Towards Training One-Step Diffusion Models Without Distillation

Mingtian Zhang[*]
University College London

Wenlin Chen[*]
University of Cambridge
MPI for Intelligent Systems

Jiajun He[*]
University of Cambridge

Zijing Ou
Imperial College London

José Miguel Hernández-Lobato
University of Cambridge

Bernhard Schölkopf
MPI for Intelligent Systems

David Barber
University College London

Abstract

Recent advances in training one-step diffusion models typically follow a two-stage pipeline: first training a teacher diffusion model and then distilling it into a one-step student model. This process often depends on both the teacher's score function for supervision and its weights for initializing the student model. In this paper, we explore whether one-step diffusion models can be trained directly without this distillation procedure. We introduce a family of new training methods that entirely forgo teacher score supervision, yet outperforms most teacher-guided distillation approaches. This suggests that score supervision is not essential for effective training of one-step diffusion models. However, we find that initializing the student model with the teacher's weights remains critical. Surprisingly, the key advantage of teacher initialization is not due to better latent-to-output mappings, but rather the rich set of feature representations across different noise levels that the teacher diffusion model provides. These insights take us one step closer towards training one-step diffusion models without distillation and provide a better understanding of the roles of teacher supervision and initialization in the distillation process.

1 Introduction

Diffusion models [45, 16, 50] have achieved remarkable success in modeling complex real-world data across a wide range of domains, including image synthesis [41, 26], 3D generation [37], video synthesis [17], equivariant modeling [18], and audio generation [27]. Typically, diffusion models consist of two processes: a forward noising process, which gradually perturbs data into a known noise prior (typically Gaussian noise), and a reverse denoising process, which learns to invert the forward corruption process to generate realistic data samples from noise. Formally, the forward process is defined over T time steps as a Markov chain of Gaussian transitions, while the reverse process is parameterized using neural networks that predict the denoising distribution given the noisy samples.

In classic diffusion models with Gaussian denoising distributions, generating high-quality data samples typically requires hundreds or thousands of sampling steps, resulting in significant sampling inefficiency due to the need for $T \gg 1$ NFEs (number of function evaluations) [16, 34]. To address this weakness, various acceleration methods have been proposed to reduce NFEs during sampling. One class of such approaches leverages advanced numerical solvers for differential equations, enabling

[*]Equal contribution. Correspondence to: Mingtian Zhang <m.zhang@cs.ucl.ac.uk>, Wenlin Chen <wc337@cam.ac.uk>, and Jiajun He <jh2383@cam.ac.uk>.

continuous-time approximations of the diffusion process [47, 28, 30]. Another line of work improves the flexibility of the posterior distribution in the denoising process, either by estimating a more accurate covariance for the Gaussian distribution [34, 4, 3, 36] or by adopting flexible non-Gaussian denoising distributions [6, 55, 57]. While these techniques can dramatically reduce NFEs from $\sim 10^3$ to around 10–20, they still fall short of achieving high-quality generation within 5 steps.

Recently, distillation-based methods have emerged as a powerful direction for training diffusion models, enabling high-quality *one-step* generation [64]. These methods fall into two categories:

- **Trajectory-based distillation methods** [43, 5, 49, 15, 23, 25] aim to approximate the full sampling trajectory by training a student model to amortize multiple intermediate steps. These methods are motivated by accelerated solvers and typically perform joint training of the full diffusion model and the distillation process.
- **Score-based distillation approaches** [31, 44, 56, 64] distill the full denoising process of the pre-trained teacher diffusion model into a one-step latent variable model. This distillation process typically involves minimizing the divergence between the student and teacher models based on their respective score estimations [37, 54].

In this paper, we focus on score-based distillation methods and investigate whether a one-step diffusion model can be effectively trained without relying on a pre-trained teacher diffusion model. In existing distillation approaches, the teacher model is typically used in two key places: (1) the teacher's score function is used to estimate the gradient for training the student model, and (2) the teacher's weights are used to initialize the student model. The goal of this paper is to investigate whether it is possible to train a one-step diffusion model without using either the teacher's score function or its weight initialization.

Our main contributions are summarized as follows:

1. We propose a novel distillation method that eliminates the need for both teacher and student score estimation during training. Despite this simplification, our method outperforms most one-step generation approaches and achieves competitive performance to the state-of-the-art method on image generation tasks without the supervision of the teacher model's score function.
2. We further analyze the importance of initializing the student model with teacher model's weights from both the weight-space and function-space perspectives, providing deeper insights into the role of teacher weight initialization. This analysis lays the groundwork for future efforts toward training one-step diffusion models entirely without reliance on a teacher model.

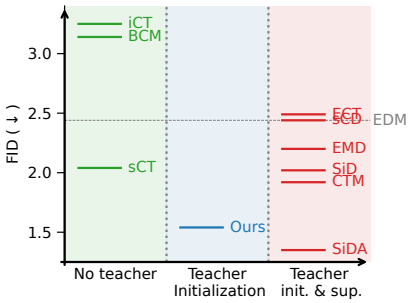


Figure 1: FID comparison of one-step generation on ImageNet 64×64. Our method achieves competitive performance to the state-of-the-art method without the supervision of the teacher model (EDM)'s score function.

Before introducing our proposed method, we will first establish the background on diffusion models and score-based distillation methods in the next section.

2 Background

2.1 Denoising Diffusion Models

Let $\{x^{(1)}, \ldots, x^{(N)}\}$ denote data samples from the true data distribution $p_d(x_0)$. Diffusion models [45, 16, 50] define a generative modeling framework that transforms samples from a simple Gaussian prior $p(x_T)$ into complex data distributions $p(x_0)$ through a learned denoising process. The model consists of two main components: a forward noising process and a reverse denoising process.

The forward process defines a Markov chain that progressively adds Gaussian noise to the data:

$$q(x_{0:T}) = p_d(x_0) \prod_{t=1}^{T} q(x_t|x_{t-1}), \tag{1}$$

with transition kernels defined as

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t|\sqrt{1-\beta_t}\, x_{t-1}, \beta_t I), \tag{2}$$

where $\beta_t \in (0,1)$ is a pre-specified variance schedule. This process gradually perturbs the data until it resembles an isotropic Gaussian distribution. The skip distribution at time t can be written as:

$$q(x_t|x_0) = \mathcal{N}(x_t|\sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I), \tag{3}$$

where $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$. As $T \to \infty$, the final state x_T approximates a standard normal distribution, i.e., $q(x_T) \to \mathcal{N}(0, I)$.

The generative process aims to reverse this trajectory. Starting from noise $x_T \sim p(x_T) = \mathcal{N}(0, I)$, the model learns a reverse process to sequentially denoise and reconstruct data samples. Since the true reverse conditional $q(x_{t-1}|x_t)$ is intractable, a common method is to approximate it with a variational Gaussian distribution:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}|\mu_{t-1}(x_t;\theta), \Sigma_{t-1}(x_t;\theta)), \tag{4}$$

where the mean function is learned from data and the covariance function can be either learned [34, 36, 3] or chosen to be a fixed value [4, 16].

