Score identity Distillation: Exponentially Fast Distillation of Pretrained Diffusion Models for One-Step Generation

Mingyuan Zhou [1 2] **Huangjie Zheng** [1] **Zhendong Wang** [1] **Mingzhang Yin** [3] **Hai Huang** [2]

Abstract

We introduce Score identity Distillation (SiD), an innovative data-free method that distills the generative capabilities of pretrained diffusion models into a single-step generator. SiD not only facilitates an exponentially fast reduction in Fréchet inception distance (FID) during distillation but also approaches or even exceeds the FID performance of the original teacher diffusion models. By reformulating forward diffusion processes as semi-implicit distributions, we leverage three score-related identities to create an innovative loss mechanism. This mechanism achieves rapid FID reduction by training the generator using its own synthesized images, eliminating the need for real data or reverse-diffusion-based generation, all accomplished within significantly shortened generation time. Upon evaluation across four benchmark datasets, the SiD algorithm demonstrates high iteration efficiency during distillation and surpasses competing distillation approaches, whether they are one-step or few-step, data-free, or dependent on training data, in terms of generation quality. This achievement not only redefines the benchmarks for efficiency and effectiveness in diffusion distillation but also in the broader field of diffusion-based generation. The PyTorch implementation is available at https://github.com/mingyuanzhou/SiD.

1. Introduction

Diffusion models, also known as score-based generative models, have emerged as the leading approach for generating high-dimensional data (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020). These models are appreciated for their training simplicity and stability, their robustness against mode collapse during generation, and their ability to produce high-resolution, diverse, and photo-realistic images (Dhariwal & Nichol, 2021; Ho et al., 2022; Ramesh et al., 2022; Rombach et al., 2022; Saharia et al., 2022; Peebles & Xie, 2023; Zheng et al., 2023c).

However, the process of generating data with diffusion models involves iterative refinement-based reverse diffusion, necessitating multiple iterations through the same generative network. This multi-step generation process, initially requiring hundreds or even thousands of steps, stands in contrast to the single-step generation capabilities of previous deep generative models such as variational auto encoders (VAEs) (Kingma & Welling, 2014; Rezende et al., 2014) and generative adversarial networks (GANs) (Goodfellow et al., 2014; Karras et al., 2020), which only require forwarding the noise through the generation network once. Diffusion models necessitate multi-step generation, making them much more expensive at inference time. A wide variety of methods have been introduced to reduce the number of sampling steps during reverse diffusion, but they often still require quite a few number of function evaluations (NFE), such as 35 NFE for CIFAR-10 32x32 and 511 NFE for ImageNet 64x64 in EDM (Karras et al., 2022), to achieve good performance.

In this study, we aim to introduce a single-step generator designed to distill the knowledge on training data embedded in the score-estimation network of a pretrained diffusion model. To achieve this goal, we propose training the generator by minimizing a model-based score-matching loss between the scores of the diffused real data and the diffused generator-synthesized fake data distributions at various noise levels. However, estimating this model-based score-matching loss, which is a form of Fisher divergence, at any given noise level proves to be intractable. To overcome this challenge, we offer a fresh perspective by viewing the forward diffusion processes of diffusion models through the lens of semi-implicit distributions. We introduce three corresponding score-related identities and illustrate their integration to formulate an innovative loss mechanism. This mechanism involves both score estimation and Monte Carlo estimation techniques to handle intractable expectations. Our method,

[1]The University of Texas at Austin [2]Google [3]The University of Florida. Correspondence to: Mingyuan Zhou <mingyuan.zhou@mccombs.utexas.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

Figure 1. Rapid advancements in the distillation of a pretrained ImageNet 64x64 diffusion model are shown using the proposed SiD method, with settings $\alpha = 1.0$, a batch size of 1024, and a learning rate of 5e-6. The series of images, generated from the same set of random noises post-training the SiD generator with varying counts of synthesized images, illustrates progressions at 0, 0.1, 0.2, 0.5, 1, 2, 5, 10, 20, and 50 million images. These are equivalent to roughly 0, 100, 200, 500, 1K, 2K, 5K, 10K, 20K, and 49K training iterations respectively, organized from the top left to the bottom right. The associated FIDs for these iterations are 153.52, 34.83, 37.42, 18.08, 10.82, 7.74, 5.94, 4.49, 3.40, and 3.07, in order. The progression of FIDs is detailed in Fig. 9 in the Appendix.

designated as Score identity Distillation (SiD), is named to underscore its roots in these three identities.

We validate the effectiveness and efficiency of SiD across all four benchmark datasets considered in Karras et al. (2022): CIFAR-10 32x32, ImageNet 64x64, FFHQ 64x64, and AFHQv2 64x64. The SiD single-step generator is trained using the VP-EDM checkpoints as the teacher diffusion models. It achieves state-of-the-art performance across all four datasets in providing high-quality generation, measured by Fréchet inception distance (FID) (Heusel et al., 2017), and also facilitates an exponentially fast reduction in FID as the distillation progresses. This is visually corroborated by Figs. 1 and 7 and detailed in the experiments section.

2. Related Work

Significant efforts have been directed towards executing the reverse diffusion process in fewer steps. A prominent line of research involves interpreting the diffusion model through the lens of stochastic differential equations (SDE) or ordinary differential equations (ODE), followed by employing advanced numerical solvers for SDE/ODE (Song et al., 2020; Liu et al., 2022a; Lu et al., 2022; Zhang & Chen, 2023; Karras et al., 2022; Xue et al., 2023). Despite these advancements, there remains a pronounced trade-off between reducing steps and preserving visual quality. Another line of work considers the diffusion model within the framework of flow matching, applying strategies to simplify the reverse diffusion process into more linear trajectories, thereby facilitating larger step advancements (Liu et al., 2022b; Lipman et al., 2022). To achieve generation within fewer steps, researchers also propose to truncate the diffusion chain and starting the generation from an implicit distribution instead of white Gaussian noise (Pandey et al., 2022; Zheng et al., 2023a; Lyu et al., 2022) and combining it with GANs for faster generation (Xiao et al., 2022; Wang et al., 2023c).

A unique avenue of research focuses on distilling the reverse diffusion chains (Luhman & Luhman, 2021; Salimans & Ho, 2022; Zheng et al., 2023b; Luo et al., 2023b). Salimans & Ho (2022) pioneered the concept of progressive distillation, which Meng et al. (2023) took further into the realm of guided diffusion models equipped with classifier-free guidance. Subsequent advancements introduced consistency models (Song et al., 2023) as an innovative strategy for distilling diffusion models, which promotes output consistency throughout the ODE trajectory. Song & Dhariwal (2023) further enhanced the generation quality of these consistency models through extensive engineering efforts and new theoretical insights. Pushing the boundaries further, Kim et al. (2023) improved prediction consistency at any intermediate stage and incorporated GAN-based loss to elevate image quality. Extending these principles, Luo et al. (2023a) applied consistency distillation techniques to text-guided latent diffusion models (Ramesh et al., 2022), facilitating efficient and high-fidelity text-to-image generation.

