One-step Diffusion with Distribution Matching Distillation

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

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

https://tianweiy.github.io/dmd/

(a) DSLR photo of a golden retriever in heavy snow. (b) Lightshow at the Dolomites. (c) [...] stylishly dressed elderly woman [...] large glasses [...]. (d) [...] warrior chief, tribal panther make up, blue on red [...]. (e) hyperrealistic photo of a fox astronaut; perfect face, artstation.

Figure 1. **Which is which?** Among these images, some were generated with baseline **Stable Diffusion (SD) [63] (2590ms each)**, the others with our **Diffusion Matching Distillation (DMD) (90ms each)**. Can you tell which is which? Answers in the footnote[1]. (Non-abbreviated prompts in Appendix G.) Our one-step text-to-image generators provide quality rivaling expensive diffusion models.

Abstract

Diffusion models generate high-quality images but require dozens of forward passes. We introduce Distribution Matching Distillation (DMD), a procedure to transform a diffusion model into a one-step image generator with minimal impact on image quality. We enforce the one-step image generator match the diffusion model at distribution level, by minimizing an approximate KL divergence whose gradient can be expressed as the difference between 2 score functions, one of the target distribution and the other of the synthetic distribution being produced by our one-step generator. The score functions are parameterized as two diffusion models trained separately on each distribution. Combined with a simple regression loss matching the large-scale structure of the multi-step diffusion outputs, our method outperforms all published few-step diffusion approaches, reaching 2.62 FID on ImageNet 64×64 and 11.49 FID on zero-shot COCO-30k, comparable to Stable Diffusion but orders of magnitude faster. Utilizing FP16 inference, our model can generate images at 20 FPS on modern hardware.

1. Introduction

Diffusion models [21, 61, 63, 64, 71, 74] have revolutionized image generation, achieving unprecedented levels of realism and diversity with a stable training procedure. In contrast to GANs [15] and VAEs [34], however, their sampling is a slow, iterative process that transforms a Gaussian noise sample into an intricate image by progressive denoising [21, 74]. This typically requires tens to hundreds of costly neural network evaluations, limiting interactivity in using the generation pipeline as a creative tool.

To accelerate sampling speed, previous methods [42, 43, 47, 48, 51, 65, 75, 90, 91] distill the noise→image mapping, discovered by the original multi-step diffusion sampling, into a single-pass student network. However, fitting such a high-dimensional, complex mapping is certainly a demanding task. A challenge is the expensive cost of run-

[1]**Ours (left to right):** bottom, top, bottom, bottom, top.

ning the full denoising trajectory, just to realize one loss computation of the student model. Recent methods mitigate this by progressively increasing the sampling distance of the student, without running the full denoising sequence of the original diffusion [3, 16, 42, 43, 51, 65, 75]. However, the performance of distilled models still lags behind the original multi-step diffusion model.

In contrast, rather than enforcing correspondences between noise and diffusion-generated images, we simply enforce that the student generations look indistinguishable from the original diffusion model. At high level, our goal shares motivation with other *distribution-matching* generative models, such as GMMN [39] or GANs [15]. Still, despite their impressive success in creating realistic images [27, 30], scaling up the model on the general text-to-image data has been challenging [26, 62, 87]. In this work, we bypass the issue by starting with a diffusion model that is already trained on large-scale text-to-image data. Concretely, we finetune the pretrained diffusion model to learn not only the data distribution, but also the *fake* distribution that is being produced by our distilled generator. Since diffusion models are known to approximate the score functions on diffused distributions [23, 73], we can interpret the denoised diffusion outputs as gradient directions for making an image "more realistic", or if the diffusion model is learned on the fake images, "more fake". Finally, the gradient update rule for the generator is concocted as the difference of the two, nudging the synthetic images toward higher realism and lower fakeness. Previous work [80], in a method called Variational Score Distillation, shows that modeling the real and fake distributions with a pretrained diffusion model is also effective for test-time optimization of 3D objects. Our insight is that a similar approach can instead train *an entire generative model*.

Furthermore, we find that pre-computing a modest number of the multi-step diffusion sampling outcomes and enforcing a simple regression loss with respect to our one-step generation serves as an effective regularizer in the presence of the distribution matching loss. Moreover, the regression loss ensures our one-step generator aligns with the teacher model (see Figure 6), demonstrating potential for real-time design previews. Our method draws upon inspiration and insights from VSD [80], GANs [15], and pix2pix [24], showing that by (1) modeling real and fake distributions with diffusion models and (2) using a simple regression loss to match the multi-step diffusion outputs, we can train a one-step generative model with high fidelity.

We evaluate models trained with our Distribution Matching Distillation procedure (DMD) across various tasks, including image generation on CIFAR-10 [36] and ImageNet 64×64 [8], and zero-shot text-to-image generation on MS COCO 512×512 [40]. On all benchmarks, our one-step generator significantly outperforms all published few-steps diffusion methods, such as Progressive Distillation [51, 65], Rectified Flow [42, 43], and Consistency Models [48, 75]. On ImageNet, DMD reaches FIDs of 2.62, an improvement of 2.4× over Consistency Model [75]. Employing the identical denoiser architecture as Stable Diffusion [63], DMD achieves a competitive FID of 11.49 on MS-COCO 2014-30k. Our quantitative and qualitative evaluations show that the images generated by our model closely resemble the quality of those generated by the costly Stable Diffusion model. Importantly, our approach maintains this level of image fidelity while achieving a 100× reduction in neural network evaluations. This efficiency allows DMD to generate 512 × 512 images at a rate of 20 FPS when utilizing FP16 inference, opening up a wide range of possibilities for interactive applications.

2. Related Work

Diffusion Model Diffusion models [2, 21, 71, 74] have emerged as a powerful generative modeling framework, achieving unparalleled success in diverse domains such as image generation [61, 63, 64], audio synthesis [6, 35], and video generation [11, 22, 70]. These models operate by progressively transforming noise into coherent structures through a reverse diffusion process [72, 74]. Despite state-of-the-art results, the inherently iterative procedure of diffusion models entails a high and often prohibitive computational cost for real-time applications. Our work builds upon leading diffusion models [31, 63] and introduces a simple distillation pipeline that reduces the multi-step generative process to a single forward pass. Our method is universally applicable to any diffusion model with deterministic sampling [31, 72, 74].

