Related Work

Diffusion Distillation. Recent diffusion acceleration techniques have focused on speeding up the generation process through distillation [9, 10, 13–20, 22, 23, 30]. They typically train a generator to approximate the ordinary differential equation (ODE) sampling trajectory of a teacher model, in fewer sampling steps. Notably, Luhman et al. [16] precompute a dataset of noise and images pairs, generated by the teacher using an ODE sampler, and use it to train the student to regress the mapping in a single network evaluation. Follow-up works like Progressive Distillation [10, 13] eliminate the need to precompute this paired dataset offline. They iteratively train a sequence of student models, each halving the number of sampling steps of its predecessor. A complementary technique, Instaflow [11] straightens the ODE trajectories, so they are easier to approximate with a one-step student. Consistency Distillation [9,12,19,26,31,32], and TRACT [33], train student models so their outputs are self-consistent at any timesteps along the ODE trajectory, and thus consistent with the teacher.

GANs. Another line of research employs adversarial training to align the student with the teacher at a broader distribution level. In ADD [23], the generator, initialized with weights from a diffusion model, is trained using a projected GAN objective with an image-space classifier [34]. Building on this, LADD [24] utilizes a pre-trained diffusion model as the discriminator and operates in latent space, thus improving scalability and enabling higher-resolution synthesis. Inspired by DiffusionGAN [28, 29], UFOGen [25] introduces noise injection prior to the *real* vs. *fake* classification in the discriminator, to smooth out the distributions, which stabilizes the training dynamics. However, purely GAN-based methods often struggle to integrate classifier-free guidance directly. For instance, LADD uses diffusion-generated images with classifier-free guidance as real data in its GAN discriminator. Other approaches combine adversarial objectives with a distillation loss to preserve the original guided sampling trajectory. For instance, SDXL-Lightning [27] integrates a DiffusionGAN loss [25] with a progressive distillation objective [10, 13], while the Consistency Trajectory Model [26] combines a GAN [35] with an improved consistency distillation [9]. In contrast, our approach based on distribution matching [22, 36, 37] inherently integrates classifier-free guidance into the training supervision, significantly simplifying the training process.

Score Distillation was initially introduced in the context of text-to-3D synthesis [37–40], utilizing a pre-trained text-to-image diffusion model as a distribution matching loss. These methods optimize a 3D object by aligning rendered views with a text-conditioned image distribution, using the scores predicted by a pretrained diffusion model. Recent works have extended score distillation [37, 38, 41–43] to diffusion distillation [22, 30, 36, 44–46]. Notably, DMD [22] minimizes an approximate KL

 https://doi.org/10.52202/079017-1505

Figure 2: 1024×1024 samples produced by our 4-step generator distilled from SDXL. Please zoom in for details.

divergence, with its gradient represented as the difference between two score functions: one, fixed and pretrained, for the target distribution and another, trained dynamically, for the output distribution of the generator.

DMD parameterizes both score functions using diffusion models. This training objective proved more stable than GAN-based methods and has demonstrated superior performance in one-step image synthesis. An important caveat, DMD requires a regression loss for stability, calculated using precomputed noise-image pairs, similar to Luhman et al. [16]. Our work does away with this requirement. We introduce techniques to stabilize the DMD training procedure without the regression regularizer, thus significantly reducing the computational costs incurred by paired data precomputation. Furthermore, we extend DMD to support multi-step generation and integrate the strengths of both GANs and distribution matching approaches [22, 30, 36, 45], leading to state-of-the-art results in text-to-image synthesis.

3 Background: Diffusion and Distribution Matching Distillation

This section gives a brief overview of diffusion models and distribution matching distillation (DMD).

Diffusion Models generate images through iterative denoising. In the forward diffusion process, noise is progressively added to corrupt a sample $x \sim p_{\text{real}}$ from the data distribution into pure Gaussian noise over a predetermined number of steps T, so that, at each timestep t, the diffused samples follow the distribution $p_{\text{real},t}(x_t) = \int p_{\text{real}}(x)q(x_t|x)dx$, with $q_t(x_t|x) \sim \mathcal{N}(\alpha_t x, \sigma_t^2 \mathbf{I})$, where $\alpha_t, \sigma_t > 0$ are scalars determined by the noise schedule [47, 48]. The diffusion model learns to iteratively reverse the corruption process by predicting a denoised estimate $\mu(x_t, t)$, conditioned on the current noisy sample x_t and the timestep t, ultimately leading to an image from the data distribution p_{real}. After training, the denoised estimate relates to the gradient of the data likelihood function, or score function [48] of the diffused distribution:

$$s_{\text{real}}(x_t, t) = \nabla_{x_t} \log p_{\text{real},t}(x_t) = -\frac{x_t - \alpha_t \mu_{\text{real}}(x_t, t)}{\sigma_t^2}. \tag{1}$$

Sampling an image typically requires dozens to hundreds of denoising steps [49–52].

Distribution Matching Distillation (DMD) distills a many-step diffusion models into a one-step generator G [22] by minimizing the expectation over t of approximate Kullback-Liebler (KL) divergences between the diffused target distribution $p_{\text{real},t}$ and the diffused generator output distribution $p_{\text{fake},t}$. Since DMD trains G by gradient descent, it only requires the gradient of this loss, which can be computed as the difference of 2 score functions:

$$\nabla \mathcal{L}_{\text{DMD}} = \mathbb{E}_t \left(\nabla_\theta \text{KL}(p_{\text{fake},t} \| p_{\text{real},t}) \right) = -\mathbb{E}_t \left(\int \left(s_{\text{real}}(F(G_\theta(z), t), t) - s_{\text{fake}}(F(G_\theta(z), t), t) \right) \frac{dG_\theta(z)}{d\theta} \, dz \right), \tag{2}$$

where $z \sim \mathcal{N}(0, \mathbf{I})$ is a random Gaussian noise input, θ are the generator parameters, F is the forward diffusion process (i.e., noise injection) with noise level corresponding to time step t, and s_{real} and s_{fake} are scores approximated using diffusion models μ_{real} and μ_{fake} trained on their respective distributions (Eq. (1)). DMD uses a frozen pre-trained diffusion model as μ_{real} (the teacher), and dynamically updates μ_{fake} while training G, using a denoising score-matching loss on samples from the one-step generator, i.e., fake data [22, 47].

Yin et al. [22] found that an additional regression term [16] was needed to regularize the distribution matching gradient (Eq. (2)) and achieve high-quality one-step models. For this, they collect a dataset of noise-image pairs (z, y) where the image y is generated using the teacher diffusion model, and a *deterministic* sampler [49, 50, 53], starting from the noise map z. Given the same input noise z, the regression loss compares the generator output with the teacher's prediction:

$$\mathcal{L}_{\text{reg}} = \mathbb{E}_{(z,y)} d(G_\theta(z), y), \tag{3}$$

where d is a distance function, such as LPIPS [54] in their implementation. While gathering this data incurs negligible cost for small datasets like CIFAR-10, it becomes a significant bottleneck with large-scale text-to-image synthesis tasks, or models with complex conditioning [55–57]. For instance, generating one noise-image pair for SDXL [58] takes around 5 seconds, amounting to about 700 A100 days to cover the 12 million prompts in the LAION 6.0 dataset [59], as utilized by Yin et al. [22]. This dataset construction cost alone is already more than $4\times$ our total training compute (as

https://doi.org/10.52202/079017-1505

detailed in Appendix J). This regularization objective is also at odds with DMD's goal of matching the student and teacher in *distribution*, since it encourages adherence to the teacher's sampling paths.

4 Improved Distribution Matching Distillation

We revisit multiple design choices in the DMD algorithm [22] and identify significant improvements.

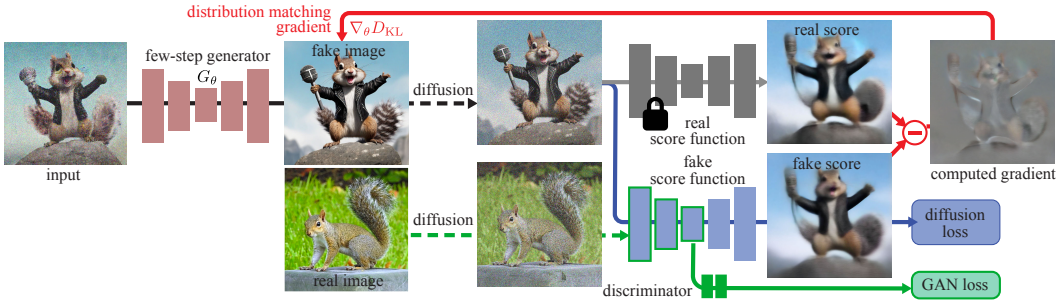

Figure 3: Our method distills a costly diffusion model (gray, right) into a one- or multi-step generator (red, left). Our training alternates between 2 steps: 1. optimizing the generator using the gradient of an implicit distribution matching objective (red arrow) and a GAN loss (green), and 2. training a score function (blue) to model the distribution of "fake" samples produced by the generator, as well as a GAN discriminator (green) to discriminate between fake samples and real images. The student generator can be a one-step or a multi-step model, as shown here, with an intermediate step input.

4.1 Removing the regression loss: true distribution matching and easier large-scale training

The regression loss [16] used in DMD [22] ensures mode coverage and training stability, but as we discussed in Section 3, it makes large-scale distillation cumbersome, and is at odds with the distribution matching idea, thus inherently limiting the performance of the distilled generator to that of the teacher model. Our first improvement is to remove this loss.