Recent research has focused on distilling diffusion models into generators capable of one or two step operations through adversarial training (Sauer et al., 2023). Following Diffusion-GAN (Wang et al., 2023c), which trains a one-step generator by minimizing the Jensen–Shannon divergence (JSD) at each diffusion time step, Xu et al. (2023) introduced UFOGen, which distills diffusion models using a time-step dependent discriminator, mirroring the initialization of the generator. UFOGen has shown proficiency in one-step text-guided image generation. Text-to-3D synthesis, using a pretrained 2D text-to-image diffusion model, effectively acts as a distillation process, and leverages the direction indicated by the score function of the 2D diffu-

sion model to guide the generation of various views of 3D objects (Poole et al., 2022; Wang et al., 2023b). Building on this concept, Diff-Instruct (Luo et al., 2023c) applies this principle to distill general pretrained diffusion models into a single-step generator, while SwiftBrush (Nguyen & Tran, 2023) further illustrates its effectiveness in distilling pretrained stable diffusion models. Distribution Matching Distillation (DMD) of Yin et al. (2023) aligns closely with this principle, and further introduces an additional regression loss term to improve the quality of distillation.

It is important to note that both Diff-Instruct and DMD are fundamentally aligned with the approach first introduced by Diffusion-GAN (Wang et al., 2023c). This method entails training the generator by aligning the diffused real and fake distributions. The primary distinction lies in whether the JSD or KL divergence is employed for any given noise level. From this perspective, the proposed SiD method adheres to the framework established by Diffusion-GAN and subsequently embraced by Diff-Instruct and DMD. However, SiD distinguishes itself by implementing a model-based score-matching loss, notably a variant of Fisher divergence, moving away from the traditional use of JSD or KL divergence applied to diffused real and fake distributions. Furthermore, it uncovers an effective strategy to approximate this loss, which is analytically intractable. In the sections that follow, we delve into both the distinctive loss mechanism and the method for its approximation, illuminating SiD's innovative strategy.

3. Forward Diffusion as Semi-Implicit Distribution: Exploring Score Identities

The marginal of a mixture distribution can be expressed as $p(\boldsymbol{x}) = \int p(\boldsymbol{x} \,|\, \boldsymbol{z}) p(\boldsymbol{z}) \mathrm{d}\boldsymbol{z}$. In cases where $p(\boldsymbol{x} \,|\, \boldsymbol{z})$ is analytically defined and $p(\boldsymbol{z})$ is straightforward to sample from, yet the marginal distribution is intractable or difficult to compute, we follow Yin & Zhou (2018) to refer to it as a semi-implicit distribution. This semi-implicit framework and its derivatives have been widely used to develop flexible variational distributions with tractable parameter inference, as evidenced by a series of studies (Yin & Zhou, 2018; Molchanov et al., 2019; Hasanzadeh et al., 2019; Titsias & Ruiz, 2019; Sobolev & Vetrov, 2019; Lawson et al., 2019; Moens et al., 2021; Yu & Zhang, 2023; Yu et al., 2023).

In our study, we explore the vital role of semi-implicit distributions in the forward diffusion process. We observe that both the observed real data and the generated fake data adhere to semi-implicit distributions in this process. The gradients of their log-likelihoods, commonly known as scores, can be formulated as the expectation of certain random variables. These expectations are amenable to approximation through deep neural networks or Monte Carlo estimation. This reformulation of the scores is enabled through the application of the semi-implicit framework, allowing for the introduction of three critical identities pertinent to score estimation, as detailed subsequently.

3.1. Forward Diffusions and Semi-Implicit Distributions

The forward marginal of a diffusion model is an exemplary illustration of a semi-implicit distribution, expressed as:

$$p_{\text{data}}(\boldsymbol{x}_t) = \int q(\boldsymbol{x}_t \,|\, \boldsymbol{x}_0) p_{\text{data}}(\boldsymbol{x}_0) \, \mathrm{d}\boldsymbol{x}_0, \tag{1}$$

where the forward conditional $q(\boldsymbol{x}_t \,|\, \boldsymbol{x}_0)$ is analytically defined, but the data distribution $p_{\text{data}}(\boldsymbol{x}_0)$ remains unknown and is typically represented through empirical samples. In this paper, we focus on Gaussian diffusion models, where the forward conditional follows a Gaussian distribution:

$$q(\boldsymbol{x}_t \,|\, \boldsymbol{x}_0) = \mathcal{N}(a_t \boldsymbol{x}_0, \sigma_t^2 \mathbf{I}),$$

with $a_t \in [0, 1]$. To generate a diffused sample from $\boldsymbol{x}_0 \sim p_{\text{data}}(\boldsymbol{x}_0)$, reparameterization is often employed:

$$\boldsymbol{x}_t := a_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I}).$$

As a_t can be assimilated into the preconditioning of neural network inputs without sacrificing generality, we set $a_t = 1$ for simplicity, in line with Karras et al. (2022).

While the exact form of $p_{\text{data}}(\boldsymbol{x}_t)$ and hence the score $S(\boldsymbol{x}_t) := \nabla_{\boldsymbol{x}_t} \ln p_{\text{data}}(\boldsymbol{x}_t)$ are not known, the score of the forward conditional $q(\boldsymbol{x}_t \,|\, \boldsymbol{x}_0)$ has an analytic expression as

$$\nabla_{\boldsymbol{x}_t} \ln q(\boldsymbol{x}_t \,|\, \boldsymbol{x}_0) = \sigma_t^{-2} (\boldsymbol{x}_0 - \boldsymbol{x}_t) = -\sigma_t^{-1} \boldsymbol{\epsilon}_t. \tag{2}$$

In this work, we explore an implicit generator $p_\theta(\boldsymbol{x}_g)$, parameterized by θ, which generates random samples as $\boldsymbol{x}_g = G_\theta(\boldsymbol{z}), \boldsymbol{z} \sim p(\boldsymbol{z})$, where $G_\theta(\cdot)$ represents a neural network, parameterized by θ, that deterministically transforms noise $\boldsymbol{z} \sim p(\boldsymbol{z})$ into generated data $\boldsymbol{x}_g$. If the distribution of $p_\theta(\boldsymbol{x}_g)$ matches that of $p_{\text{data}}(\boldsymbol{x}_0)$, it then follows that the semi-implicit distribution

$$p_\theta(\boldsymbol{x}_t) = \int q(\boldsymbol{x}_t \,|\, \boldsymbol{x}_g) p_\theta(\boldsymbol{x}_g) \, \mathrm{d}\boldsymbol{x}_g \tag{3}$$

would be identical to $p_{\text{data}}(\boldsymbol{x}_t)$ for any t. Conversely, as proved in Wang et al. (2023c), if $p_\theta(\boldsymbol{x}_t)$ coincides with $p_{\text{data}}(\boldsymbol{x}_t)$ for any t, this implies a match between the generator distribution $p_\theta(\boldsymbol{x}_g)$ and the data distribution $p_{\text{data}}(\boldsymbol{x}_0)$.