Diffusion Acceleration Accelerating the inference process of diffusion models has been a key focus in the field, leading to the development of two types of approaches. The first type advances fast diffusion samplers [31, 41, 45, 46, 90], which can dramatically reduce the number of sampling steps required by pre-trained diffusion models—from a thousand down to merely 20-50. However, a further reduction in steps often results in a catastrophic decrease in performance. Alternatively, diffusion distillation has emerged as a promising avenue for further boosting speed [3, 16, 42, 47, 51, 65, 75, 82, 91]. They frame diffusion distillation as knowledge distillation [19], where a student model is trained to distill the multi-step outputs of the original diffusion model into a single step. Luhman *et al*. [47] and DSNO [92] proposed a simple approach of pre-computing the denoising trajectories and training the student model with a regression loss in pixel space. However, a significant challenge is the expensive cost of running the full denoising trajectory for each realization of the loss function. To address this issue, Progressive

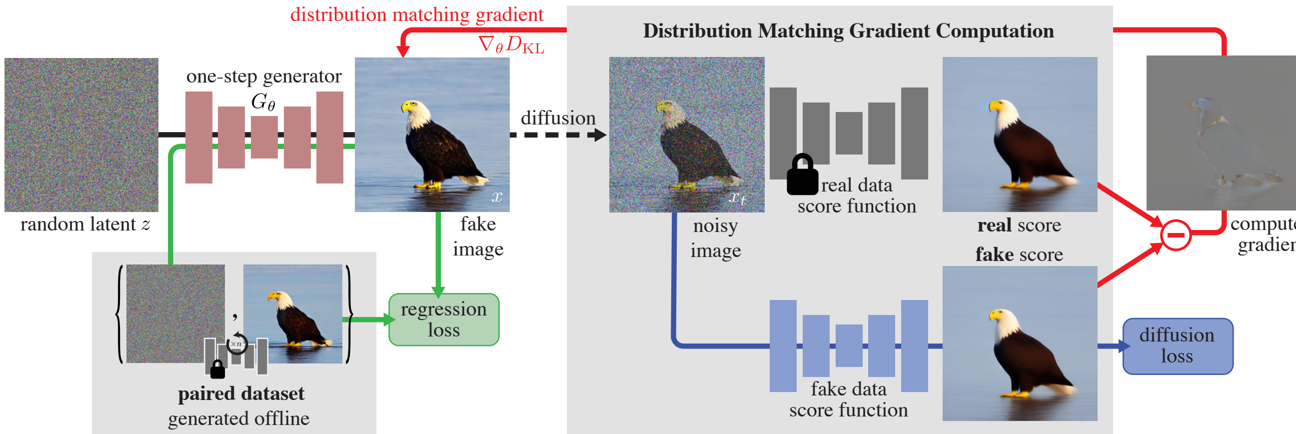


Figure 2. **Method overview.** We train one-step generator G_θ to map random noise z into a realistic image. To match the multi-step sampling outputs of the diffusion model, we pre-compute a collection of noise–image pairs, and occasionally load the noise from the collection and enforce LPIPS [88] regression loss between our one-step generator and the diffusion output. Furthermore, we provide **distribution matching gradient** $\nabla_\theta D_{KL}$ to the fake image to enhance realism. We inject a random amount of noise to the fake image and pass it to two diffusion models, one pretrained on the real data and the other continually trained on the fake images with a diffusion loss, to obtain its denoised versions. The denoising scores (visualized as mean prediction in the plot) indicate directions to make the images more realistic or fake. The difference between the two represents the direction toward more realism and less fakeness and is backpropagated to the one-step generator.

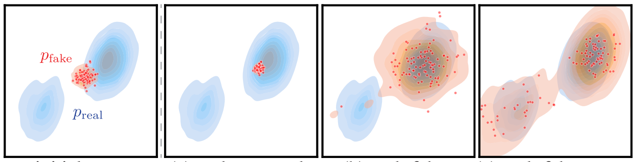
Distillation (PD) [51, 65] train a series of student models that halve the number of sampling steps of the previous model. InstaFlow [42, 43] progressively learn straighter flows on which the one step prediction maintains accuracy over a larger distance. Consistency Distillation (CD) [75], TRACT [3], and BOOT [16] train a student model to match its own output at a different timestep on the ODE flow, which in turn is enforced to match its own output at yet another timestep. In contrast, our method shows that the simple approach of Luhman *et al*. and DSNO to pre-compute the diffusion outputs is sufficient, once we introduce distribution matching as the training objective.

Distribution Matching Recently, a few classes of generative models have shown success in scaling up to complex datasets by recovering samples that are corrupted by a predefined mechanism, such as noise injection [21, 61, 64] or token masking [5, 60, 86]. On the other hand, there exist generative methods that do not rely on sample reconstruction as the training objective. Instead, they match the synthetic and target samples at a distribution level, such as GMMD [10, 39] or GANs [15]. Among them, GANs have shown unprecedented quality in realism [4, 26–28, 30, 67], particularly when the GAN loss can be combined with task-specific, auxiliary regression losses to mitigate training instability, ranging from paired image translation [24, 54, 79, 89] to unpaired image editing [37, 55, 94]. Still, GANs are a less popular choice for text-guided synthesis, as careful architectural design is needed to ensure training stability at large scale [26].

Lately, several works [1, 12, 85] drew connections between score-based models and distribution matching. In particular, ProlificDreamer [80] introduced Variational Score Distillation (VSD), which leverages a pretrained text-to-image diffusion model as a distribution matching loss. Since VSD can utilize a large pretrained model for unpaired settings [17, 58], it showed impressive results at particle-based optimization for text-conditioned 3D synthesis. Our method refines and extends VSD for training a deep generative neural network for distilling diffusion models. Furthermore, motivated by the success of GANs in image translation, we complement the stability of training with a regression loss. As a result, our method successfully attains high realism on a complex dataset like LAION [69]. Our method is different from recent works that combine GANs with diffusion [68, 81–83], as our formulation is not grounded in GANs. Our method shares motivation with concurrent works [50, 84] that leverage the VSD objective to train a generator, but differs in that we specialize the method for diffusion distillation by introducing regression loss and showing state-of-the-art results for text-to-image tasks.