4.2 Stabilizing pure distribution matching with a Two Time-scale Update Rule

Naively omitting the regression objective, shown in Eq. (3), from DMD leads to training instabilities and significantly degrades quality (Tab. 3). For example, we observed that the average brightness, along with other statistics, of generated samples fluctuates significantly, without converging to a stable point (See Appendix G). We attribute this instability to approximation errors in the fake diffusion model μ_{fake}, which does not track the fake score accurately, since it is dynamically optimized on the non-stationary output distribution of the generator. This causes approximation errors and biased generator gradients (as also discussed in [30]).

We address this using the two time-scale update rule inspired by Heusel et al. [60]. Specifically, we train μ_{fake} and the generator G at different frequencies to ensure that μ_{fake} accurately tracks the generator's output distribution. We find that using 5 fake score updates per generator update, without the regression loss, provides good stability and matches the quality of the original DMD on ImageNet (Tab. 3) while achieving much faster convergence. Further analysis are included in Appendix G.

4.3 Surpassing the teacher model using a GAN loss and real data

Our model so far achieves comparable training stability and performance to DMD [22] without the need for costly dataset construction (Tab. 3). However, a performance gap remains between the distilled generator and the teacher diffusion model. We hypothesize this gap could be attributed to approximation errors in the real score function μ_{real} used in DMD, which would propagate to the generator and lead to suboptimal results. Since DMD's distilled model is never trained with real data, it cannot recover from these errors.

We address this issue by incorporating an additional GAN objective into our pipeline, where the discriminator is trained to distinguish between *real* images and images produced by our generator.

Trained using real data, the GAN classifier does not suffer from the teacher's limitation, potentially allowing our student generator to surpass it in sample quality. Our integration of a GAN classifier into DMD follows a minimalist design: we add a classification branch on top of the bottleneck of the fake diffusion denoiser (see Fig. 3). The classification branch and upstream encoder features in the UNet are trained by maximizing the standard non-saturing GAN objective:

$$\mathcal{L}_{\text{GAN}} = \mathbb{E}_{x\sim p_{\text{real}}, t\sim[0,T]}[\log D(F(x,t))] + \mathbb{E}_{z\sim p_{\text{noise}}, t\sim[0,T]}[-\log(D(F(G_\theta(z),t)))], \tag{4}$$

where D is the discriminator, and F is the forward diffusion process (i.e., noise injection) defined in Section 3, with noise level corresponding to time step t. The generator G minimizes this objective. Our design is inspired by prior works that use diffusion models as discriminators [24, 25, 27]. We note that this GAN objective is more consistent with the distribution matching philosophy since it does not require paired data, and is independent of the teacher's sampling trajectories.

4.4 Multi-step generator

With the proposed improvements, we are able to match the performance of teacher diffusion models on ImageNet and COCO (see Tab. 1 and Tab. 5). However, we found that larger scale models like SDXL [58] remain challenging to distill into a one-step generator because of limited model capacity and a complex optimization landscape to learn the direct mapping from noise to highly diverse and detailed images. This motivated us to extend DMD to support multi-step sampling.

We fix a predetermined schedule with N timestep $\{t_1, t_2, \dots t_N\}$, identical during training and inference. During inference, at each step, we alternate between denoising and noise injection steps, following the consistency model [9], to improve sample quality. Specifically, starting from Gaussian noise $z_0 \sim \mathcal{N}(0, \mathbf{I})$, we alternate between denoising updates $\hat{x}_{t_i} = G_\theta(x_{t_i}, t_i)$, and forward diffusion steps $x_{t_{i+1}} = \alpha_{t_{i+1}}\hat{x}_{t_i} + \sigma_{t_{i+1}}\epsilon$ with $\epsilon \sim \mathcal{N}(0, \mathbf{I})$, until we obtain our final image $\hat{x}_{t_N}$. Our 4-step model uses the following schedule: 999, 749, 499, 249, for a teacher model trained with 1000 steps.

4.5 Multi-step generator simulation to avoid training/inference mismatch

Previous multi-step generators are typically trained to denoise noisy *real* images [23, 24, 27]. However, during inference, except for the first step, which starts from pure noise, the generator's input come from a previous generator sampling step $\hat{x}_{t_i}$. This creates a training-inference mismatch that adversely impacts quality (Fig. 4). We address this issue by replacing the noisy real images during training, with noisy synthetic images x_{t_i} produced by the current student generator running several steps, similar to our inference pipeline (§ 4.4). This is tractable because, unlike the teacher diffusion model, our generator only runs for a few steps. Our generator then denoises these simulated images and the outputs are supervised with the proposed loss functions. Using noisy synthetic images avoids the mismatch and improves overall performance (See Sec. 5.3).

Figure 4: Most multi-step distillation methods simulate intermediate steps using forward diffusion during training (left). This creates a mismatch with the inputs the model sees during inference. Our proposed solution (right) remedies the problem by simulating the inference-time backward process during training.

A concurrent work, Imagine Flash [61], proposed a similar technique. Their backward distillation algorithm shares our motivation of reducing the training and testing gap by using the student-generated images as the input to the subsequent sampling steps at training time. However, they do not entirely resolve the mismatch issue, because the teacher model of the regression loss now suffers from the training–test gap: it is never trained with synthetic images. This error is accumulated along the sampling path. In contrast, our distribution matching loss is not dependent on the input to the student model, alleviating this issue.

https://doi.org/10.52202/079017-1505

4.6 Putting everything together

In summary, our distillation method lifts DMD [22] stringent requirements for precomputed noise–image pairs. It further integrates the strength of GANs and supports multi-step generators. As shown in Fig. 3, starting from a pretrained diffusion model, we alternate between optimizing the generator G_θ to minimize the original distribution matching objective as well as a GAN objective, and optimizing the fake score estimator μ_{fake} using both a denoising score matching objective on the fake data, and the GAN classification loss. To ensure the fake score estimate is accurate and stable, despite being optimized on-line, we update it with higher frequency than the generator (5 steps vs. 1). A comparison of the training algorithms between DMD and DMD2 (Ours) is shown in Appendix Alg. 7.

5 Experiments

We evaluate our approach, DMD2, using several benchmarks, including class-conditional image generation on ImageNet-64×64 [62], and text-to-image synthesis on COCO 2014 [63] with various teacher models [1, 58]. We use the Fréchet Inception Distance (FID) [60] to measure image quality and diversity, and the CLIP Score [64] to evaluate text-to-image alignment. For SDXL models, we additionally report patch FID [27, 65], which measures FID on 299x center-cropped patches of each image, to assess high-resolution details. Finally, we conduct human evaluations to compare our approach with other state-of-the-art methods. Comprehensive evaluations confirm that distilled models trained using our approach outperform previous work, and even rival the performance of the teacher models. Detailed training and evaluation procedures are provided in the appendix.

5.1 Class-conditional Image Generation

Table 1 compares our model with recent baselines on ImageNet-64×64. With a single forward pass, our method significantly outperforms existing distillation techniques and even outperforms the teacher model using ODE sampler [53]. We attribute this remarkable performance to the removal of DMD's regression loss (Sec. 4.1 and 4.2), which eliminates the performance upper bound imposed by the ODE sampler, as well as our additional GAN term (Sec. 4.3), which mitigates the adverse impact of the teacher diffusion model's score approximation error.

Table 1: Image quality comparison on ImageNet-64×64.

Method	# Fwd Pass (↓)	FID (↓)
BigGAN-deep [66]	1	4.06
ADM [67]	250	2.07
RIN [68]	1000	1.23
StyleGAN-XL [35]	1	1.52
Progress. Distill. [10]	1	15.39
DFNO [69]	1	7.83
BOOT [20]	1	16.30
TRACT [33]	1	7.43
Meng et al. [13]	1	7.54
Diff-Instruct [36]	1	5.57
Consistency Model [9]	1	6.20
iCT-deep [12]	1	3.25
CTM [26]	1	1.92
DMD [22]	1	2.62
DMD2 (Ours)	1	**1.51**
+longer training (Ours)	1	**1.28**
EDM (Teacher, ODE) [53]	511	2.22
EDM (Teacher, SDE) [53]	511	1.36

Table 2: Image quality comparison with SDXL backbone on 10K prompts from COCO 2014.