3.2. Score Identities

In this paper, we illustrate that the semi-implicit distribution defined in (3) is characterized by three crucial identities, each playing a vital role in score-based distillation. The first identity concerns the diffused real data distribution, a well-established concept fundamental to denoising score matching. The second identity is analogous to the first but applies to diffused generator distributions. The third identity, though not as widely recognized, is essential for the development of our proposed method.

Identity 1 (Tweedie's Formula for Diffused Real Data)**.** *Consider the semi-implicit distribution $p_{data}(\boldsymbol{x}_t)$ in (1), defined by diffusing real data. The expected value of $\boldsymbol{x}_0$ given $\boldsymbol{x}_t$, in line with $q(\boldsymbol{x}_0 \mid \boldsymbol{x}_t) = \frac{q(\boldsymbol{x}_t \mid \boldsymbol{x}_0) p_{\text{data}}(\boldsymbol{x}_0)}{p_{\text{data}}(\boldsymbol{x}_t)}$ as per Bayes' rule, is connected to the score of $p_{\text{data}}(\boldsymbol{x}_t)$ as*

$$\mathbb{E}[\boldsymbol{x}_0 \mid \boldsymbol{x}_t] = \int \boldsymbol{x}_0 q(\boldsymbol{x}_0 \mid \boldsymbol{x}_t) \, \mathrm{d}\boldsymbol{x}_0 = \boldsymbol{x}_t + \sigma_t^2 \nabla_{\boldsymbol{x}_t} \ln p_{\text{data}}(\boldsymbol{x}_t). \quad (4)$$

This identity, known as Tweedie's Formula (Robbins, 1992; Efron, 2011), provides an estimate for the original data $\boldsymbol{x}_0$ given $\boldsymbol{x}_t$, where $\boldsymbol{x}_t$ is generated as $\boldsymbol{x}_t \sim \mathcal{N}(\boldsymbol{x}_0, \sigma_t^2 \mathbf{I})$, $\boldsymbol{x}_0 \sim p_{\text{data}}(\boldsymbol{x}_0)$. The significance of this relationship has been discussed in Luo (2022) and Chung et al. (2022). An equivalent identity applies to the diffused fake data distribution.

Identity 2 (Tweedie's Formula for Diffused Fake Data)**.** *For the semi-implicit distribution $p_\theta(\boldsymbol{x}_t)$ defined in (3), resulting from diffusing fake data, the expected value of $\boldsymbol{x}_g$ given $\boldsymbol{x}_t$, following $q(\boldsymbol{x}_g \mid \boldsymbol{x}_t) = \frac{q(\boldsymbol{x}_t \mid \boldsymbol{x}_g) p_\theta(\boldsymbol{x}_g)}{p_\theta(\boldsymbol{x}_t)}$ according to Bayes' rule, is associated with the score of $p_\theta(\boldsymbol{x}_t)$ as*

$$\mathbb{E}[\boldsymbol{x}_g \mid \boldsymbol{x}_t] = \int \boldsymbol{x}_g q(\boldsymbol{x}_g \mid \boldsymbol{x}_t) \, \mathrm{d}\boldsymbol{x}_g = \boldsymbol{x}_t + \sigma_t^2 \nabla_{\boldsymbol{x}_t} \ln p_\theta(\boldsymbol{x}_t). \quad (5)$$

Transitioning from the initial two identities, we introduce a third identity that is crucial for the proposed computational methodology of score distillation, taking advantage of the properties of semi-implicit distributions.

Identity 3 (Score Projection Identity)**.** *Given the intractability of $\nabla_{\boldsymbol{x}_t} \ln p_\theta(\boldsymbol{x}_t)$, we introduce a projection vector to estimate the expected value of its product with the score:*

$$\begin{aligned} &\mathbb{E}_{\boldsymbol{x}_t \sim p_\theta(\boldsymbol{x}_t)} \left[u^T(\boldsymbol{x}_t) \nabla_{\boldsymbol{x}_t} \ln p_\theta(\boldsymbol{x}_t) \right] \\ &= \mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{x}_g) \sim q(\boldsymbol{x}_t \mid \boldsymbol{x}_g) p_\theta(\boldsymbol{x}_g)} \left[u^T(\boldsymbol{x}_t) \nabla_{\boldsymbol{x}_t} \ln q(\boldsymbol{x}_t \mid \boldsymbol{x}_g) \right]. \end{aligned}$$

This identity was leveraged by Vincent (2011) to draw a parallel between the explicit score matching (ESM) loss,

$$\mathcal{L}_{\text{ESM}} = \mathbb{E}_{\boldsymbol{x}_t \sim p_{\text{data}}(\boldsymbol{x}_t)} \left\| S_\phi(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_t} \ln p_{\text{data}}(\boldsymbol{x}_t) \right\|_2^2, \quad (6)$$

and denoising score matching (DSM) loss, given by

$$\mathcal{L}_{\text{DSM}} = \mathbb{E}_{q(\boldsymbol{x}_t \mid \boldsymbol{x}_0) p_{\text{data}}(\boldsymbol{x}_0)} \left\| S_\phi(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_t} \ln q(\boldsymbol{x}_t \mid \boldsymbol{x}_0) \right\|_2^2. \quad (7)$$

Integrating the DSM loss with Unet architectures (Ronneberger et al., 2015) and stochastic-gradient Langevin dynamics (Welling & Teh, 2011) based reverse sampling, Song & Ermon (2019) have elevated score-based, or diffusion, models to a prominent position in deep generative modeling. Additionally, Yu & Zhang (2023) used this identity for semi-implicit variational inference (Yin & Zhou, 2018), while Yu et al. (2023) applied it to refine multi-step reverse diffusion.

Distinct from these prior applications of this identity, we integrate it with two previously discussed identities. This fusion, combined with a model-based score-matching loss, culminates in a unique loss mechanism facilitating single-step distillation of a pretrained diffusion model.

4. SiD: Score identity Distillation

In this section, we introduce the model-based score-matching loss as the theoretical basis for our distillation loss. We then demonstrate how the three identities previously discussed can be fused to approximate this loss.