3. Distribution Matching Distillation

Our goal is to distill a given pretrained diffusion denoiser, the *base model*, μ_{base}, into a fast "one-step" image generator, G_θ, that produces high-quality images without the costly iterative sampling procedure (Sec. 3.1). While we wish to produce samples from the same distribution, we do not necessarily seek to reproduce the exact mapping.



initial state | (a) real score only | (b) real+fake scores | (c) real+fake scores & regression loss

Figure 3. Optimizing various objectives starting from the same configuration (left) leads to different outcomes. (a) Maximizing the real score only, the fake samples all collapse to the closest mode of the real distribution. (b) With our distribution matching objective but not regression loss, the generated fake data covers more of the real distribution, but only recovers the closest mode, missing the second mode entirely. (c) Our full objective, with the regression loss, recovers both modes of the target distribution.

By analogy with GANs, we denote the outputs of the distilled model as *fake*, as opposed to the *real* images from the training distribution. We illustrate our approach in Figure 2. We train the fast generator by minimizing the sum of two losses: a distribution matching objective (Sec. 3.2), whose gradient update can be expressed as the difference of two score functions, and a regression loss (Sec. 3.3) that encourages the generator to match the large scale structure of the base model's output on a fixed dataset of noise-image pairs. Crucially, we use two diffusion denoisers to model the score functions of the real and fake distributions, respectively, perturbed with Gaussian noise of various magnitudes. Finally, in Section 3.4, we show how to adapt our training procedure with classifier-free guidance.

3.1. Pretrained base model and One-step generator

Our distillation procedure assumes a pretrained diffusion model μ_{base} is given. Diffusion models are trained to reverse a Gaussian diffusion process that progressively adds noise to a sample from a real data distribution $x_0 \sim p_{\text{real}}$, turning it into white noise $x_T \sim \mathcal{N}(0, \mathbf{I})$ over T time steps [21, 71, 74]; we use $T = 1000$. We denote the diffusion model as $\mu_{\text{base}}(x_t, t)$. Starting from a Gaussian sample x_T, the model iteratively denoises a running noisy estimate x_t, conditioned on the timestep $t \in \{0, 1, ..., T-1\}$ (or noise level), to produce a sample of the target data distribution. Diffusion models typically require 10 to 100s steps to produce realistic images. Our derivation uses the mean-prediction form of diffusion for simplicity [31] but works identically with ϵ-prediction [21, 63] with a change of variable [33] (see Appendix H). Our implementation uses pretrained models from EDM [31] and Stable Diffusion [63].

One-step generator. Our one-step generator G_θ has the architecture of the base diffusion denoiser but without time-conditioning. We initialize its parameters θ with the base model, i.e., $G_\theta(z) = \mu_{\text{base}}(z, T-1), \forall z$, before training.

3.2. Distribution Matching Loss

Ideally, we would like our fast generator to produce samples that are indistinguishable from real images. Inspired by the ProlificDreamer [80], we minimize the Kullback–Leibler (KL) divergence between the real and fake image distributions, p_{real} and p_{fake}, respectively:

$$\begin{aligned} D_{KL}\left(p_{\text{fake}} \,\|\, p_{\text{real}}\right) &= \mathop{\mathbb{E}}_{x \sim p_{\text{fake}}} \left(\log \left(\frac{p_{\text{fake}}(x)}{p_{\text{real}}(x)} \right) \right) \\ &= \mathop{\mathbb{E}}_{\substack{z \sim \mathcal{N}(0;\mathbf{I}) \\ x = G_\theta(z)}} -\big(\log\, p_{\text{real}}(x) - \log\, p_{\text{fake}}(x) \big). \end{aligned} \tag{1}$$

Computing the probability densities to estimate this loss is generally intractable, but we only need the gradient with respect to θ to train our generator by gradient descent.

Gradient update using approximate scores. Taking the gradient of Eq. (1) with respect to the generator parameters:

$$\nabla_\theta D_{KL} = \mathop{\mathbb{E}}_{\substack{z \sim \mathcal{N}(0;\mathbf{I}) \\ x = G_\theta(z)}} \Big[-\big(s_{\text{real}}(x) - s_{\text{fake}}(x) \big) \frac{dG}{d\theta} \Big], \tag{2}$$

where $s_{\text{real}}(x) = \nabla_x \log\, p_{\text{real}}(x)$, $s_{\text{fake}}(x) = \nabla_x \log\, p_{\text{fake}}(x)$ are the scores of the respective distributions. Intuitively, s_{real} moves x toward the modes of p_{real}, and $-s_{\text{fake}}$ spreads them apart, as shown in Figure 3(a, b). Computing this gradient is still challenging for two reasons: first, the scores diverge for samples with low probability — in particular p_{real} vanishes for fake samples, and second, our intended tool for estimating score, namely the diffusion models, only provide scores of the diffused distribution. Score-SDE [73, 74] provides an answer to these two issues.

By perturbing the data distribution with random Gaussian noise of varying standard deviations, we create a family of "blurred" distributions that are fully-supported over the ambient space, and therefore overlap, so that the gradient in Eq. (2) is well-defined (Figure 4). Score-SDE then shows that a trained diffusion model approximates the score function of the diffused distribution.

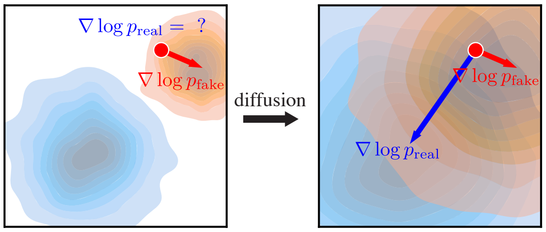
Accordingly, our strategy is to use a pair of diffusion denoisers to model the scores of the real and fake distributions after Gaussian diffusion. With slight abuse of notation, we define these as $s_{\text{real}}(x_t, t)$ and $s_{\text{fake}}(x_t, t)$, respectively. Diffused sample $x_t \sim q(x_t|x)$ is obtained by adding noise to generator output $x = G_\theta(z)$ at diffusion time step t:

$$q_t(x_t|x) \sim \mathcal{N}(\alpha_t x; \sigma_t^2 \mathbf{I}), \tag{3}$$

where α_t and σ_t are from the diffusion noise schedule.