From a score-based perspective, learning the mean function $\mu_{t-1}(x_t;\theta)$ is equivalent to learning the score function $\nabla_{x_t}\log q(x_t)$, which represents the gradient of the log-density of the noised data x_t. This score can be approximated using denoising score matching (DSM) [52, 50], and transformed into the reverse mean through Tweedie's Lemma [8, 40]:

$$\mu_{t-1}(x_t;\theta) = \frac{1}{\sqrt{1-\beta_t}}\left(x_t + \beta_t \nabla_{x_t}\log p_\theta(x_t)\right), \tag{5}$$

where $\nabla_{x_t}\log p_\theta(x_t) \approx \nabla_{x_t}\log q(x_t)$. This establishes a connection between the denoising distributional perspective and the score estimation perspective of diffusion models. In the following section, we introduce score-based distillation methods through the lens of divergence minimization.

2.2 Score-Based Distillation Methods

Score-based distillation methods aim to distill a teacher diffusion model p_θ (pre-trained on the true data distribution $p_d(x_0)$) into a one-step implicit generative model [11, 19, 59]:

$$q_\theta(x_0) = \int \delta(x_0 - g_\theta(z))p(z)dz, \tag{6}$$

where $\delta(\cdot)$ is the Dirac delta function, $p(z)$ is a standard Gaussian prior for the latent variable z, and $g_\theta : \mathcal{Z} \to \mathcal{X}$ is a deterministic neural network that generates data x from the latent variable z in one step. We emphasize that when the function $g_\theta(\cdot)$ is not bijective, the model distribution q_θ is not absolutely continuous with respect to the Lebesgue measure. As a result, the corresponding density function may not be well-defined, and consequently, the KL divergence between $q_\theta(x_0)$ and the data distribution $p_d(x_0)$ may also be ill-defined [2, 59].

Inspired by diffusion models, one can use a set of (scaled) Gaussian convolution kernels $\mathcal{K} = \{k_1, \cdots, k_T\}$ defined by $k_t(x_t|x_0) = \mathcal{N}(x_t|\alpha_t x_0, \sigma_t^2 I)$ to define the Diffusive KL divergence (DiKL) between the model density $q_\theta(x_0)$ and target distribution $p_d(x_0)$:

$$\mathrm{DiKL}_{\mathcal{K}}(q_\theta(x_0)||p_d(x_0)) \equiv \sum_{t=1}^{T} w(t)\mathrm{KL}(q_\theta(x_t)||p_d(x_t)), \tag{7}$$

where $w(t)$ is a positive scalar weighting function that sums to one, and $q_\theta(x_t)$ and $p_d(x_t)$ are noisy model density and noisy target density, respectively, defined by

$$q_\theta(x_t) = \int q_\theta(x_0)k_t(x_t|x_0)dx_0 \quad \text{and} \quad p_d(x_t) = \int p_d(x_0)k_t(x_t|x_0)dx_0. \tag{8}$$

In this case, the model distribution $q_\theta(x_t)$ is always absolutely continuous, and thus the KL divergence between them is always well-defined. For a single Gaussian kernel, the divergence was previously known as *Spread KL divergence* [59, 58]. It is straightforward to show that it is a valid divergence, i.e.,

Algorithm 1 Score-Based Distillation of One-Step Diffusion Models

Require: Data samples $\{x^{(1)}, \ldots, x^{(N)}\} \sim p_d(x_0)$

Stage 1: Train a multi-step teacher diffusion model

1: Train the teacher model's score network $s^{p_d}_{\psi_1}(x_t, t)$ using DSM (Eq. 10) until convergence

Stage 2: Train a one-step student generative model

2: Initialize the one-step generator with the teacher's score network $g_{\theta_{\text{init}}}(\cdot) \equiv s^{p_d}_{\psi_1}(\cdot, t = t_{\text{init}})$
3: **for** each training iteration **do**
4: Estimate the student model's score by training a score network $s^{q_\theta}_{\psi_2}(x_t, t)$ using DSM
5: Update the one-step generator g_θ with $s^{q_\theta}_{\psi_2}(x_t, t)$ and $s^{p_d}_{\psi_1}(x_t, t)$ using Eq. 12
6: **end for**

$\text{DiKL}_{\mathcal{K}}(q_\theta || p_d) = 0 \Leftrightarrow q_\theta = p_d$; see [59] for a proof. In addition to the diffusion distillation [31, 56], this divergence has successfully been used in 3D generative models [38, 54] and training neural samplers [14].

Without loss of generality, we consider a single Gaussian convolution kernel k_t. The gradient of DiKL with respect to the model parameters θ can be obtained analytically [14]:

$$\nabla_\theta \text{KL}(q_\theta(x_t) || p_d(x_t)) = \int q_\theta(x_t) \left(\nabla_{x_t} \log q_\theta(x_t) - \nabla_{x_t} \log p_d(x_t)\right) \frac{\partial x_t}{\partial \theta} dx_t. \tag{9}$$

However, neither the noisy model score $\nabla_{x_t} \log q_\theta(x_t)$ nor the noisy target score $\nabla_{x_t} \log p_d(x_t)$ are directly accessible. In the distillation setting, the noisy target score is provided by a pre-trained diffusion model, which has been trained using *denoising score matching* (DSM) [52]. Specifically, the score network $s^{p_d}_{\psi_1}(x_t, t) \approx \nabla_{x_t} \log p_d(x_t)$ provides an estimate of the noisy data score based on access to samples from $p_d(x_t)$ and the tractable score $\nabla_{x_t} \log k_t(x_t \mid x_0)$, which is learned by

$$\min_{\psi_1} \mathcal{L}_{\text{DSM}}(\psi_1) = \frac{1}{2} \iint \| s^{p_d}_{\psi_1}(x_t, t) - \nabla_{x_t} \log k_t(x_t|x_0) \|_2^2 p_d(x_0) p(x_t|x_0) dx_t dx_0. \tag{10}$$

Regarding the noisy model score $\nabla_{x_t} \log q_\theta(x_t)$, we note that since we can efficiently sample from the one-step student model $q_\theta(x_t)$, we can approximate its score function using another score network $s^{q_\theta}_{\psi_2}(x_t, t) \approx \nabla_{x_t} \log q_\theta(x_t)$, trained with the DSM loss:

$$\min_{\psi_2} \mathcal{L}_{\text{DSM}}(\psi_2) = \frac{1}{2} \iint \| s^{q_\theta}_{\psi_2}(x_t, t) - \nabla_{x_t} \log k_t(x_t|x_0) \|_2^2 q_\theta(x_0) p(x_t|x_0) dx_t dx_0. \tag{11}$$

Thus, the gradient of DiKL with respect to the parameters θ of the student model can be estimated as follows, a method known as Variational Score Distillation (VSD) [37, 54, 31]:

$$\nabla_\theta \text{DiKL}(q_\theta(x_0) || p_d(x_0)) \approx \sum_{t=1}^{T} w(t) \int q_\theta(x_t) \left(s^{q_\theta}_{\psi_2}(x_t, t) - s^{p_d}_{\psi_1}(x_t, t)\right) \frac{\partial x_t}{\partial \theta} dx_t. \tag{12}$$

Unlike the teacher score function which remains fixed after pre-training, the noisy model score $\nabla_{x_t} \log q_\theta(x_t)$ dynamically changes as we update the student model's parameters θ during training. Therefore, the score network $s^{q_\theta}_{\psi_2}(x_t, t) \approx \nabla_{x_t} \log q_\theta(x_t)$ needs to be updated every time we update the student model, which results in an interleaved training procedure as detailed in Algorithm 1.