4.1. Model-based Explicit Score Matching (MESM)

Assuming the existence of a diffusion model for the data, with parameter ϕ pretrained to estimate the score $\nabla_{\boldsymbol{x}_t} \ln p_{\text{data}}(\boldsymbol{x}_t)$, we use the following approximation:

$$\nabla_{\boldsymbol{x}_t} \ln p_{\text{data}}(\boldsymbol{x}_t) \approx S_\phi(\boldsymbol{x}_t) := \sigma_t^{-2}(f_\phi(\boldsymbol{x}_t, t) - \boldsymbol{x}_t).$$

In other words, we adopt $f_\phi(\boldsymbol{x}_t, t) \approx \mathbb{E}[\boldsymbol{x}_0 \mid \boldsymbol{x}_t]$ as our approximation, according to (4) in Identity 1. Our goal is to distill the knowledge encapsulated in ϕ, extracting which for data generation typically requires many iterations through the same network $f_\phi(\cdot, \cdot)$.

We use the pretrained score $S_\phi(\boldsymbol{x}_t)$ to construct our distillation loss. Our aim is to train G_θ to distill the iterative, multi-step reverse diffusion process into a single network evaluation step. For a specific reverse diffusion time step $t \sim p(t)$, we define the theoretical distillation loss as

$$\mathcal{L}_\theta = \mathbb{E}_{\boldsymbol{x}_t \sim p_\theta(\boldsymbol{x}_t)}[\|\delta_{\phi,\theta}(\boldsymbol{x}_t)\|_2^2], \quad (8)$$

$$\delta_{\phi,\theta}(\boldsymbol{x}_t) := S_\phi(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_t} \ln p_\theta(\boldsymbol{x}_t). \quad (9)$$

We refer to $\delta_{\phi,\theta}(\boldsymbol{x}_t)$ as score difference and designate its expected L2 norm $\mathcal{L}_\theta$ as the model-based explicit score-matching (MESM) loss, also known in the literature as a Fisher divergence (Lyu, 2009; Holmes & Walker, 2017; Yang et al., 2019; Yu & Zhang, 2023). This differs from the ESM loss defined in (6) as the expectation is computed with respect to the diffused fake data distribution $p_\theta(\boldsymbol{x}_t)$ rather than diffused real data distribution $p_{\text{data}}(\boldsymbol{x}_t)$.

A common assumption is that the performance of the student model used for distillation will be limited by the outcomes of reverse diffusion using $S_\phi(\boldsymbol{x}_t)$, the teacher model. However, our results demonstrate that the student model, utilizing single-step generation, can indeed exceed the performance of the teacher model, EDM of Karras et al. (2022), which relies on iterative refinement. This indicates that the aforementioned hypothesis might not necessarily hold true. It implies that reverse diffusion could accumulate errors throughout its process, even with very fine-grained reverse steps and the use of advanced numerical solvers designed to counteract error accumulations.

4.2. Loss Approximation based on Identities 1 and 2

To estimate the score $\nabla_{\boldsymbol{x}_t} \ln p_\theta(\boldsymbol{x}_t)$, an initial thought would be to adopt a deep neural network-based approximation $f_\psi(\boldsymbol{x}_t, t) \approx \mathbb{E}[\boldsymbol{x}_g \mid \boldsymbol{x}_t]$ by (5), which can be trained

with the usual diffusion or denoising score-matching loss as

$$\min_\psi \mathbb{E}_{q(\boldsymbol{x}_t \mid \boldsymbol{x}_g, t) p_\theta(\boldsymbol{x}_g)}[\gamma(t) \| f_\psi(\boldsymbol{x}_t, t) - \boldsymbol{x}_g \|_2^2], \quad (10)$$

where the timestep distribution $t \sim p(t)$ and weighting function $\gamma(t)$ can be defined as in Karras et al. (2022). Assuming $\boldsymbol{x}_t \sim q(\boldsymbol{x}_t \mid \boldsymbol{x}_g)$, $\boldsymbol{x}_g = G_\theta(\boldsymbol{z})$, $\boldsymbol{z} \sim p(\boldsymbol{z})$, the optimal solution $\psi^*(\theta)$, which depends on the generator distribution determined by θ, satisfies

$$f_{\psi^*(\theta)}(\boldsymbol{x}_t, t) = \mathbb{E}[\boldsymbol{x}_g \mid \boldsymbol{x}_t] = \boldsymbol{x}_t + \sigma_t^2 \nabla_{\boldsymbol{x}_t} \ln p_\theta(\boldsymbol{x}_t)$$

and we can express the score difference defined in (9) as

$$\delta_{\phi, \psi^*(\theta)}(\boldsymbol{x}_t) = \sigma_t^{-2}(f_\phi(\boldsymbol{x}_t, t) - f_{\psi^*(\theta)}(\boldsymbol{x}_t, t)). \quad (11)$$

As $\psi^*(\theta)$ depends on θ, the minimization of $\mathcal{L}_\theta$ in (8) could potentially be cast as a bilevel optimization problem (Ye et al., 1997; Hong et al., 2023; Shen et al., 2023).

It is tempting to estimate the score difference $\delta_{\phi, \psi^*(\theta)}(\boldsymbol{x}_t)$ using an approximated score difference defined as

$$\delta_{\phi, \psi}(\boldsymbol{x}_t) := \sigma_t^{-2}(f_\phi(\boldsymbol{x}_t, t) - f_\psi(\boldsymbol{x}_t, t)), \quad (12)$$

which means we approximate $\psi^*(\theta)$ with ψ, ignoring its dependence on θ, and define an approximated MESM loss $\mathcal{L}_\theta^{(1)}$ as

$$\mathcal{L}_\theta \approx \mathcal{L}_\theta^{(1)} := \mathbb{E}_{\boldsymbol{x}_t \sim p_\theta(\boldsymbol{x}_t)} \left[\| \delta_{\phi, \psi}(\boldsymbol{x}_t) \|_2^2 \right]. \quad (13)$$

However, defining the score approximation error as

$$\triangle_{\psi, \psi^*(\theta)}(\boldsymbol{x}_t) := \sigma_t^{-2}(f_\psi(\boldsymbol{x}_t, t) - f_{\psi^*(\theta)}(\boldsymbol{x}_t, t)), \quad (14)$$

we have $\delta_{\phi, \psi}(\boldsymbol{x}_t) = \delta_{\phi, \psi^*(\theta)}(\boldsymbol{x}_t) - \triangle_{\psi, \psi^*(\theta)}(\boldsymbol{x}_t)$ and

$$\begin{aligned} \mathcal{L}_\theta^{(1)} = \mathcal{L}_\theta + \mathbb{E}_{p_\theta(\boldsymbol{x}_t)} \big[& \| \triangle_{\psi, \psi^*(\theta)}(\boldsymbol{x}_t) \|_2^2 \\ & - 2 \triangle_{\psi, \psi^*(\theta)}(\boldsymbol{x}_t)^T \delta_{\phi, \psi^*(\theta)}(\boldsymbol{x}_t) \big]. \end{aligned} \quad (15)$$

Therefore, how well $\mathcal{L}_\theta^{(1)}$ approximates the true loss $\mathcal{L}_\theta$ heavily depends on not only the score approximation error $\triangle_{\psi, \psi^*(\theta)}(\boldsymbol{x}_t)$ but also the score difference $\delta_{\phi, \psi^*(\theta)}(\boldsymbol{x}_t)$. For a given θ, although one can control $\triangle_{\psi, \psi^*(\theta)}(\boldsymbol{x}_t)$ by minimizing (10), it would be difficult to ensure that influence of the score difference $\delta_{\phi, \psi^*(\theta)}(\boldsymbol{x}_t)$ would not dominate the true loss $\mathcal{L}_\theta$, especially during the intial phase of training when $p_\theta(\boldsymbol{x}_t)$ does not match $p_{\text{data}}(\boldsymbol{x}_t)$ well.