Method	# Fwd Pass (↓)	FID (↓)	Patch FID (↓)	CLIP (↑)
LCM-SDXL [32]	1	81.62	154.40	0.275
	4	22.16	33.92	0.317
SDXL-Turbo [23]	1	24.57	23.94	**0.337**
	4	23.19	23.27	0.334
SDXL Lightning [27]	1	23.92	31.65	0.316
	4	24.46	24.56	0.323
DMD2 (Ours)	1	**19.01**	26.98	0.336
	4	19.32	**20.86**	0.332
SDXL Teacher, cfg=6 [58]	100	19.36	21.38	0.332
SDXL Teacher, cfg=8 [58]	100	20.39	23.21	0.335

5.2 Text-to-Image Synthesis

We evaluate DMD2's text-to-image generation performance on zero-shot COCO 2014 [63]. Our generators are trained by distilling SDXL [58] and SD v1.5 [1], respectively, using a subset of 3 million prompts from LAION-Aesthetics [59]. Additionally, we collect 500k images from LAION-Aesthetic as training data for the GAN discriminator. Table 2 summarizes distillation results for the SDXL model. Our 4-step generator produces high quality and diverse samples, achieving a FID score of 19.32 and a CLIP score of 0.332, rivaling the teacher diffusion model for both image quality and prompt coherence. To further verify our method's effectiveness, we conduct an extensive user study comparing our model's output with those from the teacher model and existing distillation methods. We use a subset of 128 prompts from PartiPrompts [70] following LADD [24]. For each comparison, we ask a random set of five evaluators to choose the image that is more visually appealing, as well as the one that better represents the text prompt. Details about the human evaluation are included in Appendix L. As shown in Figure 5, our model achieves much higher user preferences than baseline approaches. Notably, our model outperforms its teacher in image quality for 24% of samples and achieves comparable prompt alignment, while requiring $25\times$ fewer forward passes (4 vs 100). Qualitative comparisons are shown in Figure 6. Results for SDv1.5 are provided in Table 5 in Appendix D. Similarly, one-step model trained using DMD2 outperforms all previous diffusion acceleration approaches, achieving a FID score of 8.35, representing a significant 3.14-point improvement over the original DMD method [22]. Our results also surpass the teacher models that uses a 50-step PNDM sampler [50].

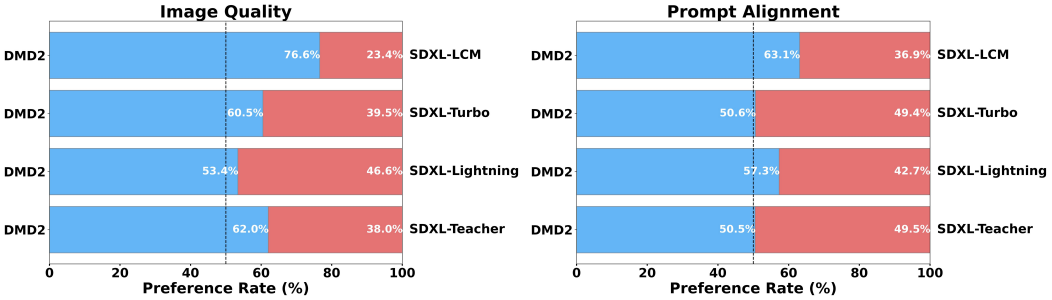

Figure 5: User study comparing our distilled model with its teacher and competing distillation baselines [23, 27, 31]. All distilled models use 4 sampling steps, the teacher uses 50. Our model achieves the best performance for both image quality and prompt alignment.

5.3 Ablation Studies

Table 3: Ablation studies on ImageNet. TTUR stands for two-timescale update rule.

DMD	No Regress.	TTUR	GAN	FID (↓)
✓				2.62
✓	✓			3.48
✓	✓	✓		2.61
✓	✓	✓	✓	**1.51**
			✓	2.56
		✓	✓	2.52

Table 4: Ablation studies with SDXL backbone on 10K prompts from COCO 2014.

Method	FID (↓)	Patch FID (↓)	CLIP (↑)
w/o GAN	26.90	27.66	0.328
w/o Distribution Matching	**13.77**	27.96	0.307
w/o Backward Simulation	20.66	24.21	0.332
DMD2 (Ours)	19.32	**20.86**	**0.332**

Table 3 ablates different components of our proposed method on ImageNet. Simply removing the ODE regression loss from the original DMD results in a degraded FID of 3.48 due to training instability (see further analysis in Appendix G). However, incorporating our Two Time-scale Update Rule (TTUR, Sec. 4.2) mitigates this performance drop, matching the DMD baseline performance without requiring additional dataset construction. Adding our GAN loss achieves a further 1.1-point improvement in FID. Our integrated approach surpasses the performance of using GAN alone (without distribution matching objective), and adding the two-timescale update rule to GAN alone does not improve it, highlighting the effectiveness of combining distribution matching with GANs in a unified framework.

https://doi.org/10.52202/079017-1505

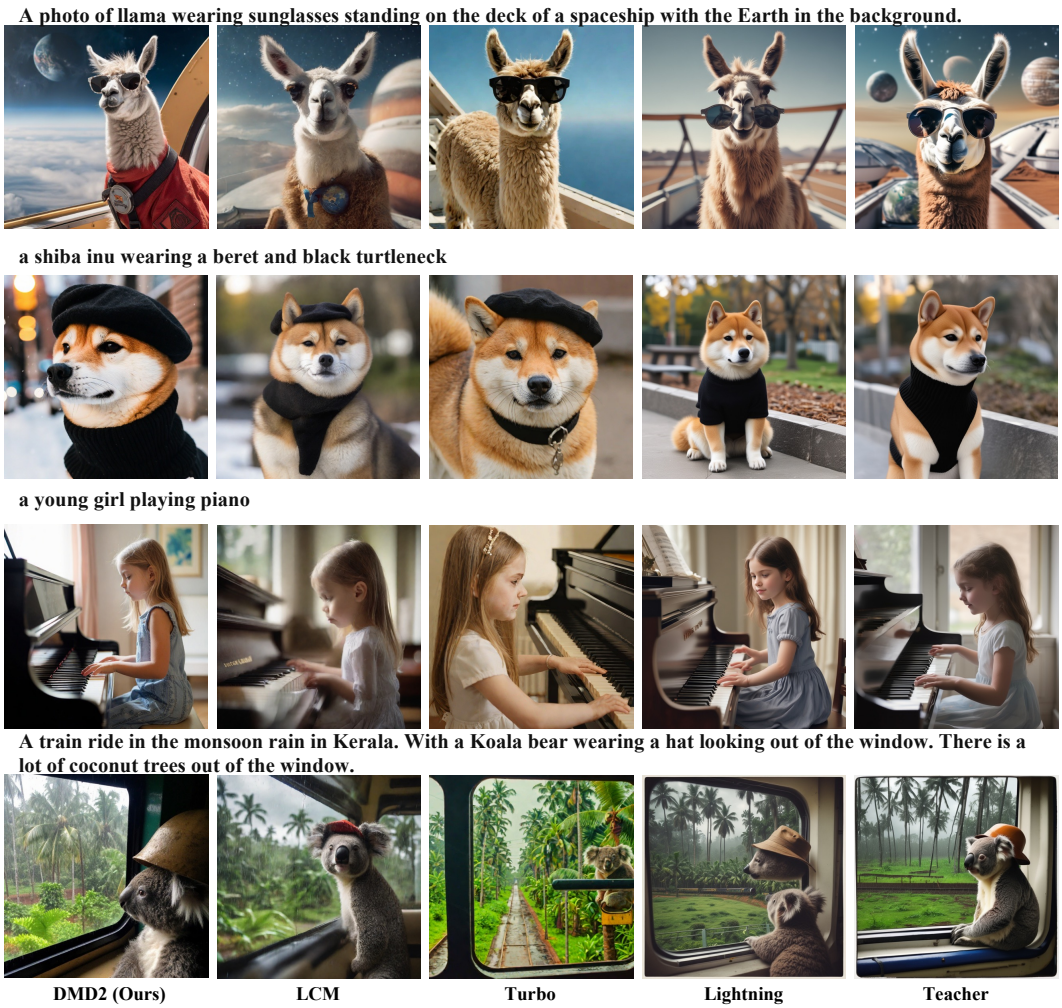

Figure 6: Visual comparison between our model, the SDXL teacher, and selected competing methods [23, 27, 31]. All distilled models use 4 sampling steps while the teacher model uses 50 sampling steps with classifier-free guidance. All images are generated using identical noise and text prompts. Our model produces images with superior realism and text alignment. (Zoom in for details.) More comparisons are available in Appendix Figure 11.

In Table 4, we ablate the influence of the GAN term (Sec. 4.3), distribution matching objective (Eq. 2), and backward simulation (Sec. 4.4) for distilling the SDXL model into a four-step generator. Qualitative results are shown in Appendix Figure. 8. In the absence of the GAN loss, our baseline model produces oversaturated and oversmoothed images (Appendix Fig. 8 third column). Similarly, eliminating distribution matching objective (Eq. 2) reduces our approach to a pure GAN-based method, which struggles with training stability [71, 72]. Moreover, pure GAN-based methods also lack a natural way to incorporate classifier-free guidance [73], essential for high-quality text-to-image synthesis [1, 2]. Consequently, while GAN-based methods achieve the lowest FID by closely matching the real distribution, they significantly underperform in text alignment and aesthetic quality (Appendix Fig. 8 second column). Likewise, omitting the backward simulation leads to worse image quality, as indicated by the degraded patch FID score.

6 Acknowledgements

We extend our gratitude to Minguk Kang and Seungwook Kim for their assistance in setting up the human evaluation. We also thank Zeqiang Lai for suggesting the timestep shift technique used in our one-step generator. Additionally, we are grateful to our friends and colleagues for their insightful discussions and valuable comments. This work was supported by the National Science Foundation under Cooperative Agreement PHY-2019786 (The NSF AI Institute for Artificial Intelligence and Fundamental Interactions, http://iaifi.org/), by NSF Grant 2105819, by NSF CISE award 1955864, and by funding from Google, GIST, Amazon, and Quanta Computer.

References

[1] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

[2] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022.

[3] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[4] 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 *NeurIPS*, 2022.

[5] Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, et al. ediffi: Text-to-image diffusion models with an ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*, 2022.

[6] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

[7] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *CVPR*, 2022.