We observe that training with DiKL typically requires estimating the student model score and using the teacher score for supervision. In the next section, we propose a method that enables training one-step diffusion models without relying on student score estimation or teacher score supervision.

3 Training One-Step Diffusion Models Without Score Distillation

We first explore whether a one-step diffusion model can be trained without teacher supervision (i.e., without relying on pre-trained teacher score, denoiser, ODE solver, etc.). Starting from Algorithm 1, we note that the DiKL gradient estimator relies on the score difference, $s^{q_\theta}_{\psi_1}(x_t, t) - s^{p_d}_{\psi_2}(x_t, t)$. To

Algorithm 2 Score-free Training of One-Step Diffusion Models

Require: Data samples $\{x^{(1)}, \ldots, x^{(N)}\} \sim p_d(x_0)$

Stage 1: Train a multi-step teacher diffusion model

1: Train the teacher model's score network $s^{p_d}_{\psi_1}(x_t, t)$ using DSM (Eq. 10) until convergence

Stage 2: Train a one-step student generative model

2: Initialize the one-step generator with the teacher's score network $g_{\theta_{\text{init}}}(\cdot) \equiv s^{p_d}_{\psi_1}(\cdot, t = t_{\text{init}})$
3: **for** each training iteration **do**
4: Estimate the density ratio $q_\theta(x_t)/p_d(x_t)$ by training a neural network classifier $c_\eta(x_t, t)$
5: Update the one-step generator g_θ with $c_\eta(x_t, t)$ using Eq. 17 or Eq. 18 or Eq. 19
6: **end for**

eliminate the dependency on the teacher's score function $s^{p_d}_{\psi_2}(x_t, t)$, we first observe that the score difference can be written as the gradient of a log-density-ratio:

$$\nabla_{x_t} \log q_\theta(x_t) - \nabla_{x_t} \log p_d(x_t) = \nabla_{x_t} \log \frac{q_\theta(x_t)}{p_d(x_t)}. \tag{13}$$

Therefore, rather than estimating the two scores separately, we can directly estimate the density ratio between the student and teacher models at all noise levels using class-ratio estimation [51, 39, 13, 60].

3.1 Class-Ratio Estimation

We first denote distributions $q_\theta(x_t)$ and $p_d(x_t)$ as two conditional distributions $m(x_t|y=0)$ and $m(x_t|y=1)$, respectively, where $y=0$ indicates samples from the student model $q_\theta(x_t)$ and $y=1$ indicates data samples from $p_d(x_t)$. With Bayes' rule, we can transform the density ratio estimation problem into a binary classification problem:

$$\frac{q_\theta(x_t)}{p_d(x_t)} \equiv \frac{m(x_t|y=0)}{m(x_t|y=1)} = \frac{p(y=0|x_t)\cancel{m(x_t)}}{\cancel{p(y=0)}} \Big/ \frac{p(y=1|x_t)\cancel{m(x_t)}}{\cancel{p(y=1)}} = \frac{p(y=0|x_t)}{p(y=1|x_t)}, \tag{14}$$

where the mixture distribution is defined as

$$m(x) \equiv m(x_t|y=1)p(y=1) + m(x_t|y=0)p(y=0), \tag{15}$$

and the Bernoulli prior distribution $p(y)$ is simply set as a uniform prior $p(y=1) = p(y=0) = 0.5$. In practice, we sample a batch of data from $p_d(x_t)$ and assign them the label $y=0$, and sample an equal number of samples from $q_\theta(x_t)$, assigning them the label $y=1$. We then train a neural network classifier $c_\eta(x_t, t)$, conditioned on the diffusion time t, to estimate the probability that a given input x_t belongs to class $y=1$. The optimal classifier approximates the posterior probability $c^*(x_t, t) = p(y=1 \mid x_t, t)$. In this case, the log-density ratio can be estimated as

$$\nabla_{x_t} \log \frac{q_\theta(x_t)}{p_d(x_t)} \approx \nabla_{x_t} \log \frac{1 - c_\eta(x_t, t)}{c_\eta(x_t, t)} = \nabla_{x_t} \operatorname{logit}(1 - c_\eta(x_t, t)). \tag{16}$$

Estimating the density ratio in the noisy space has the advantage of increasing the overlap between the supports of the two distributions, thereby stabilizing the training process.

Importantly, our method does not require any forms of teacher score supervision, as it avoids the need of using the teacher score $s^{q_\theta}_{\psi_1}(x_t, t)$ to approximate the noisy data score $\nabla_{x_t} \log p_d(x_t)$. Furthermore, compared to VSD, our approach employs a single class-ratio estimator, which is more memory-efficient and consistent than the two independently trained score networks $s^{q_\theta}_{\psi_1}$ and $s^{p_d}_{\psi_2}$ used in the original VSD loss (Equation 12).

3.2 Class-Ratio Gradient Estimators for Training One-Step Diffusion Models

We can already obtain a new gradient estimator for DiKL by plugging in our class-ratio estimator to Equation 9:

$$\nabla_\theta \text{DiKL}(q_\theta(x_0) || p_d(x_0)) \approx \sum_{t=1}^{T} w(t) \int q_\theta(x_t) \nabla_{x_t} \operatorname{logit}(1 - c_\eta(x_t, t)) \frac{\partial x_t}{\partial \theta} dx_t. \tag{17}$$

In addition to the DiKL, we can use the learned classifier function c_η to obtain a family of gradient estimators for alternative training objectives. For instance, replacing the logit function with the logarithm function yields an objective that minimizes the probability of generated samples being classified as fake. This formulation aligns with GAN [11, 35] across different diffusion time steps, which is equivalent to minimizing the *Diffusive Jensen-Shannon (DiJS)* divergence:

$$\nabla_\theta \mathrm{DiJS}(q_\theta(x_0)||p_d(x_0)) \approx \sum_{t=1}^{T} w(t) \int q_\theta(x_t) \nabla_{x_t} \log(1 - c_\eta(x_t, t)) \frac{\partial x_t}{\partial \theta} dx_t. \tag{18}$$

Alternatively, rather than minimizing the probability of the generated images being fake as used in GAN, one can also maximize the probability of them being real. This approach is referred to as *Diffusive Realism Maximization (DiRM)*, which has the following gradient estimator:

$$\nabla_\theta \mathrm{DiRM}(\theta) \approx -\sum_{t=1}^{T} w(t) \int q_\theta(x_t) \nabla_{x_t} \log c_\eta(x_t, t) \frac{\partial x_t}{\partial \theta} dx_t. \tag{19}$$

The overall training procedure of our proposed framework is summarized in Algorithm 2. Notably, the DiRM objective—maximizing the likelihood of being real—also mirrors the *non-saturating GAN* formulation [11], which is known to provide more stable gradients for the generator compared to the original minimax objective. In principle, once the density ratio is available, any ratio-based divergence measure—such as an f-divergence—can be employed to formulate a learning criterion for diffusion distillation. This is reminiscent of the f-GAN framework [35]: we will discuss the connection between our proposed method and GAN-based approaches in the next section.

4 Related Work

4.1 Comparison with GANs

Estimating density ratios is central to many GAN variants [11, 35]. In classic GANs, the discriminator implicitly estimates the density ratio between real and model distributions. However, for high-dimensional image modeling tasks, both data and model distributions are supported on low-dimensional manifolds and are therefore not absolutely continuous, rendering their densities and the density ratio ill-defined. This further causes the Jensen-Shannon (JS) divergence to be ill-defined and contributes to GAN training instability [1, 2, 32, 42].