This concern is confirmed through the distillation of EDM models pretrained on CIFAR-10, employing a loss estimated via reparameterization and Monte Carlo estimation as

$$\hat{\mathcal{L}}_\theta^{(1)} = \| \delta_{\phi, \psi}(\boldsymbol{x}_t) \|_2^2, \quad (16)$$

$$\boldsymbol{x}_t = \boldsymbol{x}_g + \sigma_t \boldsymbol{\epsilon}_t, \ \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad (17)$$

$$\boldsymbol{x}_g = G_\theta(\sigma_{\text{init}} \boldsymbol{z}), \ \boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \quad (18)$$

This loss fails to yield meaningful results. Below, we present a toy example that highlights a failure case when using $\hat{\mathcal{L}}_\theta^{(1)}$ as the loss function to optimize θ.

Proposition 4 (An example failure case)**.** *Suppose* $p_{data}(x_0) = \mathcal{N}(0, 1)$, $p_{data}(x_t) = \mathcal{N}(0, 1 + \sigma_t^2)$, $q(x_t \mid x_g) = \mathcal{N}(x_g, \sigma_t^2)$, *and* $p_\theta(x_g) = \mathcal{N}(\theta, 1)$. *Assume* $\psi^*(\theta) = \theta$ *and* $f_\psi(x_t, t) = x_t(1+\sigma_t^2)^{-1} + \psi \sigma_t^2 (1+\sigma_t^2)^{-1}$. *Then we have*

(i) $\delta_{\phi, \psi^*(\theta)}(x_t) = -\frac{\theta}{1+\sigma_t^2}$, $\delta_{\phi, \psi}(x_t) = -\frac{\psi}{1+\sigma_t^2}$;

(ii) $\mathcal{L}_\theta = \frac{\theta^2}{(1+\sigma_t^2)^2}$, $\hat{\mathcal{L}}_\theta^{(1)} = \frac{\psi^2}{(1+\sigma_t^2)^2}$.

The proof is presented in Appendix D. The example in this proposition shows that although minimizing the objective $\mathcal{L}_\theta$ leads to the optimal generator parameter $\theta^* = 0$, the loss $\hat{\mathcal{L}}_\theta^{(1)}$ would provide no meaningful gradient towards θ^*.

4.3. Loss Approximation via Projected Score Matching

We provide an alternative formulation of the MESM loss:

Theorem 5 (Projected Score Matching)**.** *The MESM loss in* (8) *can be equivalently expressed as*

$$\mathcal{L}_\theta = \mathbb{E}_{q(\boldsymbol{x}_t \mid \boldsymbol{x}_g, t) p_\theta(\boldsymbol{x}_g)} \left[\sigma_t^{-2} \delta_{\phi, \psi^*(\theta)}(\boldsymbol{x}_t)^T (f_\phi(\boldsymbol{x}_t, t) - \boldsymbol{x}_g) \right]. \quad (19)$$

The proof, based on Identity 3, is deferred to Appendix C. We approximate the loss by substituting $\psi^*(\theta)$ in (19) with its approximation ψ, leading to an approximated loss $\mathcal{L}_\theta^{(2)}$ as

$$\begin{aligned} \mathcal{L}_\theta^{(2)} &= \mathbb{E}_{q(\boldsymbol{x}_t \mid \boldsymbol{x}_g, t) p_\theta(\boldsymbol{x}_g)} \left[\sigma_t^{-2} \delta_{\phi, \psi}(\boldsymbol{x}_t)^T (f_\phi(\boldsymbol{x}_t, t) - \boldsymbol{x}_g) \right] \\ &= \mathcal{L}_\theta - \mathbb{E}_{q(\boldsymbol{x}_t \mid \boldsymbol{x}_g, t) p_\theta(\boldsymbol{x}_g)} \\ &\qquad \left[\sigma_t^{-2} \triangle_{\psi, \psi^*(\theta)}(\boldsymbol{x}_t)^T (f_\phi(\boldsymbol{x}_t, t) - \boldsymbol{x}_g) \right]. \end{aligned} \quad (20)$$

Comparing (20) to (15) indicates that $\mathcal{L}_\theta^{(2)}$ is directly influenced by neither the norm $\| \triangle_{\psi, \psi^*(\theta)}(\boldsymbol{x}_t) \|_2^2$ nor the score difference $\delta_{\phi, \psi^*(\theta)}(\boldsymbol{x}_t)$ given by (11). Initially in training, the discrepancy between the estimated and actual scores for the generator distribution may amplify the value of $\triangle_{\psi, \psi^*(\theta)}(\boldsymbol{x}_t)$, whereas the difference between the pre-trained score for the real data distribution and the actual score for the generator distribution may inflate $\delta_{\phi, \psi^*(\theta)}(\boldsymbol{x}_t)$. By contrast, the term $f_\phi(\boldsymbol{x}_t, t) - \boldsymbol{x}_g$ within (20) reflects the efficacy of the pre-trained model in denoising corrupted fake data, which tends to be more stable.

Let's verify the failure case for $\mathcal{L}_\theta^{(1)}$ and see whether it is still the case for $\mathcal{L}_\theta^{(2)}$.

Proposition 6. *Under the setting of Proposition 4, the gradient of loss* $\mathcal{L}_\theta^{(2)}$ *can be estimated as*

$$\nabla_\theta \hat{L}_\theta^{(2)} = -(1 + \sigma_t^2)^{-1} \delta_{\phi, \psi}(\boldsymbol{x}_t) \nabla_\theta G_\theta(\sigma_{init} \boldsymbol{z}),$$

which involves the product of the approximated score difference $\delta_{\phi, \psi}(\boldsymbol{x}_t) = -\frac{\psi}{1+\sigma_t^2}$ *and the gradient of the generator.*

We note the product of the approximated score difference and $\nabla_\theta G_\theta(\sigma_{\text{init}}\boldsymbol{z})$ is used to construct the loss for Diff-Instruct (Luo et al., 2023c), which has been shown to be able to distill a pretrained diffusion model with satisfactory performance. Thus for the toy example where $\mathcal{L}_\theta^{(1)}$ fails, using $\mathcal{L}_\theta^{(2)}$ can provide useful gradient to guide the generator.