Real score. The real distribution is fixed, corresponding to the training images of the base diffusion model, so we model its score using a fixed copy of the pretrained diffusion model $\mu_{\text{base}}(x, t)$. The score given a diffusion model is given by Song *et al.* [74]:

$$s_{\text{real}}(x_t, t) = -\frac{x_t - \alpha_t \mu_{\text{base}}(x_t, t)}{\sigma_t^2}. \tag{4}$$



(a) for unperturbed distributions, both scores may not be defined simulateneously everywhere

(b) after diffusion, the distributions overlap, making our objective well-defined

Figure 4. Without perturbation, the real/fake distributions may not overlap (a). Real samples only get a valid gradient from the real score, and fake samples from the fake score. After diffusion (b), our distribution matching objective is well-defined everywhere.

Dynamically-learned fake score. We derive the fake score function, in the same manner as the real score case:

$$s_{\text{fake}}(x_t, t) = -\frac{x_t - \alpha_t \mu_{\text{fake}}^{\phi}(x_t, t)}{\sigma_t^2}. \tag{5}$$

However, as the distribution of our generated samples changes throughout training, we dynamically adjust the fake diffusion model μ_{fake}^{ϕ} to track these changes. We initialize the fake diffusion model from the pretrained diffusion model μ_{base}, updating parameters ϕ during training, by minimizing a standard denoising objective [21, 77]:

$$\mathcal{L}_{\text{denoise}}^{\phi} = ||\mu_{\text{fake}}^{\phi}(x_t, t) - x_0||_2^2, \tag{6}$$

where $\mathcal{L}_{\text{denoise}}^{\phi}$ is weighted according to the diffusion timestep t, using the same weighting strategy employed during the training of the base diffusion model [31, 63].

Distribution matching gradient update. Our final approximate distribution matching gradient is obtained by replacing the exact score in Eq. (2) with those defined by the two diffusion models on the perturbed samples x_t and taking the expectation over the diffusion time steps:

$$\nabla_\theta D_{KL} \simeq \mathop{\mathbb{E}}_{z,t,x,x_t}\left[w_t \alpha_t \left(s_{\text{fake}}(x_t, t) - s_{\text{real}}(x_t, t)\right) \frac{dG}{d\theta}\right], \tag{7}$$

where $z \sim \mathcal{N}(0; \mathbf{I})$, $x = G_\theta(z)$, $t \sim \mathcal{U}(T_{\min}, T_{\max})$, and $x_t \sim q_t(x_t|x)$. We include the derivations in Appendix F. Here, w_t is a time-dependent scalar weight we add to improve the training dynamics. We design the weighting factor to normalize the gradient's magnitude across different noise levels. Specifically, we compute the mean absolute error across spatial and channel dimensions between the denoised image and the input, setting

$$w_t = \frac{\sigma_t^2}{\alpha_t} \frac{CS}{||\mu_{\text{base}}(x_t, t) - x||_1}, \tag{8}$$

where S is the number of spatial locations and C is the number of channels. In Sec. 4.2, we show that this weighting outperforms previous designs [58, 80]. We set $T_{\min} = 0.02T$ and $T_{\max} = 0.98T$, following DreamFusion [58].

3.3. Regression loss and final objective

The distribution matching objective introduced in the previous section is well-defined for $t \gg 0$, i.e., when the generated samples are corrupted with a large amount of noise. However, for a small amount of noise, $s_{\text{real}}(x_t, t)$ often becomes unreliable, as $p_{\text{real}}(x_t, t)$ goes to zero. Furthermore, as the score $\nabla_x \log(p)$ is invariant to scaling of probability density function p, the optimization is susceptible to mode collapse/dropping, where the fake distribution assigns higher overall density to a subset of the modes. To avoid this, we use an additional regression loss to ensure all modes are preserved; see Figure 3(b), (c).

This loss measures the pointwise distance between the generator and base diffusion model outputs, given the *same* input noise. Concretely, we build a paired dataset $\mathcal{D} = \{z, y\}$ of random Gaussian noise images z and the corresponding outputs y, obtained by sampling the pretrained diffusion model μ_{base} using a deterministic ODE solver [31, 41, 72]. In our CIFAR-10 and ImageNet experiments, we utilize the Heun solver from EDM [31], with 18 steps for CIFAR-10 and 256 steps for ImageNet. For the LAION experiments, we use the PNDM [41] solver with 50 sampling steps. We find that even a small number of noise–image pairs, generated using less than 1% of the training compute, in the case of CIFAR10, for example, acts as an effective regularizer. Our regression loss is given by:

$$\mathcal{L}_{\text{reg}} = \mathop{\mathbb{E}}_{(z,y)\sim\mathcal{D}} \ell(G_\theta(z), y). \tag{9}$$

We use Learned Perceptual Image Patch Similarity (LPIPS) [88] as the distance function ℓ, following InstaFlow [43] and Consistency Models [75].

Final objective. Network μ_{fake}^{ϕ} is trained with $\mathcal{L}_{\text{denoise}}^{\phi}$, which is used to help calculate $\nabla_\theta D_{KL}$. For training G_θ, the final objective is $D_{KL} + \lambda_{\text{reg}}\mathcal{L}_{\text{reg}}$, using $\lambda_{\text{reg}} = 0.25$ unless otherwise specified. The gradient $\nabla_\theta D_{KL}$ is computed in Eq. (7), and gradient $\nabla_\theta \mathcal{L}_{\text{reg}}$ is computed from Eq. (9) with automatic differentiation. We apply the two losses to distinct data streams: unpaired fake samples for the distribution matching gradient and paired examples described in Section 3.3 for the regression loss. Algorithm 1 outlines the final training procedure. Additional details are provided in Appendix B.