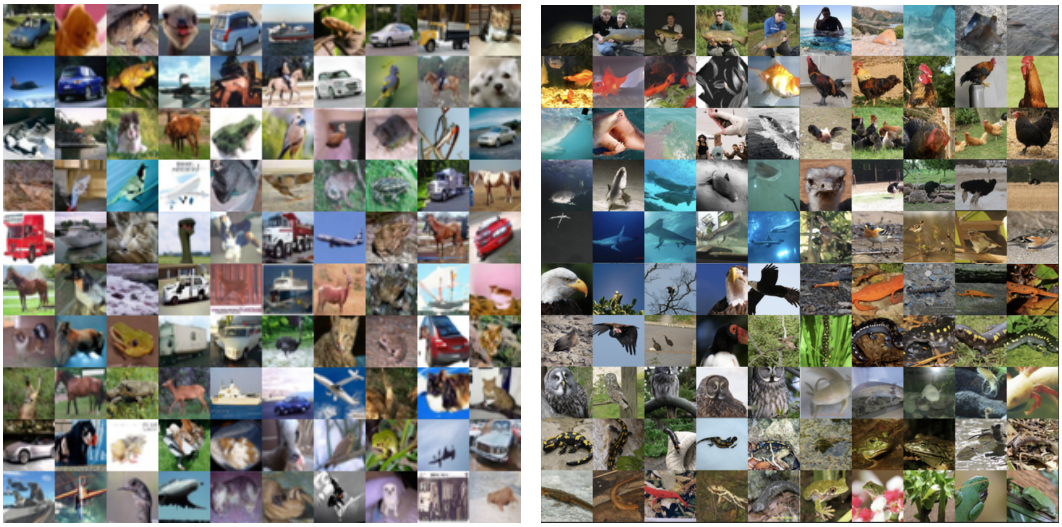
Alternative Divergence. To address this, Arjovsky et al. [2] proposed to replace the JS divergence with the Wasserstein-1 distance, which can yield meaningful gradients even when the distributions are disjoint. However, training Wasserstein GANs requires enforcing a 1-Lipschitz constraint on the critic, which is challenging in practice and has been approximated using heuristics such as weight clipping [2], gradient penalties [12], and spectral normalization [33]. Despite its theoretical appeal, the divergence minimized in practice often differs from the idealized objective [32], and that stable GAN training relies more on regularization (e.g., gradient penalties) than on strict divergence minimization [9].

Noise as Regularization. Adding Gaussian noise to real and fake samples has been proposed as a way to ensure distributions are fully supported, making the density ratio well-defined [46, 42, 35, 59]. This model-agnostic approach requires no architectural changes but hinges on choosing an effective noise level—something hard to fix throughout training.

4.2 Comparison with Diffusion GAN

Diffusion GAN [53] addresses the challenge of selecting a fixed noise level by introducing a diffusion-inspired noise schedule that gradually increases noise in tandem with the model's learning capacity. While this represents the most closely related work to ours, our approach differs in several important aspects, as outlined below.

Generator Architecture. Diffusion GAN stabilizes training using a StyleGAN-based generator [22], which is implicitly trained progressively from low to high resolutions while keeping the network topology fixed. In contrast, our method does not rely on a specialized generator architecture or progressive resolution training. Instead, we adopt a generic U-Net architecture, resulting in a simpler and more broadly applicable framework.



(a) CIFAR-10 (b) ImageNet 64×64

Figure 2: Visualization of sample images generated by DiJS.

Regularization Techniques. We do not use common GAN-specific training tricks such as gradient penalties or spectral normalization. Our method provides a clean framework for training one-step diffusion models yet still achieves stable convergence, demonstrating robustness without any additional regularization tricks during training.

Adversarial Nature. Unlike traditional GANs, our framework is no longer adversarial in the strict sense: our generator's performance is *not* dependent on the convergence of a discriminator, which simplifies the training process and mitigates common issues arising from adversarial dynamics (e.g., GANs require a careful balancing between discriminator and generator training).

Because our method does not rely on discriminators, gradient penalties, or adversarial training, we are estimating and minimizing the actual DiJS divergence—a divergence that remains well-defined even when traditional density ratios remain undefined. This leads to a conceptually cleaner training objective that is more amenable to theoretical analysis and empirical diagnosis, as it avoids the complexities and instabilities associated with adversarial min-max optimization and eliminates the need for additional regularization tricks.

In the next section, we will demonstrate the effectiveness of our method by applying it to train one-step diffusion models for image generation.

5 Experiments and Analysis

5.1 Image Generation Experiments

We evaluate the performance of our method by training one-step image generative models on two standard benchmarks: unconditional generation on the CIFAR-10 (32×32) dataset [24] and class-label-conditioned generation on the ImageNet (64×64) dataset [7]. Our implementation builds upon the EDM codebase [20] and uses the official pre-trained EDM models as teacher models for both datasets. All experiments are conducted on a node of 4 NVIDIA H100 80GB GPUs.

Our one-step student model adopts the same neural network architecture as the teacher model. We use the variance-exploding (VE) noise schedule to define the DiKL divergence for training the student model. Our class-ratio estimator $c_\eta(x_t, t)$ is implemented using the encoder portion of a U-Net to produce a scalar output. This network is approximately half the size of the full U-Net used for score estimation, leading to improved training and memory efficiency. The one-step generator $g_\theta(z)$ is initialized with pre-trained EDM weights from the teacher model with diffusion time $t_{\text{init}} = 2.5$ fixed throughout training and sampling. We follow standard hyperparameter settings for training generative models on CIFAR-10 and ImageNet. Specifically, we set the learning rate to 10^{-5}, use the weight

Table 1: Unconditional generation performance of one-step models on CIFAR-10.

METHOD	NFE (↓)	FID (↓)	IS (↑)
Teacher model			
EDM [20]	35	2.04	9.84
EDM [20]	1	8.70	8.49
Training from scratch (w/o teacher)			
CT [49]	1	8.70	8.49
iCT [48]	1	2.83	9.54
iCT-deep [48]	1	**2.51**	**9.76**
BCM [25]	1	3.10	9.45
BCM-deep [25]	1	2.64	9.67
sCT [29]	1	2.85	-
Diffusion-GAN [53]	1	3.19	-
IMM [62]	1	3.20	-
Distillation w/ teacher init. & supervision			
Progressive Distillation [43]	1	8.34	8.69
DSNO [61]	1	3.78	-
TRACT [5]	1	3.78	-
CD [49]	1	3.55	9.48
CTM (w/ GAN) [23]	1	1.98	-
CTM (w/o GAN) [23]	1	>5.0	-
sCD [29]	1	3.66	-
Diff-Instruct [31]	1	4.53	-
ECT [10]	1	3.60	-
SiD [64]	1	2.03	10.02
SiDA [63]	1	**1.52**	**10.32**
Distillation w/ teacher initialization only			
DiRM (ours)	1	4.87	9.85
DiKL (ours)	1	3.81	9.90
DiJS (ours)	1	**2.39**	**9.93**

Table 2: Class-label-conditioned generation performance of one-step generative models on ImageNet 64×64.