4.4. Fused Loss of SiD

Examining $\mathcal{L}_\theta^{(2)}$ and $\mathcal{L}_\theta^{(1)}$ unveils their interconnections:

$$\mathcal{L}_\theta^{(2)} = \mathcal{L}_\theta^{(1)} + \mathbb{E}_{\boldsymbol{x}_g\sim p_\theta(\boldsymbol{x}_g)}\mathbb{E}_{\boldsymbol{x}_t\sim q(\boldsymbol{x}_t\,|\,\boldsymbol{x}_g,t)} \left[\sigma_t^{-2}\delta_{\phi,\psi}(\boldsymbol{x}_t)^T(f_\psi(\boldsymbol{x}_t,t)-\boldsymbol{x}_g)\right]. \tag{21}$$

Empirically, while $\hat{\mathcal{L}}_\theta^{(1)}$ fails, our distillation experiments on CIFAR-10 reveal that $\hat{\mathcal{L}}_\theta^{(2)}$ performs well in terms of Inception Score (IS), but yields poor FID. This outcome is illustrated in the visualizations for $\alpha = 0$ in Figs. 7 and 8.

Visual inspection indicates that the generated images are darker in comparison to the training images. Given that $\hat{\mathcal{L}}_\theta^{(1)}$ fails while $\hat{\mathcal{L}}_\theta^{(2)}$ shows promis, albeit with poor FID due to mismatched color, we hypothesize that the difference term

$$\hat{\mathcal{L}}_\theta^{\triangle} = \hat{\mathcal{L}}_\theta^{(2)} - \hat{\mathcal{L}}_\theta^{(1)} = \sigma_t^{-2}\delta_{\phi,\psi}(\boldsymbol{x}_t)^T(f_\psi(\boldsymbol{x}_t,t)-\boldsymbol{x}_g)$$

directs the gradient towards the desired direction.

Thus we are propelled to consider the loss

$$\mathcal{L}_\theta^{(2)} - \alpha\mathcal{L}_\theta^{(1)} = (1-\alpha)\mathcal{L}_\theta^{(1)} + \mathcal{L}_\theta^{\triangle}. \tag{22}$$

We empirically find that setting $\alpha \in [-0.25, 1.2]$ produces visually coherent images, with $\alpha \in [0.75, 1.2]$ typically leading to superior results, as shown in Figs. 7 and 8.

In summary, the weighted loss is expressed as

$$\tilde{L}_\theta(\boldsymbol{x}_t,t,\phi,\psi) = (1-\alpha)\tfrac{\omega(t)}{\sigma_t^4}\|f_\phi(\boldsymbol{x}_t,t)-f_\psi(\boldsymbol{x}_t,t)\|_2^2 + \tfrac{\omega(t)}{\sigma_t^4}(f_\phi(\boldsymbol{x}_t,t)-f_\psi(\boldsymbol{x}_t,t))^T(f_\psi(\boldsymbol{x}_t,t)-\boldsymbol{x}_g), \tag{23}$$

where $\boldsymbol{x}_t$ is generated as in (18) and $\omega(t)$ are weighted coefficients that need to be specified. To compute the gradient of the above equation, SiD backpropagates the gradient through both ϕ and ψ by calculating two score gradients ($i.e.$, gradients of scores) as

$$\begin{aligned}\nabla_\theta f_\phi(\boldsymbol{x}_t,t) &= \tfrac{\partial f_\phi(\boldsymbol{x}_t,t)}{\partial \boldsymbol{x}_t}\nabla_\theta G_\theta(\sigma_{\text{init}}\boldsymbol{z})\\ \nabla_\theta f_\psi(\boldsymbol{x}_t,t) &= \tfrac{\partial f_\psi(\boldsymbol{x}_t,t)}{\partial \boldsymbol{x}_t}\nabla_\theta G_\theta(\sigma_{\text{init}}\boldsymbol{z}).\end{aligned} \tag{24}$$

This feature distinguishes SiD from Diff-Instruct and DMD that do not use score gradients $\frac{\partial f_\phi(\mathbf{x}_t,t)}{\partial \boldsymbol{x}_t}$ and $\frac{\partial f_\psi(\boldsymbol{x}_t,t)}{\partial \boldsymbol{x}_t}$.

4.5. Noise Weighting and Scheduling

The proposed SiD algorithm iteratively updates the score estimation parameters ψ, given θ, following (10), and updates the generator parameters θ, given ψ, as per (23). This alternating update scheme aligns with related approaches (Wang et al., 2023c; Luo et al., 2023c; Yin et al., 2023). Consequently, we largely adopt the methodology outlined by Luo et al. (2023c) and Yin et al. (2023) for setting model parameters, including weighting coefficients $\omega(t)$ and the distribution of $t \sim p(t)$. Specifically, denoting C as the total pixel count of an image and $\|\cdot\|_{1,sg}$ as the L1 norm combined with the stop gradient operation, we define

$$\omega(t) = C\sigma_t^4/\|\boldsymbol{x}_g - f_\phi(\boldsymbol{x}_t,t)\|_{1,sg}\ . \tag{25}$$

Choosing $\sigma_{\min} = 0.002$, $\sigma_{\max} = 80$, $\rho = 7.0$, and $t_{\max} \in [0, 1000]$, we sample $t \sim \text{Unif}[0, t_{\max}/1000]$ and define the noise levels as

$$\sigma_t = \left(\sigma_{\max}^{\frac{1}{\rho}} + (1-t)(\sigma_{\min}^{\frac{1}{\rho}} - \sigma_{\max}^{\frac{1}{\rho}})\right)^\rho. \tag{26}$$

The distillation process is outlined in Algorithm 1. The one-step generation procedure is straightforward: $\boldsymbol{x} = G_\theta(\sigma_{\text{init}}\boldsymbol{z})$, $\boldsymbol{z}\sim\mathcal{N}(\mathbf{0},\mathbf{I})$, where σ_{init}, by default set to 2.5, remains consistent throughout distillation and generation.

5. Experimental Results

Initially, we demonstrate the capability of the Score identity Distillation (SiD) generator to rapidly train and generate photo-realistic images by leveraging the pretrained score network and its own synthesized fake images. Subsequently, we conduct an ablation study to investigate the impact of the parameter α and discuss the settings of several other parameters. Through extensive experimentation, we assess both the effectiveness and efficiency of SiD in the context of diffusion-based image generation.

Datasets. To thoroughly assess the effectiveness of SiD, we utilize four representative datasets considered in Karras et al. (2022), including CIFAR-10 32×32 (cond/uncond) (Krizhevsky et al., 2009), ImageNet 64×64 (Deng et al., 2009), FFHQ 64×64 (Karras et al., 2019), and AFHQ-v2 64×64 (Choi et al., 2020).