3.4. Distillation with classifier-free guidance

Classifier-Free Guidance [20] is widely used to improve the image quality of text-to-image diffusion models. Our approach also applies to diffusion models that use classifier-free guidance. We first generate the corresponding noise-output pairs by sampling from the guided model to construct the paired dataset needed for regression loss $\mathcal{L}_{\text{reg}}$. When computing the distribution matching gradient $\nabla_\theta D_{KL}$, we substitute the real score with that derived from the mean

Algorithm 1: DMD Training procedure

Input: Pretrained real diffusion model μ_{real}, paired dataset $\mathcal{D} = \{z_{\text{ref}}, y_{\text{ref}}\}$
Output: Trained generator G.
// Initialize generator and fake score estimators from pretrained model
$G \leftarrow \text{copyWeights}(\mu_{\text{real}}),\ \mu_{\text{fake}} \leftarrow \text{copyWeights}(\mu_{\text{real}})$
while *train* **do**
  // Generate images
  Sample batch $z \sim \mathcal{N}(0, \mathbf{I})^B$ and $(z_{\text{ref}}, y_{\text{ref}}) \sim \mathcal{D}$
  $x \leftarrow G(z),\ x_{\text{ref}} \leftarrow G(z_{\text{ref}})$
  $x = \text{concat}(x, x_{\text{ref}})$ **if** dataset is LAION **else** x

  // Update generator
  $\mathcal{L}_{\text{KL}} \leftarrow \text{distributionMatchingLoss}(\mu_{\text{real}}, \mu_{\text{fake}}, x)$ // Eq 7
  $\mathcal{L}_{\text{reg}} \leftarrow \text{LPIPS}(x_{\text{ref}}, y_{\text{ref}})$ // Eq 9
  $\mathcal{L}_G \leftarrow \mathcal{L}_{\text{KL}} + \lambda_{\text{reg}} \mathcal{L}_{\text{reg}}$
  $G \leftarrow \text{update}(G, \mathcal{L}_G)$

  // Update fake score estimation model
  Sample time step $t \sim \mathcal{U}(0, 1)$
  $x_t \leftarrow \text{forwardDiffusion}(\text{stopgrad}(x), t)$
  $\mathcal{L}_{\text{denoise}} \leftarrow \text{denoisingLoss}(\mu_{\text{fake}}(x_t, t), \text{stopgrad}(x))$ // Eq 6
  $\mu_{\text{fake}} \leftarrow \text{update}(\mu_{\text{fake}}, \mathcal{L}_{\text{denoise}})$
end while

prediction of the guided model. Meanwhile, we do not modify the formulation for the fake score. We train our one-step generator with a fixed guidance scale.

4. Experiments

We assess the capabilities of our approach using several benchmarks, including class-conditional generation on CIFAR-10 [36] and ImageNet [8]. We use the Fréchet Inception Distance (FID) [18] to measure image quality and CLIP Score [59] to evaluate text-to-image alignment. First, we perform a direct comparison on ImageNet (Sec. 4.1), where our distribution matching distillation substantially outperforms competing distillation methods with identical base diffusion models. Second, we perform detailed ablation studies verifying the effectiveness of our proposed modules (Sec. 4.2). Third, we train a text-to-image model on the LAION-Aesthetic-6.25+ dataset [69] with a classifier-free guidance scale of 3 (Sec. 4.3). In this phase, we distill Stable Diffusion v1.5, and we show that our distilled model achieves FID comparable to the original model, while offering a 30× speed-up. Finally, we train another text-to-image model on LAION-Aesthetic-6+, utilizing a higher guidance value of 8 (Sec. 4.3). This model is tailored to enhance visual quality rather than optimize the FID metric. Quantitative and qualitative analysis confirm that models trained with our distribution matching distillation procedure can produce high-quality images rivaling Stable Diffusion. We describe additional training and evaluation details in the appendix.

4.1. Class-conditional Image Generation

We train our model on class-conditional ImageNet-64×64 and benchmark its performance with competing methods. Results are shown in Table 1. Our model surpasses established GANs like BigGAN-deep [4] and recent diffusion distillation methods, including the Consistency Model [75] and TRACT [3]. Our method remarkably bridges the fidelity gap, achieving a near-identical FID score (within 0.3) compared to the original diffusion model, while also attaining a 512-fold increase in speed. On CIFAR-10, our class-conditional model reaches a competitive FID of 2.66. We include the CIFAR-10 results in the appendix.

Method	# Fwd Pass (↓)	FID (↓)
BigGAN-deep [4]	1	4.06
ADM [9]	250	**2.07**
Progressive Distillation [65]	1	15.39
DFNO [91]	1	7.83
BOOT [16]	1	16.30
TRACT [3]	1	7.43
Meng et al. [51]	1	7.54
Diff-Instruct [50]	1	5.57
Consistency Model [75]	1	6.20
DMD (Ours)	1	**2.62**
EDM† (Teacher) [31]	512	2.32

Table 1. Sample quality comparison on ImageNet-64×64. Baseline numbers are derived from Song et al. [75]. The upper section of the table highlights popular diffusion and GAN approaches [4, 9]. The middle section includes a list of competing diffusion distillation methods. The last row shows the performance of our teacher model, EDM† [31].

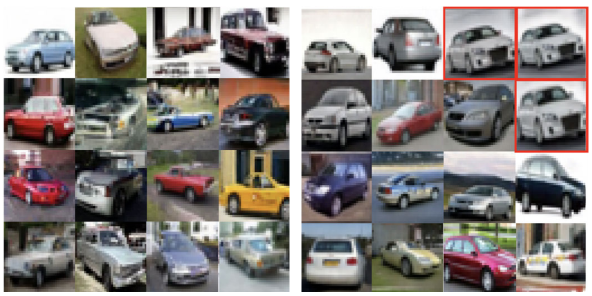
4.2. Ablation Studies

We first compare our method with two baselines: one omitting the distribution matching objective and the other missing the regression loss in our framework. Table 2 (left) summarizes the results. In the absence of distribution matching loss, our baseline model produces images that lack realism and structural integrity, as illustrated in the top section of Figure 5. Likewise, omitting the regression loss leads to training instability and a propensity for mode collapse, resulting in a reduced diversity of the generated images. This issue is illustrated in the bottom section of Figure 5.