[8] Shaoan Xie, Zhifei Zhang, Zhe Lin, Tobias Hinz, and Kun Zhang. Smartbrush: Text and shape guided object inpainting with diffusion model. In *CVPR*, 2023.

[9] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *ICML*, 2023.

[10] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *ICLR*, 2022.

[11] Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, and Qiang Liu. Instaflow: One step is enough for high-quality diffusion-based text-to-image generation. *arXiv preprint arXiv:2309.06380*, 2023.

[12] Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. In *ICLR*, 2024.

[13] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *CVPR*, 2023.

[14] Jonathan Heek, Emiel Hoogeboom, and Tim Salimans. Multistep consistency models. *arXiv preprint arXiv:2403.06807*, 2024.

[15] Hanshu Yan, Xingchao Liu, Jiachun Pan, Jun Hao Liew, Qiang Liu, and Jiashi Feng. Perflow: Piecewise rectified flow as universal plug-and-play accelerator. *arXiv preprint arXiv:2405.07510*, 2024.

[16] Eric Luhman and Troy Luhman. Knowledge distillation in iterative generative models for improved sampling speed. *arXiv preprint arXiv:2101.02388*, 2021.

[17] Yuxi Ren, Xin Xia, Yanzuo Lu, Jiacheng Zhang, Jie Wu, Pan Xie, Xing Wang, and Xuefeng Xiao. Hyper-sd: Trajectory segmented consistency model for efficient image synthesis. *arXiv preprint arXiv:2404.13686*, 2024.

[18] Chen Xu, Tianhui Song, Weixin Feng, Xubin Li, Tiezheng Ge, Bo Zheng, and Limin Wang. Accelerating image generation with sub-path linear approximation model. *arXiv preprint arXiv:2404.13903*, 2024.

[19] Jianbin Zheng, Minghui Hu, Zhongyi Fan, Chaoyue Wang, Changxing Ding, Dacheng Tao, and Tat-Jen Cham. Trajectory consistency distillation. *arXiv preprint arXiv:2402.19159*, 2024.

[20] Jiatao Gu, Shuangfei Zhai, Yizhe Zhang, Lingjie Liu, and Joshua M Susskind. Boot: Data-free distillation of denoising diffusion models with bootstrapping. In *ICML 2023 Workshop on Structured Probabilistic Inference & Generative Modeling*, 2023.

 https://doi.org/10.52202/079017-1505

[21] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In *NIPS*, 2014.

[22] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Frédo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *CVPR*, 2024.

[23] Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. *arXiv preprint arXiv:2311.17042*, 2023.

[24] Axel Sauer, Frederic Boesel, Tim Dockhorn, Andreas Blattmann, Patrick Esser, and Robin Rombach. Fast high-resolution image synthesis with latent adversarial diffusion distillation. *arXiv preprint arXiv:2403.12015*, 2024.

[25] Yanwu Xu, Yang Zhao, Zhisheng Xiao, and Tingbo Hou. Ufogen: You forward once large scale text-to-image generation via diffusion gans. In *CVPR*, 2024.

[26] Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. In *ICLR*, 2024.

[27] Shanchuan Lin, Anran Wang, and Xiao Yang. Sdxl-lightning: Progressive adversarial diffusion distillation. *arXiv*, 2024.

[28] Zhendong Wang, Huangjie Zheng, Pengcheng He, Weizhu Chen, and Mingyuan Zhou. Diffusion-gan: Training gans with diffusion. In *ICLR*, 2023.

[29] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion gans. In *ICLR*, 2022.

[30] Mingyuan Zhou, Huangjie Zheng, Zhendong Wang, Mingzhang Yin, and Hai Huang. Score identity distillation: Exponentially fast distillation of pretrained diffusion models for one-step generation. In *ICML*, 2024.

[31] Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023.

[32] Simian Luo, Yiqin Tan, Suraj Patil, Daniel Gu, Patrick von Platen, Apolinário Passos, Longbo Huang, Jian Li, and Hang Zhao. Lcm-lora: A universal stable-diffusion acceleration module. *arXiv preprint arXiv:2310.04378*, 2023.

[33] David Berthelot, Arnaud Autef, Jierui Lin, Dian Ang Yap, Shuangfei Zhai, Siyuan Hu, Daniel Zheng, Walter Talbot, and Eric Gu. Tract: Denoising diffusion models with transitive closure time-distillation. *arXiv preprint arXiv:2303.04248*, 2023.

[34] Axel Sauer, Kashyap Chitta, Jens Müller, and Andreas Geiger. Projected gans converge faster. In *NeurIPS*, 2021.

[35] Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *SIGGRAPH*, 2022.

[36] Weijian Luo, Tianyang Hu, Shifeng Zhang, Jiacheng Sun, Zhenguo Li, and Zhihua Zhang. Diff-instruct: A universal approach for transferring knowledge from pre-trained diffusion models. In *NeurIPS*, 2023.

[37] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *arXiv preprint arXiv:2305.16213*, 2023.

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

[39] Amir Hertz, Kfir Aberman, and Daniel Cohen-Or. Delta denoising score. In *ICCV*, 2023.

[40] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A Yeh, and Greg Shakhnarovich. Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. In *CVPR*, 2023.

[41] Mingxuan Yi, Zhanxing Zhu, and Song Liu. Monoflow: Rethinking divergence gans via the perspective of wasserstein gradient flows. In *ICML*, 2023.

[42] Siddarth Asokan, Nishanth Shetty, Aadithya Srikanth, and Chandra Sekhar Seelamantula. Gans settle scores! *arXiv preprint arXiv:2306.01654*, 2023.

[43] Romann M Weber. The score-difference flow for implicit generative modeling. In *TMLR*, 2023.

[44] Jean-Yves Franceschi, Mike Gartrell, Ludovic Dos Santos, Thibaut Issenhuth, Emmanuel de Bézenac, Mickaël Chen, and Alain Rakotomamonjy. Unifying gans and score-based diffusion as generative particle models. In *NeurIPS*, 2023.

[45] Thuan Hoang Nguyen and Anh Tran. Swiftbrush: One-step text-to-image diffusion model with variational score distillation. In *CVPR*, 2024.

[46] Mingyuan Zhou, Zhendong Wang, Huangjie Zheng, and Hai Huang. Long and short guidance in score identity distillation for one-step text-to-image generation. *arXiv preprint arXiv:2406.01561*, 2024.

[47] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

[48] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021.

[49] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021.

[50] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. In *ICLR*, 2022.

[51] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. In *arXiv preprint arXiv:2211.01095*, 2022.

[52] 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 *NeurIPS*, 2022.

[53] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *NeurIPS*, 2022.

[54] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[55] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *ICCV*, 2023.

[56] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, 2023.

[57] Shelly Sheynin, Adam Polyak, Uriel Singer, Yuval Kirstain, Amit Zohar, Oron Ashual, Devi Parikh, and Yaniv Taigman. Emu edit: Precise image editing via recognition and generation tasks. *arXiv preprint arXiv:2311.10089*, 2023.

[58] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[59] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. In *NeurIPS*, 2022.

[60] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017.

[61] Jonas Kohler, Albert Pumarola, Edgar Schönfeld, Artsiom Sanakoyeu, Roshan Sumbaly, Peter Vajda, and Ali Thabet. Imagine flash: Accelerating emu diffusion models with backward distillation. *arXiv preprint arXiv:2405.05224*, 2024.

[62] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[63] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[64] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[65] Lucy Chai, Michael Gharbi, Eli Shechtman, Phillip Isola, and Richard Zhang. Any-resolution training for high-resolution image synthesis. In *ECCV*, 2022.

[66] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *ICLR*, 2019.

[67] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *NeurIPS*, 2021.

[68] Allan Jabri, David Fleet, and Ting Chen. Scalable adaptive computation for iterative generation. In *ICML*, 2023.

[69] Hongkai Zheng, Weili Nie, Arash Vahdat, Kamyar Azizzadenesheli, and Anima Anandkumar. Fast sampling of diffusion models via operator learning. In *ICML*, 2023.

[70] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, et al. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2022.

 https://doi.org/10.52202/079017-1505

[71] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *ICML*, 2018.

[72] Minguk Kang, Jun-Yan Zhu, Richard Zhang, Jaesik Park, Eli Shechtman, Sylvain Paris, and Taesung Park. Scaling up gans for text-to-image synthesis. In *CVPR*, 2023.

[73] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *arXiv preprint arXiv:2207.12598*, 2022.

[74] Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization. *arXiv preprint arXiv:2311.12908*, 2023.

[75] Zhendong Wang, Jianmin Bao, Wengang Zhou, Weilun Wang, Hezhen Hu, Hong Chen, and Houqiang Li. Dire for diffusion-generated image detection. In *ICCV*, 2023.

[76] Sheng-Yu Wang, Oliver Wang, Richard Zhang, Andrew Owens, and Alexei A Efros. Cnn-generated images are surprisingly easy to spot... for now. In *CVPR*, 2020.

[77] Xudong Shen, Chao Du, Tianyu Pang, Min Lin, Yongkang Wong, and Mohan Kankanhalli. Finetuning text-to-image diffusion models for fairness. In *ICLR*, 2024.

[78] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *ICML*, 2021.

[79] Oran Gafni, Adam Polyak, Oron Ashual, Shelly Sheynin, Devi Parikh, and Yaniv Taigman. Make-a-scene: Scene-based text-to-image generation with human priors. In *ECCV*, 2022.