Evaluation protocol. We measure image generation quality using FID and Inception Score (IS; Salimans et al. (2016)). Following Karras et al. (2019; 2022), we measure FIDs using 50k generated samples, with the training set used by the EDM teacher model[1] as reference. We also consider Precision and Recall (Kynkäänniemi et al., 2019) when evaluating SiD on ImageNet 64x64, where we use a predefined

[1] https://github.com/NVlabs/edm

Table 1. Comparison of various deep generative models trained on CIFAR-10 without label conditioning. The best and second-best one/few-step generators under the FID or IS metric are highlighted with **bold** and ***italic bold***, respectively.

Family	Model	NFE	FID (↓)	IS (↑)
Teacher	VP-EDM (Karras et al., 2022)	35	1.97	9.68
Diffusion	DDPM (Ho et al., 2020)	1000	3.17	9.46±0.11
	DDIM (Song et al., 2020)	100	4.16	
	DPM-Solver-3 (Lu et al., 2022)	48	2.65	
	VDM (Kingma et al., 2021)	1000	4.00	
	iDDPM (Nichol & Dhariwal, 2021)	4000	2.90	
	HSIVI-SM (Yu et al., 2023)	15	4.17	
	TDPM (Zheng et al., 2023a)	5	3.34	
	TDPM+ (Zheng et al., 2023a)	100	2.83	9.34
	VP-EDM+LEGO-PR (Zheng et al., 2023c)	35	1.88	9.84
One Step	NVAE (Vahdat & Kautz, 2020)	1	23.5	
	StyleGAN2+ADA (Karras et al., 2020)	1	5.33±0.35	***10.02***±0.07
	StyleGAN2+ADA+Tune (Karras et al., 2020)	1	2.92±0.05	9.83±0.04
	CT-StyleGAN2 (Zheng & Zhou, 2021)	1	2.9±0.4	**10.1**±0.1
	StyleGAN2+ DiffAug (Zhao et al., 2020)	1	5.79	
	ProjectedGAN (Sauer et al., 2021)	1	3.10	
	DiffusionGAN (Wang et al., 2023c)	1	3.19	
	Diffusion ProjectedGAN (Wang et al., 2023c)	1	2.54	
	KD (Luhman & Luhman, 2021)	1	9.36	
	TDPM (Zheng et al., 2023a)	1	7.34	
	PD (Salimans & Ho, 2022)	1	8.34	8.69
	Score Mismatching (Ye & Liu, 2023)	1	8.10	
	2-ReFlow (Liu et al., 2022b)	1	4.85	9.01
	DFNO (Zheng et al., 2023b)	1	3.78	
	CD-LPIPS (Song et al., 2023)	1	3.55	9.48
	iCT (Song & Dhariwal, 2023)	1	2.83	9.54
	iCT-deep (Song & Dhariwal, 2023)	1	2.51	9.76
	G-distill (Meng et al., 2023) (w=0.3)	1	7.34	8.9
	GET-Base (Geng et al., 2023)	1	6.91	9.16
	Diff-Instruct (Luo et al., 2023c)	1	4.53	9.89
	StyleGAN2+ADA+Tune+DI (Luo et al., 2023c)	1	2.71	9.86±0.04
	PID (Tee et al., 2024)	1	3.92	9.13
	TRACT (Berthelot et al., 2023)	1	3.78	
	DMD (Yin et al., 2023)	1	3.77	
	CTM (Kim et al., 2023)	1	***1.98***	
	SiD (ours), $\alpha = 1.0$	1	2.028 ±0.020	***10.017***±0.047
	SiD (ours), $\alpha = 1.2$	1	**1.923**±0.017	9.980± 0.042

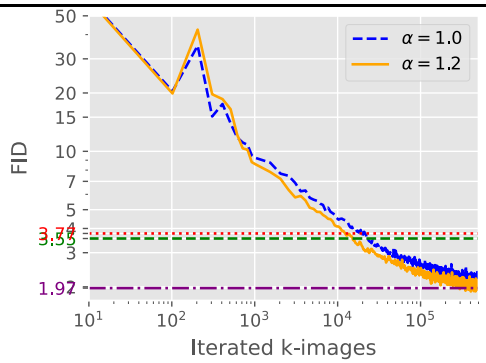

Figure 2. Evolution of FIDs for the SiD generator during the distillation of the EDM teacher model pretrained on CIFAR-10 (unconditional), using $\alpha = 1.0$ or $\alpha = 1.2$ and a batch size of 256. The performance of EDM, along with DMD and Diff-Instruct, is depicted with horizontal lines in purple, green, and red, respectively.

reference batch[2] to compute both metrics[3] (Dhariwal & Nichol, 2021; Nichol & Dhariwal, 2021; Song et al., 2023; Song & Dhariwal, 2023).

Implementation details. We implement SiD based on the EDM (Karras et al., 2022) code base and we initialize both the generator G_θ and its score estimation network f_ψ by copying the architecture and parameters of the pretrained score network f_ϕ from EDM (Karras et al., 2022). We provide the other implementation details in Appendix E.

[2] https://openaipublic.blob.core.windows.net/diffusion/jul-2021/ref_batches/imagenet/64/VIRTUAL_imagenet64_labeled.npz

[3] https://github.com/openai/guided-diffusion/tree/main/evaluations

Table 2. Analogous to Table 1 for CIFAR-10 (conditional).

Family	Model	NFE	FID (↓)
Teacher	VP-EDM (Karras et al., 2022)	35	1.79
Direct generation	BigGAN (Brock et al., 2019)	1	14.73
	StyleGAN2+ADA (Karras et al., 2020)	1	3.49±0.17
	StyleGAN2+ADA+Tune (Karras et al., 2020)	1	2.42±0.04
Distillation	GET-Base (Geng et al., 2023)	1	6.25
	Diff-Instruct (Luo et al., 2023c)	1	4.19
	StyleGAN2+ADA+Tune+DI (Luo et al., 2023c)	1	2.27
	DMD (Yin et al., 2023)	1	2.66
	DMD (*w.o.* KL) (Yin et al., 2023)	1	3.82
	DMD (*w.o. reg.*) (Yin et al., 2023)	1	5.58
	CTM (Kim et al., 2023)	1	***1.73***
	SiD (ours), $\alpha = 1.0$	1	1.932±0.019
	SiD (ours), $\alpha = 1.2$	1	**1.710**±0.011

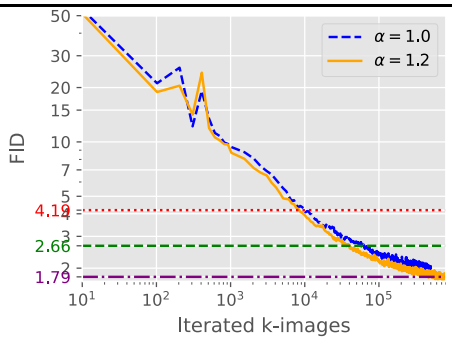

Figure 3. Analogous to Fig. 2 for CIFAR-10 (conditional).