Table 2 (right) demonstrates the advantage of our proposed sample weighting strategy (Section 3.2). We compare with σ_t/α_t and σ_t^3/α_t, two popular weighting schemes utilized by DreamFusion [58] and ProlificDreamer [80]. Our weighting strategy achieves a healthy 0.9 FID improvement as it normalizes the gradient magnitudes across noise levels and stabilizes the optimization.

DMD (ours) without distribution matching

(a) Qualitative comparison between our model (*left*) and the baseline model excluding the distribution matching objective (*right*). The baseline model generates images with compromised realism and structural integrity. Images are generated from the same random seed.



DMD (ours) without regression loss

(b) Qualitative comparison between our model (*left*) and the baseline model omitting the regression loss (*right*). The baseline model tends to exhibit mode collapse and a lack of diversity, as evidenced by the predominant appearance of the grey car (highlighted with a red square). Images are generated from the same random seed.

Figure 5. Ablation studies of our training loss, including the distribution matching objective (top) and the regression loss (bottom).

Training loss	CIFAR	ImageNet
w/o Dist. Matching	3.82	9.21
w/o Regress. Loss	5.58	5.61
DMD (Ours)	**2.66**	**2.62**

Sample weighting	CIFAR
σ_t/α_t [58]	3.60
σ_t^3/α_t [58, 80]	3.71
Eq. 8 (Ours)	**2.66**

Table 2. **Ablation study.** *(left)* We ablate elements of our training loss. We show the FID results on CIFAR-10 and ImageNet-64×64. *(right)* We compare different sample weighting strategies for the distribution matching loss.

4.3. Text-to-Image Generation

We use zero-shot MS COCO to evaluate our model's performance for text-to-image generation. We train a text-to-image model by distilling Stable Diffusion v1.5 [63] on the LAION-Aesthetics-6.25+ [69]. We use a guidance scale of 3, which yields the best FID for the base Stable Diffusion model. The training takes around 36 hours on a cluster of 72 A100 GPUs. Table 3 compares our model to state-of-the-art approaches. Our method showcases superior performance over StyleGAN-T [67], surpasses all other diffusion acceleration methods, including advanced diffusion solvers [46, 90], and diffusion distillation techniques such as Latent Consistency Models [48, 49], UFOGen [83], and InstaFlow [43]. We substantially close the gap between distilled and base models, reaching within 2.7 FID from Stable Diffusion v1.5, while running approximately 30× faster. With FP16 inference, our model generates images at 20 frames per second, enabling interactive applications.

Family	Method	Resolution (↑)	Latency (↓)	FID (↓)
Original, unaccelerated	DALL·E [60]	256	-	27.5
	DALL·E 2 [61]	256	-	10.39
	Parti-750M [86]	256	-	10.71
	Parti-3B [86]	256	6.4s	8.10
	Make-A-Scene [13]	256	25.0s	11.84
	GLIDE [52]	256	15.0s	12.24
	LDM [63]	256	3.7s	12.63
	Imagen [64]	256	9.1s	7.27
	eDiff-I [2]	256	32.0s	**6.95**
GANs	LAFITE [93]	256	0.02s	26.94
	StyleGAN-T [67]	512	0.10s	13.90
	GigaGAN [26]	512	0.13s	**9.09**
Accelerated diffusion	DPM++ (4 step) [46]†	512	0.26s	22.36
	UniPC (4 step) [90]†	512	0.26s	19.57
	LCM-LoRA (4 step)[49]†	512	0.19s	23.62
	InstaFlow-0.9B [43]	512	0.09s	13.10
	UFOGen [83]	512	0.09s	12.78
	DMD (Ours)	512	0.09s	**11.49**
Teacher	SDv1.5† [63]	512	2.59s	8.78

Table 3. **Sample quality comparison on zero-shot text-to-image generation on MS COCO-30k.** Baseline numbers are derived from GigaGAN [26]. The dashed line indicates that the result is unavailable. †Results are evaluated by us using the released models. LCM-LoRA is trained with a guidance scale of 7.5. We use a guidance scale of 3 for all the other methods. Latency is measured with a batch size of 1.

High guidance-scale diffusion distillation. For text-to-image generation, diffusion models typically operate with a high guidance scale to enhance image quality [57, 63]. To evaluate our distillation method in this high guidance-scale regime, we trained an additional text-to-image model. This model distills SD v1.5 using a guidance scale of 8 on the LAION-Aesthetics-6+ dataset [69]. Table 4 benchmarks our approach against various diffusion acceleration methods [46, 49, 90]. Similar to the low guidance model, our one-step generator significantly outperforms competing methods, even when they utilize a four-step sampling process. Qualitative comparisons with competing approaches and the base diffusion model are shown in Figure 6.

5. Limitations

While our results are promising, a slight quality discrepancy persists between our one-step model and finer discretizations of the diffusion sampling path, such as those with 100 or 1000 neural network evaluations. Additionally, our framework fine-tunes the weights of both the fake score