[80] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. In *ICML*, 2022.

[81] Yufan Zhou, Ruiyi Zhang, Changyou Chen, Chunyuan Li, Chris Tensmeyer, Tong Yu, Jiuxiang Gu, Jinhui Xu, and Tong Sun. Towards language-free training for text-to-image generation. In *CVPR*, 2022.

[82] Axel Sauer, Tero Karras, Samuli Laine, Andreas Geiger, and Timo Aila. Stylegan-t: Unlocking the power of gans for fast large-scale text-to-image synthesis. *ICML*, 2023.

[83] Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. *arXiv preprint arXiv:2302.04867*, 2023.

[84] Yifan Zhang and Bryan Hooi. Hipa: Enabling one-step text-to-image diffusion models via high-frequency-promoting adaptation. *arXiv preprint arXiv:2311.18158*, 2023.

[85] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *ECCV*, 2018.

[86] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *NeurIPS*, 2017.

[87] 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 *NeurIPS*, 2024.

[88] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

[89] Junsong Chen, Chongjian Ge, Enze Xie, Yue Wu, Lewei Yao, Xiaozhe Ren, Zhongdao Wang, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-\sigma: Weak-to-strong training of diffusion transformer for 4k text-to-image generation. *arXiv preprint arXiv:2403.04692*, 2024.

[90] Zeqiang Lai. Opendmd: Open source implementation and models of one-step diffusion with distribution matching distillation. `https://github.com/Zeqiang-Lai/OpenDMD`, 2024. Accessed: 2024-05-21.

[91] Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. On aliased resizing and surprising subtleties in gan evaluation. In *CVPR*, 2022.

Table of Contents

A Limitations

While achieving superior image quality and text alignment, our distilled generator experiences a slight degradation in image diversity compared to the teacher models (see Appendix F). Additionally, our generator still requires four steps to match the quality of the largest SDXL model. These limitations, while not unique to our model, highlight areas for further improvement. Like most previous distillation methods, we use a fixed guidance scale during training, limiting user flexibility. Introducing a variable guidance scale [13, 31] could be a promising direction for future research. Furthermore, our methods are optimized for distribution matching; incorporating human feedback or other reward functions could further enhance performance [17, 74]. Lastly, training large-scale generative models is computationally intensive, making it inaccessible for most researchers. We hope our efficient approach and optimized, user-friendly codebase will help democratize future research in this field.

B Code, dataset, and more results

We are continually updating the paper and related materials. For access to our code, model, and the latest version of the paper with expanded results and analysis, please visit our project website.

https://doi.org/10.52202/079017-1505

Table 5: Sample quality comparison on 30K prompts from COCO 2014.

Family	Method	Resolution (↑)	Latency (↓)	FID (↓)
Original, unaccelerated	DALL·E [78]	256	-	27.5
	DALL·E 2 [3]	256	-	10.39
	Parti-750M [70]	256	-	10.71
	Parti-3B [70]	256	6.4s	8.10
	Make-A-Scene [79]	256	25.0s	11.84
	GLIDE [80]	256	15.0s	12.24
	LDM [1]	256	3.7s	12.63
	Imagen [4]	256	9.1s	7.27
	eDiff-I [5]	256	32.0s	**6.95**
GANs	LAFITE [81]	256	0.02s	26.94
	StyleGAN-T [82]	512	0.10s	13.90
	GigaGAN [72]	512	0.13s	**9.09**
Accelerated diffusion	DPM++ (4 step) [51]	512	0.26s	22.36
	UniPC (4 step) [83]	512	0.26s	19.57
	LCM-LoRA (4 step) [32]	512	0.19s	23.62
	InstaFlow-0.9B [11]	512	0.09s	13.10
	SwiftBrush [45]	512	0.09s	16.67
	HiPA [84]	512	0.09s	13.91
	UFOGen [25]	512	0.09s	12.78
	SLAM (4 step) [18]	512	0.19s	10.06
	DMD [22]	512	0.09s	11.49
	DMD2 (Ours)	512	0.09s	**8.35**
Teacher	SDv1.5 (50 step, cfg=3, ODE) [1, 50]	512	2.59s	8.59
	SDv1.5 (200 step, cfg=2, SDE) [1, 47]	512	10.25s	7.21

C Broader Impact

Our work on improving the efficiency and quality of diffusion model has several potential societal impacts, both positive and negative. On the positive side, the advancements in fast image synthesis can significantly benefit various creative industries. These models can enhance graphic design, animation, and digital art by providing artists with powerful tools to generate high-quality visuals efficiently. Additionally, improved text-to-image synthesis capabilities can be used in education and entertainment, enabling the creation of personalized learning materials and immersive experiences.

However, potential negative societal impacts must be considered. Misuse risks include generating misinformation and creating fake profiles, which could spread false information and manipulate public opinion. Deploying these technologies could result in biases that unfairly impact specific groups, especially if models are trained on biased datasets, potentially perpetuating or amplifying existing societal biases. To mitigate these risks, we are interested in developing monitoring mechanisms to detect and prevent misuse [75, 76] and methods to enhance output diversity and fairness [77].

D SD v1.5 Results

Table 5 presents detailed comparisons between our one-step generator distilled from SD v1.5 and competing approaches.

E Comparison with DMD1

A comparison between DMD and DMD2 is shown in Algorithm 7.

F Text-to-Image Synthesis Further Analysis

Qualitative ablation results using SDXL backbone are shown in Figure 8. Additionally, we compare the image diversity of our 4-step generator with other competing approaches distilled from SDXL [23,

Table 6: Image quality and diversity comparison with SDXL backbone.

Method	# Fwd Pass (↓)	FID (↓)	Patch FID (↓)	CLIP (↑)	Diversity Score (↑)
LCM-SDXL [32]	4	22.16	33.92	0.317	0.61
SDXL-Turbo [23]	4	23.19	23.27	**0.334**	0.58
SDXL-Lightning [27]	4	24.46	24.56	0.323	**0.63**
DMD2 (Ours)	4	**19.32**	**20.86**	0.332	0.61
SDXL-Teacher, cfg=6 [58]	100	19.36	21.38	0.332	0.64
SDXL-Teacher, cfg=8 [58]	100	20.39	23.21	0.335	0.64

Method	ImageReward	Aesthetic Score
SDXL	0.86	6.16
DMD2	**1.07**	**6.30**

Table 7: Comparison of ImageReward and Aesthetic Score for Different Methods

27, 31]. We employ an LPIPS-based diversity score, similar to that used in multi-modal image-to-image translation [85, 86]. Specifically, we generate four images per prompt and calculate the average pairwise LPIPS distance [54]. For this evaluation, we use the LADD [24] subset of PartiPrompts [70]. We also report the FID and CLIP score measured on 10K prompts from COCO 2014 on the side. Table 6 summarizes the results. Table 7 provides further comparisons using image reward [87] and aesthetic score metrics [59]. Our model achieves the best image quality, indicated by the lowest FID and Patch FID scores. We also achieve text alignment comparable to SDXL-Turbo while attaining a better diversity score. While SDXL-Lightning [27] exhibits a higher diversity score than our approach, it suffers from considerably worse text alignment, as reflected by the lower CLIP score and human evaluation (Fig. 5). This suggests that the improved diversity is partially due to random outputs lacking prompt coherence. We note that it is possible to increase the diversity of our model by raising the weights for the GAN objective, which aligns with the more diverse unguided distribution. Further investigation into finding the optimal balance between distribution matching and the GAN objective is left for future work.

G Two Time-scale Update Rule Further Analysis

In Section 4.2, we discuss that updating the fake score multiple times (5 updates) per generator update leads to better stability. Here, we provide further analysis. Figure 9 visualizes pixel brightness variations throughout training. The baseline approach, which omits the regression objective from DMD and uses just 1 fake score update, results in significant training instability, as evidenced by periodic fluctuations in pixel brightness. In contrast, our two time-scale update rule with 5 fake score updates per generator update stabilizes the training and leads to better sample quality, as shown in Tab. 3.

We further examine the influence of the update frequency for the fake diffusion model μ_{fake} in Figure 10. An update frequency of 1 fake diffusion update per generator update corresponds to the naive baseline (red line) and suffers from training instability. Although a frequency of 10 updates (magenta line) provides excellent stability, it significantly slows down the training process. We found that a moderate frequency of 5 updates (green line) achieves the best balance between stability and convergence speed on ImageNet. Our approach proves more effective than using asynchronous learning rates [60] (cyan line) and converges significantly faster than the original DMD method that employs a regression loss [22] (dark blue line). For new models and datasets, we recommend adjusting the iteration number to the smallest value that ensures the stability of general image statistics, such as pixel brightness.