METHOD	NFE (↓)	FID (↓)
Teacher model		
EDM [20]	79	2.44
Training from scratch (w/o teacher)		
EDM2-L/XL [22]	1	13.0
CT [49]	1	13.0
iCT [48]	1	4.02
iCT-deep [48]	1	3.25
BCM [25]	1	4.18
BCM-deep [25]	1	3.14
sCT [29]	1	**2.04**
Distillation w/ teacher init. & supervision		
Progressive Distillation [43]	1	7.88
DSNO [61]	1	7.83
TRACT [5]	1	7.43
CD [49]	1	6.20
CTM (w/ GAN) [23]	1	1.92
sCD [29]	1	2.44
Diff-Instruct [31]	1	5.57
EM Distillation [56]	1	2.20
ECT [10]	1	2.49
SiD [64]	1	2.02
SiDA [63]	1	**1.35**
Distillation w/ teacher initialization only		
DiJS (ours)	1	**1.54**

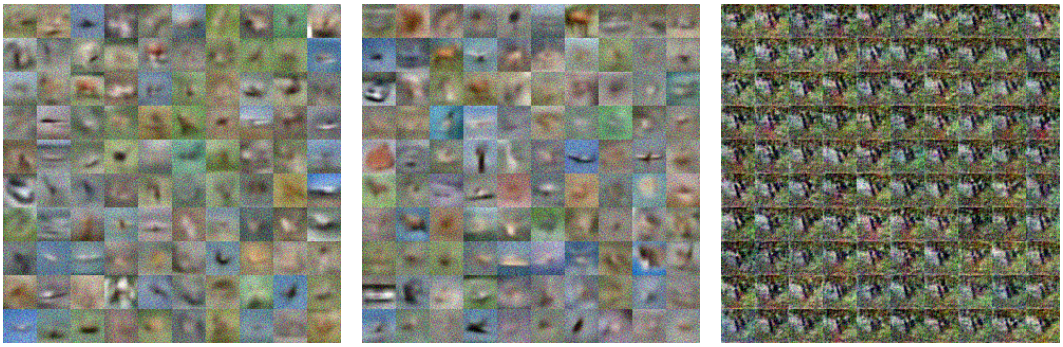
function $w(t) = \sigma_t^2$, and employ non-leaky data augmentation [21] for both datasets. For CIFAR-10, we use a batch size of 64 and an EMA decay rate of 0.5. For ImageNet, all training configurations, including the batch size and EMA decay rate, are set to be the same as those described in [20].

Interestingly, we observe that performing multiple gradient updates for the class-ratio estimator in each training iteration can accelerate convergence (i.e., reducing the number of training iterations for the student model) without destabilizing the training process, which is distinct from GANs where a careful balancing between the training steps for the discriminator and generator is required for stable training. However, such strategies often increase the overall wall-clock time for training. Hence, we adopt a single-step update strategy for the class-ratio estimator throughout our experiments, consistent with previous works [31, 64]; we leave multi-step class-ratio estimation for future work.

In Tables 1 and 2, we compare our method to previous methods for training one-step generative models. To highlight methodological differences, we categorize these approaches into three groups: (1) training from scratch (e.g., Diffusion-GAN, trajectory-based distillation), (2) training with teacher initialization and supervision (e.g., using teacher score, denoiser or ODE solvers in the loss), and (3) training with teacher weight initialization only (our method). We find that our proposed method, DiJS, achieves competitive one-step generation performance despite not using any teacher score information, outperforming most state-of-the-art distillation methods that rely on full teacher supervision. The only method that outperforms ours is SiDA [63], which depends on training data, teacher score supervision, teacher weight initialization, and student score estimation. In contrast, our method requires only training data, teacher weight initialization, and class-ratio estimation. Notably, this eliminates the need for teacher supervision in the training process. Also, class-ratio estimation is both simpler and more lightweight than student score estimation, as the ratio network is approximately half the size of a full score network. This results in a more streamlined and memory-efficient training framework.

5.2 Analysis of the Role of Teacher Weight Initialization

In the previous section, our one-step student model was initialized with the teacher model's weights. We observed that training from random initialization led to mode collapse; see Figure 3c for an example of mode collapse. One possible explanation is that mode collapse arises from the training



(a) Single-level DSM Init. (b) Multi-level DSM Init. (c) Collapsed Samples

Figure 3: Visualization of different initializations and collapsed samples on CIFAR-10.

objectives (i.e., reverse KL or JS divergence), a phenomenon also observed in GAN literature [11]. To understand why initializing the student model with the teacher model's weights prevents mode collapse in the training process, we investigate the following two hypotheses.

Function Space Hypothesis. *Teacher weight initialization provides a more structured latent-to-output functional mapping, i.e., different locations in the latent space are initially mapped to distinct images, preventing mode collapse.*

This hypothesis originally arose from visualizing initialized samples as shown in Figure 3a, showing that initialization already induces diverse mappings, with the student model training stage primarily refining these initializations into sharper images. Somewhat surprisingly, however, we find that functional initialization alone is insufficient to prevent mode collapse. To show this, instead of training the teacher model across different diffusion time steps t and selecting a single time step t_{init} for initialization, we only pre-train the teacher model at the selected time step t_{init} and use its weight to initialize the one-step student model. This setup ensures identical latent-to-output mappings for the student model at initialization as shown in Figure 3b. However, with this initialization, the student model still exhibits mode collapse early in the student model training stage, which suggests that the functional mapping perspective alone does not fully explain the mode-collapse issue.

Feature Space Hypothesis. *Teacher weight initialization provides a rich set of multi-level features learned when pre-training the teacher diffusion model, which help prevent mode collapse.*

To verify this hypothesis and isolate the role of learned features from functional mapping effects, we pre-train the teacher model on CIFAR-100 while excluding all classes that overlap with CIFAR-10. This ensures that images from the target classes that the student model aims to generate are absent during pre-training, allowing us to focus solely on the contribution of the learned features. We train the teacher model using increasingly larger subsets of CIFAR-100 with (10, 50, 90) classes, creating a setting with increasing feature diversity. Table 3 shows the performance of our one-step student model on CIFAR-10 initialized with the weights of teacher models trained on varying numbers of CIFAR-100 classes. We find that when the teacher model is trained on only 10 classes, mode collapse still occurs. However, as the number of training classes increases, the student model no longer collapses, indicating that feature richness plays a crucial role in preventing mode collapse. Nevertheless, despite mitigating mode collapse, this initialization strategy achieves an FID of 6.01 when the teacher model is pre-trained on all 90 non-overlapping classes in CIFAR-100, which is significantly worse than the FID (2.39) obtained when directly using CIFAR-10 as the pre-training dataset. This suggests that

Table 3: Performance of one-step models trained by DiJS with different initializations on various CIFAR subsets.

Initialization method	Initialization dataset	FID
No initialization	-	collapsed
Single-level DSM	full CIFAR-10	collapsed
Multi-level DSM	10 classes in CIFAR-100	collapsed
	50 classes in CIFAR-100	6.20
	90 classes in CIFAR-100	6.01
	full CIFAR-10	2.39

while feature richness is essential for stabilizing training, functional mapping initialization remains important for achieving higher sample quality.

6 Conclusions

This work investigated whether one-step diffusion models could be trained without pre-trained teacher diffusion model. We introduced a family of score-free training methods based on class-ratio estimation, removing the need for both teacher and student score supervision. Our DiJS method achieved competitive generation quality—comparable to state-of-the-art methods with full teacher supervision—while reducing training complexity and memory usage. Notably, our method also simplifies GAN-style training by eliminating the need for adversarial optimization tricks such as gradient penalties, spectral normalization, and special time schedule. This shows that a single time-conditioned class-ratio estimator suffices for stable and effective training.