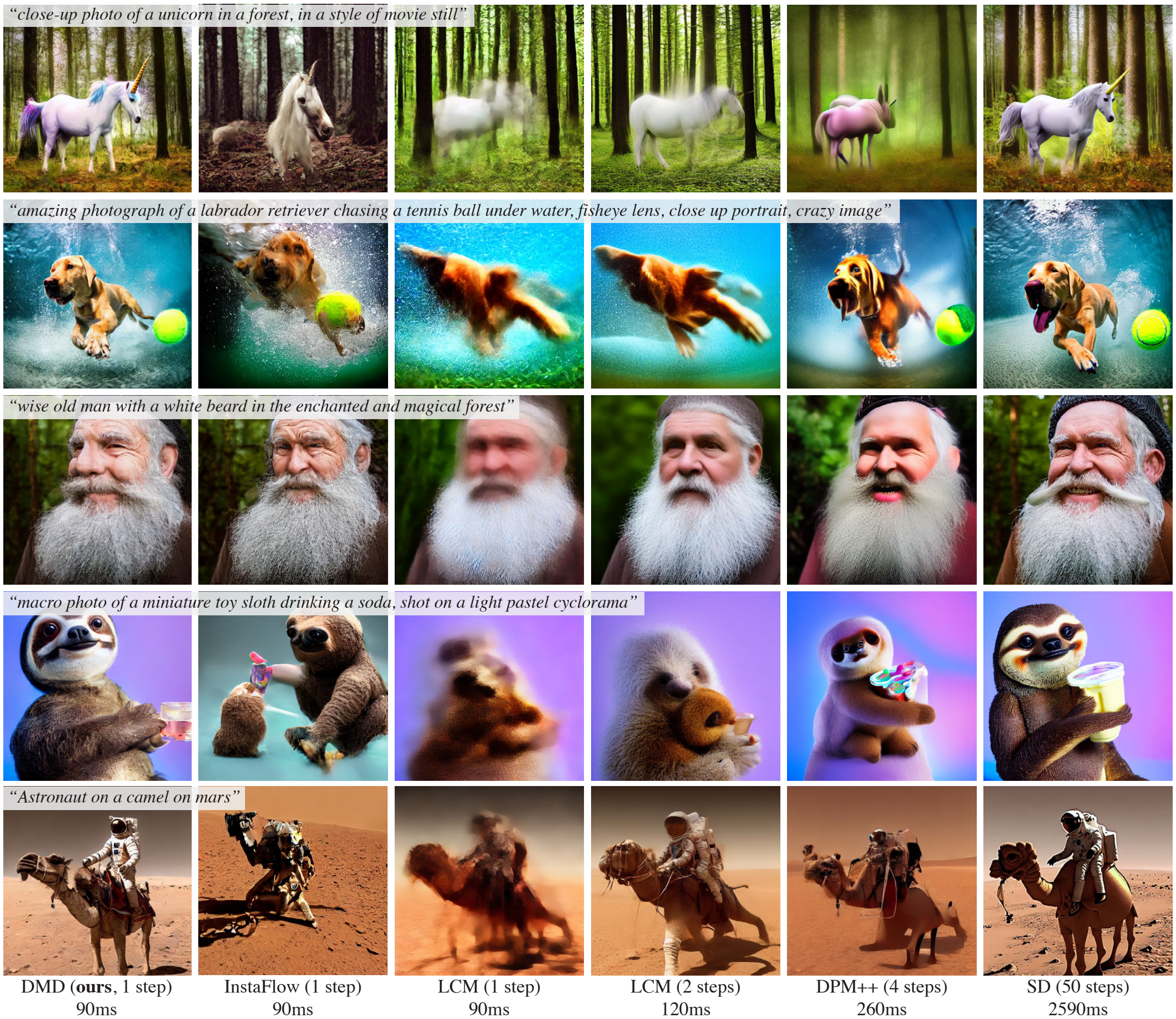


Figure 6. Starting from a pretrained diffusion model, here Stable Diffusion (right), our distribution matching distillation algorithm yields a model that can generate images with much higher quality (left) than previous few-steps generators (middle), with the same speed or faster.

Method	Latency (↓)	FID (↓)	CLIP-Score (↑)
DPM++ (4 step)[46]†	0.26s	22.44	0.309
UniPC (4 step)[90]†	0.26s	23.30	0.308
LCM-LoRA (1 step) [49]†	0.09s	77.90	0.238
LCM-LoRA (2 step) [49]†	0.12s	24.28	0.294
LCM-LoRA (4 step) [49]†	0.19s	23.62	0.297
DMD (Ours)	0.09s	**14.93**	**0.320**
SDv1.5† (Teacher) [63]	2.59s	13.45	0.322

Table 4. **FID/CLIP-Score comparison on MS COCO-30K.** †Results are evaluated by us. LCM-LoRA is trained with a guidance scale of 7.5. We use a guidance scale of 8 for all the other methods. Latency is measured with a batch size of 1.

function and the generator, leading to significant memory usage during training. Techniques such as LORA offer potential solutions for addressing this issue.

Acknowledgements

This work was started while TY was an intern at Adobe Research. We are grateful for insightful discussions with Yilun Xu, Guangxuan Xiao, and Minguk Kang. This work is supported, in part by NSF grants 2105819, 1955864, and 2019786 (IAIFI), by the Singapore DSTA under DST00OECI20300823 (New Representations for Vision), as well as by funding from GIST and Amazon.

References

[1] Siddarth Asokan, Nishanth Shetty, Aadithya Srikanth, and Chandra Sekhar Seelamantula. Gans settle scores! *arXiv preprint arXiv:2306.01654*, 2023. 3

[2] 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. 2, 7

[3] 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. 2, 3, 6, 14

[4] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *ICLR*, 2019. 3, 6, 14

[5] Huiwen Chang, Han Zhang, Jarred Barber, AJ Maschinot, Jose Lezama, Lu Jiang, Ming-Hsuan Yang, Kevin Murphy, William T Freeman, Michael Rubinstein, et al. Muse: Text-to-image generation via masked generative transformers. In *ICML*, 2023. 3

[6] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation. In *ICLR*, 2021. 2

[7] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016. 13

[8] 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. 2, 6

[9] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *NeurIPS*, 2021. 6

[10] Gintare Karolina Dziugaite, Daniel M Roy, and Zoubin Ghahramani. Training generative neural networks via maximum mean discrepancy optimization. In *UAI*, 2015. 3

[11] Patrick Esser, Johnathan Chiu, Parmida Atighehchian, Jonathan Granskog, and Anastasis Germanidis. Structure and content-guided video synthesis with diffusion models. In *CVPR*, 2023. 2

[12] 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. 3

[13] 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. 7

[14] Xinyu Gong, Shiyu Chang, Yifan Jiang, and Zhangyang Wang. Autogan: Neural architecture search for generative adversarial networks. In *ICCV*, 2019. 14

[15] 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. 1, 2, 3

[16] 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. 2, 3, 6

[17] Amir Hertz, Kfir Aberman, and Daniel Cohen-Or. Delta denoising score. In *ICCV*, 2023. 3

[18] 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. 6

[19] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NeurIPS 2014 Deep Learning Workshop*, 2015. 2