H Additional Text-to-Image Synthesis Results

Additional visual comparisons for the 4-step distilled models are shown in Figure 11. Sample outputs from our one-step generator are presented in Figure 12.

https://doi.org/10.52202/079017-1505

Algorithm 1: DMD (original)

Input: Pretrained real diffusion model μ_{real}, paired ODE solution pairs $\mathcal{D} = \{z_{\text{ref}}, y_{\text{ref}}\}$
Output: Trained generator G

// Initialize generator & fake score model
$G \leftarrow \text{copyWeights}(\mu_{\text{real}})$
$\mu_{\text{fake}} \leftarrow \text{copyWeights}(\mu_{\text{real}})$
while *train* **do**
  // Generate batch of images
  Sample $z \sim \mathcal{N}(0, \mathbf{I})^B$; $(z_{\text{ref}}, y_{\text{ref}}) \sim \mathcal{D}$;
  $x \leftarrow G(z)$; $x_{\text{ref}} \leftarrow G(z_{\text{ref}})$;
  // 1. Update generator
  $\mathcal{L}_{\text{KL}} \leftarrow \text{distributionMatchingLoss}\big(\mu_{\text{real}}, \mu_{\text{fake}}, x\big)$;
  $\mathcal{L}_{\text{reg}} \leftarrow \text{LPIPS}(x_{\text{ref}}, y_{\text{ref}})$;
  $\mathcal{L}_G \leftarrow \mathcal{L}_{\text{KL}} + \lambda_{\text{reg}}\, \mathcal{L}_{\text{reg}}$;
  $G \leftarrow \text{update}(G, \mathcal{L}_G)$;
  // 2. Update fake score model
  Sample $t \sim \mathcal{U}(0, 1)$;
  $x_t \leftarrow \text{forwardDiffusion}(\text{stopgrad}(x), t)$;
  $\mathcal{L}_{\text{denoise}} \leftarrow \text{denoisingLoss}\big(\mu_{\text{fake}}(x_t, t), \text{stopgrad}(x)\big)$;
  $\mu_{\text{fake}} \leftarrow \text{update}\big(\mu_{\text{fake}}, \mathcal{L}_{\text{denoise}}\big)$;
end while

Algorithm 2: DMD2 (ours)

Input: Pretrained real diffusion model μ_{real}, real image dataset $\mathcal{D}_{\text{real}}$,
update_G_freq: frequency of generator updates
Output: Trained generator G

// Initialize generator, fake score model, discriminator
$G \leftarrow \text{copyWeights}(\mu_{\text{real}})$
$\mu_{\text{fake}} \leftarrow \text{copyWeights}(\mu_{\text{real}})$
$D \leftarrow \text{initializeDiscriminator}()$
for *iteration = 1 to maxIters* **do** // main training loop
  // (a) Generate batch of images
  Sample $z \sim \mathcal{N}(0, \mathbf{I})^B$;
  if *multi-step* **then**
    $x \leftarrow G(\text{multiStepSampling}(G, z))$;
    // simulate multi-step inference
  end if
  else
    $x \leftarrow G(z)$
  end if
  // (b) Update G only once per update_G_freq iterations
  if *iteration* % **update_G_freq == 0 then**
    // (b1) Distribution matching loss
    $\mathcal{L}_{\text{KL}} \leftarrow \text{distributionMatchingLoss}\big(\mu_{\text{real}}, \mu_{\text{fake}}, x\big)$;
    // (b2) GAN loss term
    $\mathcal{L}_{\text{GAN}} \leftarrow -\mathbb{E}\big[\log\, D(F(G(z), t))\big]$;
    // (b3) Final generator loss
    $\mathcal{L}_G \leftarrow \mathcal{L}_{\text{KL}} + \lambda_{\text{GAN}}\, \mathcal{L}_{\text{GAN}}$;
    // (b4) Apply gradient update to G
    $G \leftarrow \text{update}(G, \mathcal{L}_G)$;
  end if
  Sample real images $x_{\text{real}} \sim \mathcal{D}_{\text{real}}$;
  Sample $t \sim \mathcal{U}(0, 1)$;
  // (c) Update μ_{fake} (fake score) via denoising on fake data
  $x_t \leftarrow \text{forwardDiffusion}(\text{stopgrad}(x), t)$;
  $\mathcal{L}_{\text{denoise}} \leftarrow \text{denoisingLoss}\big(\mu_{\text{fake}}(x_t, t), \text{stopgrad}(x)\big)$;
  $\mu_{\text{fake}} \leftarrow \text{update}\big(\mu_{\text{fake}}, \mathcal{L}_{\text{denoise}}\big)$;
  // (d) Update discriminator D (GAN classification)
  $\mathcal{L}_{\text{GAN-D}} \leftarrow \mathbb{E}\big[\log D(F(x_{\text{real}}, t))\big] + \mathbb{E}\big[\log(1 - D(F(\text{stopgrad}(x), t)))\big]$;
  $D \leftarrow \text{update}(D, \mathcal{L}_{\text{GAN-D}})$;
end for

Figure 7: Side-by-side comparison of **DMD** (left) and our improved **DMD2** (right).

I ImageNet Visual Results

In Figure 13, we present qualitative results obtained from our one-step distilled model trained on the ImageNet dataset.

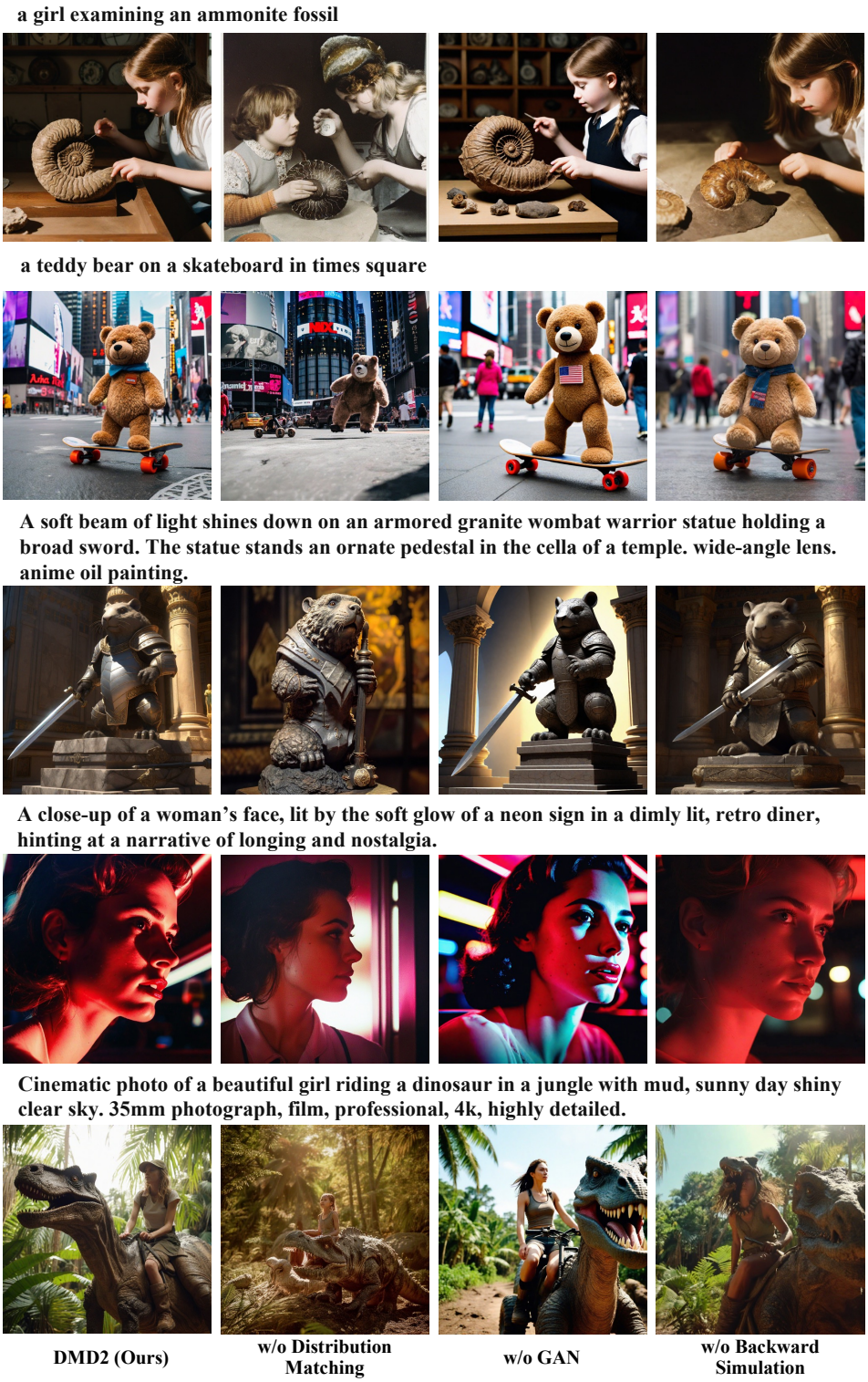

Figure 8: SDXL Qualitative Ablations. All images are generated using identical noise and text prompts. Removing the distribution matching objective significantly degrades aesthetic quality and text alignment. Omitting the GAN loss results in oversaturated and overly smoothed images. The baseline without backward simulation produces images of lower quality.

https://doi.org/10.52202/079017-1505

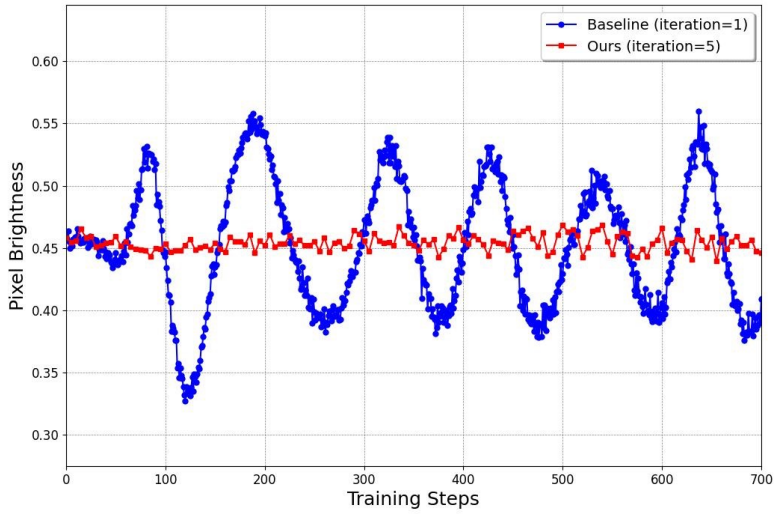

Figure 9: Visualization of pixel brightness variations throughout training. The baseline approach, which naively removes the regression loss from the original DMD [22], suffers from significant training instability, leading to fluctuating general image statistics like the overall pixel brightness. In contrast, our two time-scale update rule, which optimizes the fake diffusion model five times per generator update, significantly stabilizes training and enhances sample quality.