A key insight from our analysis was that teacher score supervision could be removed, but weight initialization from the teacher remained essential. We found that this was not due to improved latent-to-output mappings, but rather the multi-level features learned across noise levels during diffusion training, which helped prevent mode collapse. Future work may explore unsupervised or self-supervised pretraining to provide feature-rich initializations without requiring a pre-trained diffusion model. Also, extending our framework to other modalities such as audio and video also remains a promising direction.

Acknowledgments and Disclosure of Funding

MZ and DB acknowledge funding from AI Hub in Generative Models, under grant EP/Y028805/1. JH is supported by the University of Cambridge Harding Distinguished Postgraduate Scholars Programme. ZO is supported by the Lee Family Scholarship. JMHL acknowledges support from a Turing AI Fellowship under grant EP/V023756/1.

References

[1] Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*, 2017.

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[3] Fan Bao, Chongxuan Li, Jiacheng Sun, Jun Zhu, and Bo Zhang. Estimating the optimal covariance with imperfect mean in diffusion probabilistic models. *arXiv preprint arXiv:2206.07309*, 2022.

[4] Fan Bao, Chongxuan Li, Jun Zhu, and Bo Zhang. Analytic-dpm: an analytic estimate of the optimal reverse variance in diffusion probabilistic models. *arXiv preprint arXiv:2201.06503*, 2022.

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

[6] Valentin De Bortoli, Alexandre Galashov, J. Swaroop Guntupalli, Guangyao Zhou, Kevin Murphy, Arthur Gretton, and Arnaud Doucet. Distributional diffusion models with scoring rules, 2025.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. IEEE, 2009.

[8] Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011.

[9] William Fedus, Mihaela Rosca, Balaji Lakshminarayanan, Andrew M Dai, Shakir Mohamed, and Ian Goodfellow. Many paths to equilibrium: Gans do not need to decrease a divergence at every step. *arXiv preprint arXiv:1710.08446*, 2017.

[10] Zhengyang Geng, Ashwini Pokle, Weijian Luo, Justin Lin, and J Zico Kolter. Consistency models made easy. In *The Thirteenth International Conference on Learning Representations*, 2025.

[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[12] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

[13] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304, 2010.

[14] Jiajun He, Wenlin Chen, Mingtian Zhang, David Barber, and José Miguel Hernández-Lobato. Training neural samplers with reverse diffusive kl divergence. *arXiv preprint arXiv:2410.12456*, 2024.

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

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[17] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in Neural Information Processing Systems*, 35:8633–8646, 2022.

[18] Emiel Hoogeboom, Vıctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In *International conference on machine learning*, pages 8867–8887. PMLR, 2022.

[19] Ferenc Huszár. Variational inference using implicit distributions. *arXiv preprint arXiv:1702.08235*, 2017.

[20] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in Neural Information Processing Systems*, 35:26565–26577, 2022.

[21] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *Advances in neural information processing systems*, 33:12104–12114, 2020.

[22] Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. Analyzing and improving the training dynamics of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24174–24184, 2024.

[23] 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 *The Twelfth International Conference on Learning Representations*, 2024.

[24] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[25] Liangchen Li and Jiajun He. Bidirectional consistency models. *arXiv preprint arXiv:2403.18035*, 2024.

[26] Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35:4328–4343, 2022.

[27] Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D Plumbley. Audioldm: Text-to-audio generation with latent diffusion models. *arXiv preprint arXiv:2301.12503*, 2023.

[28] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. *arXiv preprint arXiv:2202.09778*, 2022.

[29] Cheng Lu and Yang Song. Simplifying, stabilizing and scaling continuous-time consistency models. In *International Conference on Learning Representations*, 2025.

[30] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022.

[31] 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. *Advances in Neural Information Processing Systems*, 36, 2024.

[32] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *International conference on machine learning*, pages 3481–3490. PMLR, 2018.

[33] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[34] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

[35] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. *Advances in neural information processing systems*, 29, 2016.

[36] Zijing Ou, Mingtian Zhang, Andi Zhang, Tim Z Xiao, Yingzhen Li, and David Barber. Improving probabilistic diffusion models with optimal covariance matching. *International Conference on Learning Representations*, 2025.

[37] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

[38] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *The Eleventh International Conference on Learning Representations*, 2022.

[39] Jing Qin. Inferences for case-control and semiparametric two-sample density ratio models. *Biometrika*, 85(3):619–630, 1998.

[40] Herbert E Robbins. An empirical bayes approach to statistics. In *Breakthroughs in Statistics: Foundations and basic theory*, pages 388–394. Springer, 1992.

[41] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[42] Kevin Roth, Aurelien Lucchi, Sebastian Nowozin, and Thomas Hofmann. Stabilizing training of generative adversarial networks through regularization. *Advances in neural information processing systems*, 30, 2017.

[43] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022.

[44] Tim Salimans, Thomas Mensink, Jonathan Heek, and Emiel Hoogeboom. Multistep distillation of diffusion models via moment matching. *arXiv preprint arXiv:2406.04103*, 2024.

[45] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[46] Casper Kaae Sønderby, Jose Caballero, Lucas Theis, Wenzhe Shi, and Ferenc Huszár. Amortised map inference for image super-resolution. *arXiv preprint arXiv:1610.04490*, 2016.

[47] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[48] Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. In *The Twelfth International Conference on Learning Representations*, 2024.

[49] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

[50] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

[51] Masashi Sugiyama, Taiji Suzuki, and Takafumi Kanamori. *Density ratio estimation in machine learning*. Cambridge University Press, 2012.

[52] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

[53] Zhendong Wang, Huangjie Zheng, Pengcheng He, Weizhu Chen, and Mingyuan Zhou. Diffusion-GAN: Training GANs with diffusion. In *The Eleventh International Conference on Learning Representations*, 2023.

[54] 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. *Advances in Neural Information Processing Systems*, 36, 2024.

[55] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion gans. *arXiv preprint arXiv:2112.07804*, 2021.

[56] Sirui Xie, Zhisheng Xiao, Diederik P Kingma, Tingbo Hou, Ying Nian Wu, Kevin Patrick Murphy, Tim Salimans, Ben Poole, and Ruiqi Gao. Em distillation for one-step diffusion models. *arXiv preprint arXiv:2405.16852*, 2024.

[57] Longlin Yu, Tianyu Xie, Yu Zhu, Tong Yang, Xiangyu Zhang, and Cheng Zhang. Hierarchical semi-implicit variational inference with application to diffusion model acceleration. *Advances in Neural Information Processing Systems*, 36, 2024.

[58] Mingtian Zhang, Thomas Bird, Raza Habib, Tianlin Xu, and David Barber. Variational f-divergence minimization. *arXiv preprint arXiv:1907.11891*, 2019.

[59] Mingtian Zhang, Peter Hayes, Thomas Bird, Raza Habib, and David Barber. Spread divergence. In *International Conference on Machine Learning*, pages 11106–11116. PMLR, 2020.

[60] Mingtian Zhang, Andi Zhang, Tim Z Xiao, Yitong Sun, and Steven McDonagh. Out-of-distribution detection with class ratio estimation. *arXiv preprint arXiv:2206.03955*, 2022.