[20] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *arXiv preprint arXiv:2207.12598*, 2022. 5

[21] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. 1, 2, 3, 4, 5

[22] 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. 2

[23] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *JMLR*, 2005. 2

[24] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 2, 3

[25] Yifan Jiang, Shiyu Chang, and Zhangyang Wang. Transgan: Two pure transformers can make one strong gan, and that can scale up. In *NeurIPS*, 2021. 14

[26] 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. 2, 3, 7, 13

[27] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *ICLR*, 2018. 2

[28] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 3

[29] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In *NeurIPS*, 2020. 14

[30] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020. 2, 3, 14

[31] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *NeurIPS*, 2022. 2, 4, 5, 6, 12, 14

[32] Sergey Kastryulin, Jamil Zakirov, Denis Prokopenko, and Dmitry V. Dylov. Pytorch image quality: Metrics for image quality assessment, 2022. 12, 13

[33] Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. In *NeurIPS*, 2021. 4

[34] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014. 1

[35] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. In *ICLR*, 2021. 2

[36] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009. 2, 6

[37] Hsin-Ying Lee, Hung-Yu Tseng, Qi Mao, Jia-Bin Huang, Yu-Ding Lu, Maneesh Singh, and Ming-Hsuan Yang. Drit++: Diverse image-to-image translation via disentangled representations. *IJCV*, 2020. 3

[38] Kwonjoon Lee, Huiwen Chang, Lu Jiang, Han Zhang, Zhuowen Tu, and Ce Liu. Vitgan: Training gans with vision transformers. In *ICLR*, 2022. 14

[39] Yujia Li, Kevin Swersky, and Rich Zemel. Generative moment matching networks. In *ICML*, 2015. 2, 3

[40] 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. 2

[41] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. In *ICLR*, 2022. 2, 5, 13

[42] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023. 1, 2, 3, 14

[43] 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. 1, 2, 3, 5, 7

[44] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 12, 13

[45] 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. 2, 14

[46] 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. 2, 7, 8, 13, 14

[47] Eric Luhman and Troy Luhman. Knowledge distillation in iterative generative models for improved sampling speed. *arXiv preprint arXiv:2101.02388*, 2021. 1, 2, 14

[48] 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. 1, 2, 7, 13

[49] 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. 7, 8, 13

[50] 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. *arXiv preprint arXiv:2305.18455*, 2023. 3, 6, 14

[51] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *CVPR*, 2023. 1, 2, 3, 6, 14

[52] 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. 7

[53] Ollin. Tiny autoencoder for stable diffusion. `https://github.com/madebyollin/taesd`, 2023. 13

[54] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019. 3

[55] Taesung Park, Jun-Yan Zhu, Oliver Wang, Jingwan Lu, Eli Shechtman, Alexei Efros, and Richard Zhang. Swapping autoencoder for deep image manipulation. In *NeurIPS*, 2020. 3

[56] Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. On aliased resizing and surprising subtleties in gan evaluation. In *CVPR*, 2022. 14

[57] 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. 7

[58] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *ICLR*, 2023. 3, 5, 6, 7

[59] 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. 6

[60] 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. 3, 7

[61] 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. 1, 2, 3, 7

[62] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *ICML*, 2016. 2

[63] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 1, 2, 4, 5, 7, 8, 13

[64] 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. 1, 2, 3, 7

[65] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *ICLR*, 2022. 1, 2, 3, 6, 14

[66] Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *SIGGRAPH*, 2022. 14

[67] 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. 3, 7

[68] Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. *arXiv preprint arXiv:2311.17042*, 2023. 3

[69] 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. 3, 6, 7, 13

[70] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022. 2

[71] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015. 1, 2, 4

[72] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021. 2, 5, 14

[73] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *NeurIPS*, 2019. 2, 4

[74] 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. 1, 2, 4

[75] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *ICML*, 2023. 1, 2, 3, 5, 6, 12, 13, 14

[76] Yuan Tian, Qin Wang, Zhiwu Huang, Wen Li, Dengxin Dai, Minghao Yang, Jun Wang, and Olga Fink. Off-policy reinforcement learning for efficient and effective gan architecture search. In *ECCV*, 2020. 14

[77] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 2011. 5

[78] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. `https://github.com/huggingface/diffusers`, 2022. 13

[79] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*, 2018. 3

[80] 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. 2, 3, 4, 5, 6, 7

[81] Zhendong Wang, Huangjie Zheng, Pengcheng He, Weizhu Chen, and Mingyuan Zhou. Diffusion-gan: Training gans with diffusion. In *ICLR*, 2023. 3, 14

[82] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion gans. In *ICLR*, 2022. 2, 14

[83] Yanwu Xu, Yang Zhao, Zhisheng Xiao, and Tingbo Hou. Ufogen: You forward once large scale text-to-image generation via diffusion gans. *arXiv preprint arXiv:2311.09257*, 2023. 3, 7

[84] Senmao Ye and Fei Liu. Score mismatching for generative modeling. *arXiv preprint arXiv:2309.11043*, 2023. 3, 14

[85] Mingxuan Yi, Zhanxing Zhu, and Song Liu. Monoflow: Rethinking divergence gans via the perspective of wasserstein gradient flows. In *ICML*, 2023. 3

[86] 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*, 2(3):5, 2022. 3, 7

[87] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. *TPAMI*, 2018. 2

[88] 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. 3, 5

[89] Shengyu Zhao, Jonathan Cui, Yilun Sheng, Yue Dong, Xiao Liang, Eric I Chang, and Yan Xu. Large scale image completion via co-modulated generative adversarial networks. In *ICLR*, 2021. 3

[90] 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. 1, 2, 7, 8, 13

[91] Hongkai Zheng, Weili Nie, Arash Vahdat, Kamyar Azizzadenesheli, and Anima Anandkumar. Fast sampling of diffusion models via operator learning. In *ICML*, 2023. 1, 2, 6, 14

[92] Hongkai Zheng, Weili Nie, Arash Vahdat, Kamyar Azizzadenesheli, and Anima Anandkumar. Fast sampling of diffusion models via operator learning. In *ICML*, 2023. 2

[93] 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. 7

[94] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 3