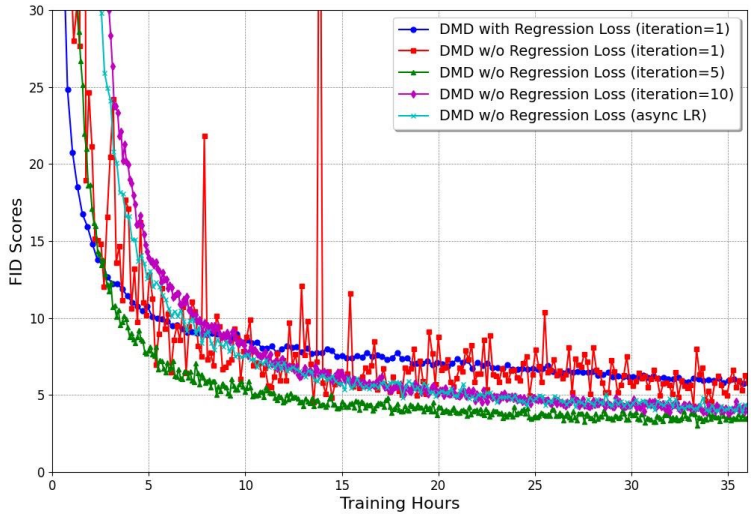

Figure 10: Visualization of FID score progression during training. Naively removing the regression loss leads to training instability (red line). A two time-scale update rule with five fake diffusion critic updates per generator update stabilizes training and is more effective than using a larger number of fake diffusion updates or an asynchronous learning rate where the fake diffusion model uses a learning rate 5 times larger than the generator. The model trained with our two time-scale update rule (green) also converges significantly faster than the original DMD method with a regression loss (dark blue), even though TTUR performs less number of the generator weight updates.

Figure 11: Additional visual comparison between our model, the SDXL teacher, and selected competing methods [23, 27, 31]. All distilled models use 4 sampling steps while the teacher model uses 50 sampling steps with classifier-free guidance. All images are generated using identical noise and text prompts. Our model produces images with superior realism and text alignment. Please zoom in for details.

https://doi.org/10.52202/079017-1505

Figure 12: Additional 1024×1024 samples produced by our 1-step generator distilled from SDXL. Please zoom in for details.

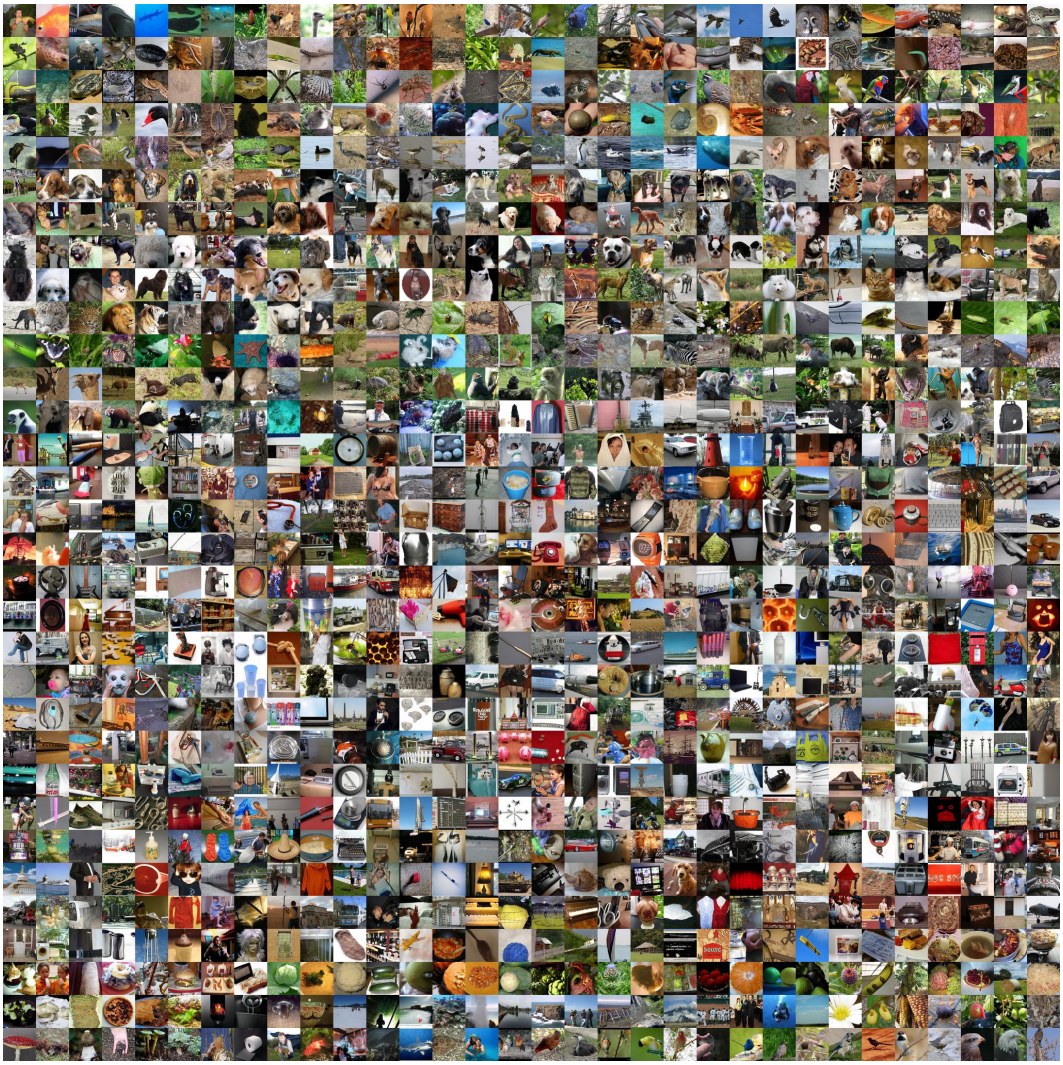

Figure 13: One-step samples from our generator trained on ImageNet (FID=1.28). Please zoom in for details.

https://doi.org/10.52202/079017-1505

J Implementation Details

This section outlines key aspects of the implementation. All results are reproducible using our open-source training and evaluation code. Generally, we employ the maximum batch size supported by our compute resources. The learning rate is set to the highest stable value, ensuring no divergent loss within the initial 500 iterations. We determine the TTUR iteration count based on the minimum needed for stability, while the guidance scale is selected to optimize performance for the teacher model.

J.1 GAN Classifier Design

Our GAN classifier design is inspired by SDXL-Lightning [27]. Specifically, we attach a prediction head to the middle block output of the fake diffusion model. The prediction head consists of a stack of 4×4 convolutions with a stride of 2, group normalization, and SiLU activations. All feature maps are downsampled to 4×4 resolution, followed by a single convolutional layer with a kernel size and stride of 4. This layer pools the feature maps into a single vector, which is then passed to a linear projection layer to predict the classification result.

J.2 ImageNet

Our ImageNet implementation closely follows the DMD paper [22]. Specifically, we distill a one-step generator from the EDM pretrained model [53], released under the CC BY-NC-SA 4.0 License. For the standard training setup, we use the AdamW optimizer [88] with a learning rate of 2×10^{-6}, a weight decay of 0.01, and beta parameters (0.9, 0.999). We use a batch size of 280 and train the model on 7 A100 GPUs for 200K iterations, which takes approximately 2 days. The number of fake diffusion model update per generator update is set to 5. The weight for the GAN loss is set to 3×10^{-3}. For the extended training setup shown in Table 1, we first pretrain the model without GAN loss for 400K iterations. We then resume from the best checkpoint (as measured by FID), enable the GAN loss with a weight of 3×10^{-3}, reduce the learning rate to 5×10^{-7}, and continue training for an additional 150K iterations. The total training time for this run is approximately 5 days.

J.3 SD v1.5

We distill a one-step generator from the SD v1.5 model [1], released under the CreativeML Open RAIL-M license, using prompts from the LAION-Aesthetic 6.25+ dataset [59]. Additionally, we collect 500K images from LAION-Aesthetic 5.5+ as training data for the GAN discriminator, filtering out images smaller than 1024×1024 and those containing unsafe content. Our training process involves two stages. In the first stage, we disable the GAN loss and use the AdamW optimizer with a learning rate of 1×10^{-5}, a weight decay of 0.01, and beta parameters of (0.9, 0.999). The fake diffusion model is updated 10 times per generator update. We set the guidance scale for the real diffusion model to be 1.75. We use a batch size of 2048 and train the model on 64 A100 GPUs for 40K iterations. In the second stage, we enable the GAN loss with a weight of 10^{-3}, reduce the learning rate to 5×10^{-7}, and continue training for an additional 5K iterations. The total training time is approximately 26 hours.