[61] Hongkai Zheng, Weili Nie, Arash Vahdat, Kamyar Azizzadenesheli, and Anima Anandkumar. Fast sampling of diffusion models via operator learning. In *International conference on machine learning*, pages 42390–42402. PMLR, 2023.

[62] Linqi Zhou, Stefano Ermon, and Jiaming Song. Inductive moment matching. *arXiv preprint arXiv:2503.07565*, 2025.

[63] Mingyuan Zhou, Huangjie Zheng, Yi Gu, Zhendong Wang, and Hai Huang. Adversarial score identity distillation: Rapidly surpassing the teacher in one step. In *The Thirteenth International Conference on Learning Representations*, 2025.

[64] 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 *Forty-first International Conference on Machine Learning*, 2024.

A Validity of Diffusive Divergence

We follow the original Spread Divergence [59] and provide a simple proof of the validity of the diffusive KL (DiKL) divergence. The extension to the diffusive Jensen-Shannon (DiJS) divergence is straightforward. See [59] for a generalized proof that includes cases with non-absolutely continuous distributions and non-Gaussian kernels.

Our goal is to show that for the DiKL defined as

$$\mathrm{DiKL}_{\mathcal{K}}(q_\theta(x_0) \,\|\, p_d(x_0)) = \sum_{t=1}^{T} w(t)\, \mathrm{KL}(q_\theta(x_t) \,\|\, p_d(x_t)), \tag{20}$$

where $w(t) > 0$, and the densities $q_\theta(x_t)$ and $p_d(x_t)$ represent the noisy model and data distributions respectively, defined as:

$$q_\theta(x_t) = \int q_\theta(x_0)\, k_t(x_t \mid x_0)\, dx_0, \tag{21}$$

$$p_d(x_t) = \int p_d(x_0)\, k_t(x_t \mid x_0)\, dx_0, \tag{22}$$

with $k_t(x_t \mid x_0)$ denoting the transition kernel (e.g., Gaussian noise). Then:

$$\mathrm{DiKL}_{\mathcal{K}}(q_\theta(x_0) \,\|\, p_d(x_0)) = 0 \iff q_\theta(x_0) = p_d(x_0).$$

Since $w(t) > 0$ and the KL divergence is non-negative, it suffices to show that:

$$\mathrm{KL}(q_\theta(x_t) \,\|\, p_d(x_t)) = 0 \iff q_\theta(x_t) = p_d(x_t) \iff q_\theta(x_0) = p_d(x_0).$$

To demonstrate this, assume that $k_t(\epsilon) = \mathcal{N}(0, \sigma^2 I)$, and rewrite the noisy densities as convolutions:

$$q_\theta(x_t) = (q_\theta * k_t)(x_t), \tag{23}$$

$$p_d(x_t) = (p_d * k_t)(x_t). \tag{24}$$

Suppose $q_\theta(x_t) = p_d(x_t)$. Applying the Fourier transform $\mathcal{F}$, we obtain:

$$\mathcal{F}(q_\theta * k_t) = \mathcal{F}(q_\theta) \cdot \mathcal{F}(k_t), \tag{25}$$

$$\mathcal{F}(p_d * k_t) = \mathcal{F}(p_d) \cdot \mathcal{F}(k_t). \tag{26}$$

Given $q_\theta(x_t) = p_d(x_t)$, we have:

$$q_\theta(x_t) = p_d(x_t) \iff \mathcal{F}(q_\theta) \cdot \cancel{\mathcal{F}(k_t)} = \mathcal{F}(p_d) \cdot \cancel{\mathcal{F}(k_t)} \iff \mathcal{F}(q_\theta) = \mathcal{F}(p_d) \iff q_\theta = p_d.$$

Therefore, $\mathrm{KL}(q_\theta(x_t) \,\|\, p_d(x_t)) = 0 \iff q_\theta(x_t) = p_d(x_t)$, and thus:

$$\mathrm{DiKL}_{\mathcal{K}}(q_\theta(x_0) \,\|\, p_d(x_0)) = 0 \iff q_\theta(x_0) = p_d(x_0).$$

B Derivation of Analytical Gradient for Reverse KL

The gradient of reverse DiKL w.r.t. the model parameter θ is given by

$$\nabla_\theta \mathrm{DiKL}_{k_t}(q_\theta || p_d) = \int q_\theta(x_t) \left(\nabla_{x_t} \log q_\theta(x_t) - \nabla_{x_t} \log p_d(x_t)\right) \frac{\partial x_t}{\partial \theta} dx_t. \tag{27}$$

The reverse DiKL at time t is defined as

$$\mathrm{DiKL}_{k_t}(q_\theta || p_d) = \int \left(\log q_\theta(x_t) - \log p_d(x_t)\right) q_\theta(x_t) dx_t. \tag{28}$$

We first reparameterize x_t as a function of z and ϵ:

$$x_t = \alpha_t g_\theta(z) + \sigma_t \epsilon_t \equiv h_\theta(z, \epsilon_t), \tag{29}$$

where $z \sim p(z) \equiv \mathcal{N}(z|0, I)$ and $\epsilon_t \sim p(\epsilon_t) \equiv \mathcal{N}(\epsilon_t|0, I)$. It then follows that

$$\begin{aligned}
\nabla_\theta \mathrm{DiKL}_{k_t}(q_\theta||p_d) &= \nabla_\theta \int (\log q_\theta(x_t) - \log p_d(x_t))\, q_\theta(x_t) dx_t \\
&= \nabla_\theta \iiint (\log q_\theta(x_t) - \log p_d(x_t))\, \delta(x_t - h_\theta(z, \epsilon_t)) p(z) p(\epsilon_t) dx_t dz d\epsilon \\
&= \nabla_\theta \iint (\log q_\theta(x_t) - \log p_d(x_t))\, |_{x_t = h_\theta(z,\epsilon_t)} p(z) p(\epsilon_t) dz d\epsilon \\
&= \int \left(\nabla_\theta \log q_\theta(x_t) + \nabla_{x_t} \log q_\theta(x_t) \frac{\partial x_t}{\partial \theta} - \nabla_{x_t} \log p_d(x_t) \frac{\partial x_t}{\partial \theta} \right) p_\theta(x_t) dx_t \\
&= \int \left(\nabla_{x_t} \log q_\theta(x_t) \frac{\partial x_t}{\partial \theta} - \nabla_{x_t} \log p_d(x_t) \frac{\partial x_t}{\partial \theta} \right) p_\theta(x_t) dx_t,
\end{aligned}$$

where the last line follows since

$$\int \nabla_\theta \log q_\theta(x_t) q_\theta(x_t) dx_t = \int \nabla_\theta q_\theta(x_t) dx_t = \nabla_\theta \int q_\theta(x_t) dx_t = \nabla_\theta 1 = 0. \tag{30}$$

This completes the proof.

C Class Raitio Estimation For DiJS Divergence

We define the diffusive Jensen-Shannon (DiJS) divergence between the model distribution $q_\theta(x_0)$ and the data distribution $p_d(x_0)$ as:

$$\mathrm{DiJS}_{\mathcal{K}}(q_\theta(x_0) \,\|\, p_d(x_0)) = \sum_{t=1}^{T} w(t)\, \mathrm{JS}(q_\theta(x_t) \,\|\, p_d(x_t)), \tag{31}$$

where $w(t) > 0$ are positive weights, and the noisy distributions $q_\theta(x_t)$ and $p_d(x_t)$ are defined via convolution with a transition kernel $k_t(x_t \mid x_0)$ (e.g., Gaussian noise):

$$q_\theta(x_t) = \int q_\theta(x_0)\, k_t(x_t \mid x_0)\, dx_0, \tag{32}$$

$$p_d(x_t) = \int p_d(x_0)\, k_t(x_t \mid x_0)\, dx_0. \tag{33}$$

The Jensen-Shannon divergence between two distributions q and p is given by:

$$\mathrm{DiJS}(q_\theta \,\|\, p_d) = \frac{1}{2}\mathrm{DiKL}(q_\theta \,\|\, \frac{1}{2}(q_\theta + p_d)) + \frac{1}{2}\mathrm{DiKL}(p_d \,\|\, \frac{1}{2}(q_\theta + p_d)). \tag{34}$$

We now derive the gradient of the DiJS divergence with respect to model parameters θ. By the chain rule:

$$\nabla_\theta \mathrm{DiJS}(q_\theta(x_0) \,\|\, p_d(x_0)) = \sum_{t=1}^{T} w(t)\, \nabla_\theta \mathrm{JS}(q_\theta(x_t) \,\|\, p_d(x_t)). \tag{35}$$

Assume we obtain the optimal classifier $c^*(x_t, t) \equiv p(y = 1 \mid x_t, t)$. We follow the GAN method [11] to ignore the second term when the class ratio estimation is optimal, the first term in the JS divergence (involving q_θ) gives:

$$\nabla_\theta \mathrm{JS}(q_\theta(x_t) \,\|\, p_d(x_t)) \approx \int \nabla_\theta q_\theta(x_t) \log \left(\frac{q_\theta(x_t)}{m(x_t)} \right) dx_t \tag{36}$$

$$= \int q_\theta(x_t) \nabla_{x_t} \log \left(\frac{q_\theta(x_t)}{m(x_t)} \right) \frac{\partial x_t}{\partial \theta} dx_t, \tag{37}$$

where $m(x_t) = \frac{1}{2}(q_\theta(x_t) + p_d(x_t))$. Using the class-ratio view, we substitute:

$$\frac{q_\theta(x_t)}{m(x_t)} = \frac{m(x_t|y=0)}{m(x_t|y=0) + m(x_t|y=1)} = 1 - c^*(x_t, t),$$

which gives:

$$\nabla_\theta \mathrm{DiJS}(q_\theta(x_0) \,\|\, p_d(x_0)) \approx \sum_{t=1}^{T} w(t) \int q_\theta(x_t) \nabla_{x_t} \log(1 - c^*(x_t, t)) \frac{\partial x_t}{\partial \theta} dx_t. \tag{38}$$

If we don't follow the GAN [11] approximation, one can also treat this gradient estimation as an exact estimation of a DiKL with a mixture target distribution

$$\nabla_\theta \mathrm{DiKL}\left(q_\theta || \frac{1}{2}(q_\theta + p_d)\right) = \sum_{t=1}^{T} w(t) \int q_\theta(x_t) \nabla_{x_t} \log(1 - c^*(x_t, t)) \frac{\partial x_t}{\partial \theta} dx_t.$$

This is also a valid divergence between q_θ and p_d since $q_\theta = \frac{1}{2} q_\theta + \frac{1}{2} p_d \iff q_\theta = p_d$.

D Additional Image Generation Results

Figure 4: Visualization of the samples from the multi-level DSM Initialization

Figure 5: Visualization of the samples from the single-level DSM Initialization

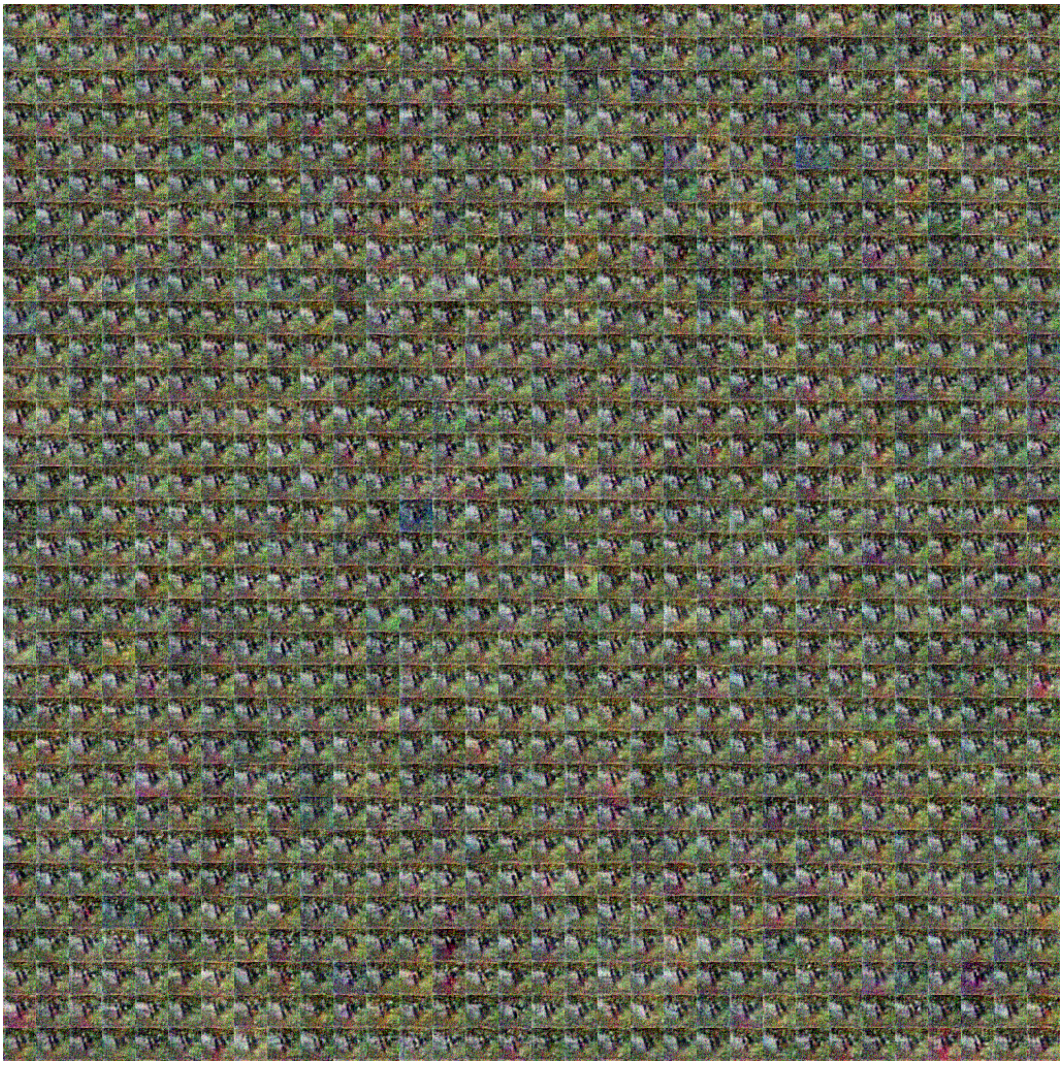


Figure 6: Visualization of the collapsed samples

Figure 7: CIFAR Visualization of the DiJS samples (FID=2.39, IS=9.93)

Figure 8: ImageNet 64x64 Visualization of the DiJS samples (FID=1.54)