J.4 SDXL

We train both one-step and four-step generators by distilling from the SDXL model [58], released under the CreativeML Open RAIL++-M License. For the one-step generator, we observed similar block noise artifacts as reported in SDXL-Lightning [27] and Pixart-Sigma [89]. We addressed this by adopting the timestep shift technique from OpenDMD [90] and Pixart-Sigma [89], setting the conditioning timestep to 399. Additionally, we initialized the one-step generator by pretraining it with a regression loss using a small set of 10K pairs for a short period. These adjustments are not necessary for the multi-step model or other backbones, suggesting this issue might be specific to SDXL. Similar to SD v1.5, we use prompts from the LAION-Aesthetic 6.25+ dataset [59] and collect 500K images from LAION-Aesthetic 5.5+ as training data for the GAN discriminator, filtering out images smaller than 1024×1024 and those containing unsafe content. The generator is trained using the AdamW optimizer with a learning rate of 5×10^{-7}, a weight decay of 0.01, and beta parameters of (0.9, 0.999). The fake diffusion model is updated 5 times per generator update. We set the guidance

scale for the real diffusion model to be 8. We use a batch size of 128 and train the model on 64 A100 GPUs for 20K iterations for the 4-step generator and 25K iterations for the 1-step generator, taking approximately 60 hours.

K Evaluation Details

For the COCO experiments, we follow the exact evaluation setup as GigaGAN [72] and DMD [22]. For the results presented in Table 5, we use 30K prompts from the COCO 2014 validation set and generate the corresponding images. The outputs are downsampled to 256×256 and compared with 40,504 real images from the same validation set using clean-FID [91]. For the results presented in Table 2, we use a random set of 10K prompts from the COCO 2014 validation set and generate the corresponding images. The outputs are downsampled to 512×512 and compared with the corresponding 10K real images from the validation set with the same prompts. We compute the CLIP score using the OpenCLIP-G backbone. For the ImageNet results, we generate 50,000 images and calculate the FID statistics using EDM's evaluation code [53].

L User Study Details

To conduct the human preference study, we use the Prolific platform (https://www.prolific.com). We use 128 prompts from the LADD [24] subset of PartiPrompts [70]. All approaches generate corresponding images, which are presented in pairs to human evaluators to measure aesthetic and prompt alignment preference. The specific questions and interface are shown in Figure 14. Consent is obtained from the voluntary participants. We manually verify that all generated images contain standard visual content that poses no risks to the study participants.

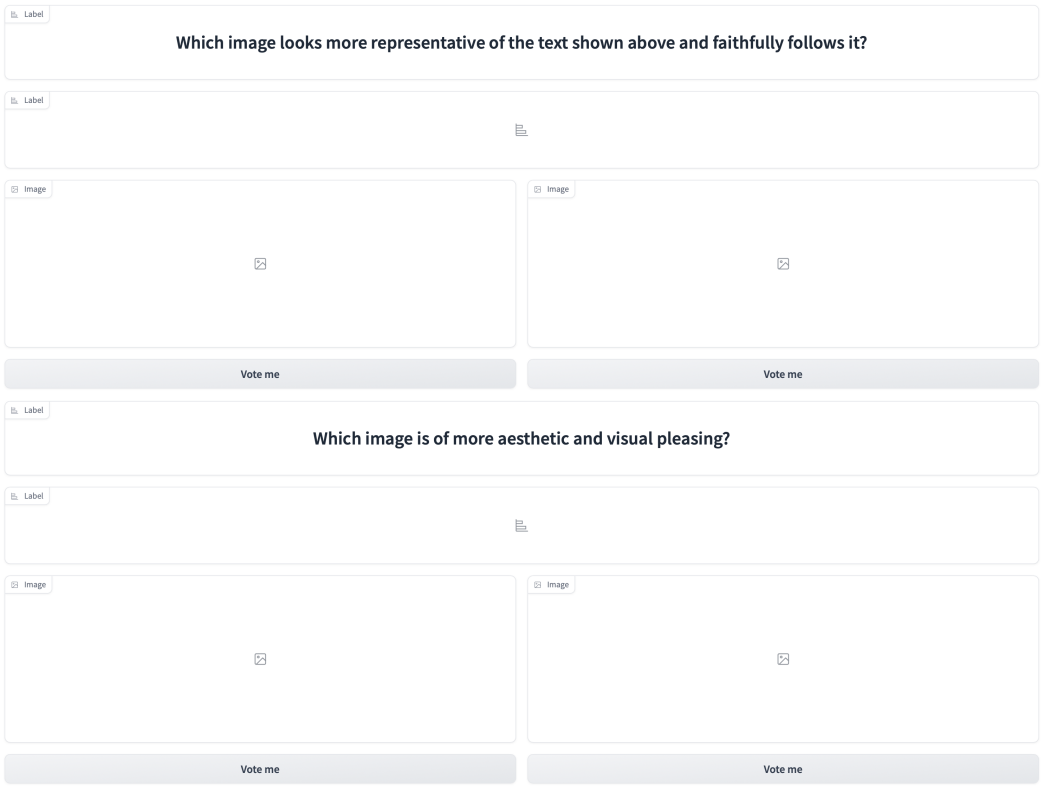

Figure 14: A sample interface for our user preference study, where images are presented in a random left/right order.

https://doi.org/10.52202/079017-1505

M Prompts for Figure 1, Figure 2, and Figure 12

We use the following prompts for Figure 1. From left to right, top to bottom:

- a girl examining an ammonite fossil
- A photo of an astronaut riding a horse in the forest.
- a giant gorilla at the top of the Empire State Building
- A close-up photo of a wombat wearing a red backpack and raising both arms in the air. Mount Rushmore is in the background.
- An oil painting of two rabbits in the style of American Gothic, wearing the same clothes as in the original.
- a portrait of an old man
- a watermelon chair
- A sloth in a go kart on a race track. The sloth is holding a banana in one hand. There is a banana peel on the track in the background.
- a penguin standing on a sidewalk
- a teddy bear on a skateboard in times square

We use the following prompts for Figure 2. From left to right, top to bottom:

- a chimpanzee sitting on a wooden bench
- a cat reading a newspaper
- A television made of water that displays an image of a cityscape at night.
- a portrait of a statue of the Egyptian god Anubis wearing aviator goggles, white t-shirt and leather jacket. The city of Los Angeles is in the background.
- a squirrell driving a toy car
- an elephant walking on the Great Wall
- a capybara made of voxels sitting in a field
- Cinematic photo of a beautiful girl riding a dinosaur in a jungle with mud, sunny day shiny clear sky. 35mm photograph, film, professional, 4k, highly detailed.
- A still image of a humanoid cat posing with a hat and jacket in a bar.
- A soft beam of light shines down on an armored granite wombat warrior statue holding a broad sword. The statue stands an ornate pedestal in the cella of a temple. wide-angle lens. anime oil painting.
- children
- A photograph of the inside of a subway train. There are red pandas sitting on the seats. One of them is reading a newspaper. The window shows the jungle in the background.
- a goat wearing headphones
- motion
- A close-up of a woman's face, lit by the soft glow of a neon sign in a dimly lit, retro diner, hinting at a narrative of longing and nostalgia.

We use the following prompts for Figure 12. From left to right, top to bottom:

- A close-up of a woman's face, lit by the soft glow of a neon sign in a dimly lit, retro diner, hinting at a narrative of longing and nostalgia.
- a cat reading a newspaper
- A television made of water that displays an image of a cityscape at night.
- a portrait of a statue of the Egyptian god Anubis wearing aviator goggles, white t-shirt and leather jacket. The city of Los Angeles is in the background.
- a squirrell driving a toy car
- an elephant walking on the Great Wall
- a capybara made of voxels sitting in a field
- A soft beam of light shines down on an armored granite wombat warrior statue holding a broad sword. The statue stands an ornate pedestal in the cella of a temple. wide-angle lens. anime oil painting.
- a goat wearing headphones
- An oil painting of two rabbits in the style of American Gothic, wearing the same clothes as in the original.
- a girl examining an ammonite fossil
- a chimpanzee sitting on a wooden bench

- children
- A still image of a humanoid cat posing with a hat and jacket in a bar.
- motion

https://doi.org/10.52202/079017-1505

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We accurately state our contribution and scope in abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Section A.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: Our paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide all experimental details in the main paper (Section 5) and the Appendix (Sections J, and K). Additionally, we release all our code and models to reproduce the results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

https://doi.org/10.52202/079017-1505

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We release them at our project website.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: In the main paper (Section 5) and Appendix (Section J, and K)

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Reporting error bars is computationally prohibitive due to the high cost of large-scale generative model training. However, to the best of our knowledge, the variance across runs and seeds is minimal.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: In Appendix J.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We fully comply with the NeurIPS code of ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: In Appendix C.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

https://doi.org/10.52202/079017-1505

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper does not pose significant risks, and future production systems could implement an output classifier to detect and prevent unsafe content.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: In Appendix J.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release code and dataset on our website.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: In Appendix L.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our human study only requires participants to compare image outputs from different models and poses no risks to them. More details are available in Appendix L.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

https://doi.org/10.52202/079017-1505