Ablation Study and Parameter Settings. We provide ablation studies and discuss parameter settings in Appendix A.

5.1. Benchmark Performance

Our comprehensive evaluation compares SiD against leading deep generative models, encompassing both distilled diffusion models and those built from scratch. Random images generated by SiD in a single step are displayed in Figs. 13-16 in the Appendix.

The comparative analysis, detailed in Tables 1-5 and illustrated in Figs. 2-6, underlines the single-step SiD generator's proficiency in leveraging the insights from the pretrained EDM (teacher diffusion model) across a variety of benchmarks, including CIFAR-10 (both conditional and unconditional formats), ImageNet 64x64, FFHQ 64x64, and AFHQ-v2 64x64. Remarkably, the SiD-trained generator surpasses the EDM teacher in nearly all tested environments, showcasing its enhanced performance not just relative to the original multi-step teacher model but also against a broad spectrum of cutting-edge models, from traditional multi-step diffusion models to the latest single-step distilled models and GANs. The sole deviation in this pattern occurs with ImageNet 64x64, where SiD, at $\alpha = 1.2$, attains an FID of 1.524, which is exceeded by Jabri et al. (2022)'s RIN at 1.23

Table 3. Analogous to Table 1 for ImageNet 64x64 with label conditioning. The Precision and Recall metrics are also included.

Family	Model	NFE	FID (↓)	Prec. (↑)	Rec. (↑)
Teacher	VP-EDM (Karras et al., 2022)	511	1.36		
		79	2.64	0.71	0.67
Direct generation	RIN (Jabri et al., 2022)	1000	1.23		
	DDPM (Ho et al., 2020)	250	11.00	0.67	0.58
	ADM (Dhariwal & Nichol, 2021)	250	2.07	0.74	0.63
	DPM-Solver-3 (Lu et al., 2022)	50	17.52		
	HSIVI-SM (Yu et al., 2023)	15	15.49		
	U-ViT (Bao et al., 2022)	50	4.26		
	DiT-L/2 (Peebles & Xie, 2023)	250	2.91		
	LEGO (Zheng et al., 2023c)	250	2.16		
	iCT (Song & Dhariwal, 2023)	1	4.02	0.70	0.63
	iCT-deep (Song & Dhariwal, 2023)	1	3.25	0.72	0.63
Distillation	PD (Salimans & Ho, 2022)	2	8.95	0.63	**0.65**
	PD (Salimans & Ho, 2022)	1	15.39	0.59	0.62
	G-distill (Meng et al., 2023) (w=1.0)	1	7.54		
	G-distill (Meng et al., 2023) (w=0.3)	8	2.05		
	BOOT (Gu et al., 2023)	1	16.3	0.68	0.36
	PID (Tee et al., 2024)	1	9.49		
	DFNO (Zheng et al., 2023b)	1	7.83		0.61
	CD-LPIPS (Song et al., 2023)	2	4.70	0.69	0.64
	CD-LPIPS (Song et al., 2023)	1	6.20	0.68	0.63
	Diff-Instruct (Luo et al., 2023c)	1	5.57		
	TRACT (Berthelot et al., 2023)	2	4.97		
	TRACT (Berthelot et al., 2023)	1	7.43		
	DMD (Yin et al., 2023)	1	2.62		
	CTM (Kim et al., 2023)	1	1.92		0.57
	CTM (Kim et al., 2023)	2	***1.73***		0.57
	DMD (*w.o.* KL) (Yin et al., 2023)	1	9.21		
	DMD (*w.o. reg.*) (Yin et al., 2023)	1	5.61		
	SiD (ours), $\alpha = 1.0$	1	2.022 ± 0.031	***0.73***	***0.63***
	SiD (ours), $\alpha = 1.2$	1	$\mathbf{1.524} \pm 0.009$	**0.74**	***0.63***

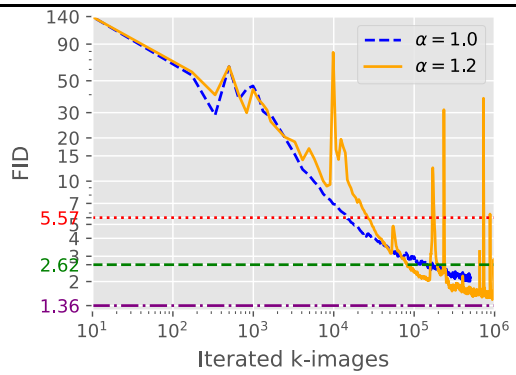

Figure 4. Analogous plot to Fig. 2 for ImageNet 64x64. The batch size is 8192. See the results of batch size 1024 in Fig. 9.

FID with 1000 steps and the teacher model VP-EDM's 1.36 FID with 511 steps.

Our assessment of SiD across various benchmarks has established, with the exception of ImageNet 64x64, potentially the first instance, to our knowledge, where a data-free diffusion distillation method outperforms the teacher diffusion model using just a single generation step. This remarkable outcome implies that reverse sampling, which utilizes the pretrained score function for generating images across multiple steps and naturally accumulates discretization errors during reverse diffusion, might not be as efficient as a single-step distillation process. The latter, by sidestepping error accumulation, could theoretically align perfectly with the true data distribution when the model-based score-matching loss is completely minimized.

Table 4. Analogous to Table 1 for FFHQ 64x64.

Family	Model	NFE	FID (↓)
Teacher	VP-EDM (Karras et al., 2022)	79	2.39
Diffusion	VP-EDM (Karras et al., 2022)	50	2.60
	Patch-Diffusion (Wang et al., 2023a)	50	3.11
Distillation	BOOT (Gu et al., 2023)	1	9.0
	SiD (ours), $\alpha = 1.0$	1	***1.710*** $\pm$ 0.018
	SiD (ours), $\alpha = 1.2$	1	**1.550** $\pm$ 0.017

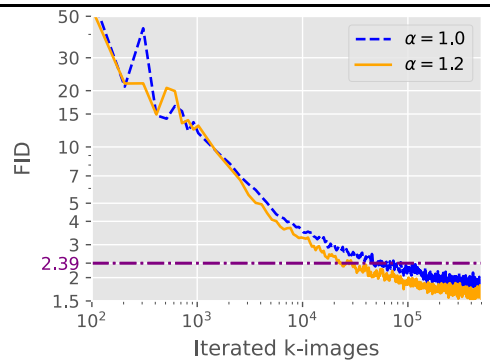

Figure 5. Analogous plot to Fig. 2 for FFHQ 64x64. The batch size is 512.

Table 5. Analogous to Table 1 for AFHQ-v2 64x64.

Family	Model	NFE	FID (↓)
Teacher	VP-EDM (Karras et al., 2022)	79	1.96
Distillation	SiD (ours), $\alpha = 1.0$	1	$\mathbf{1.628} \pm 0.017$
	SiD (ours), $\alpha = 1.2$	1	$\mathbf{1.711} \pm 0.020$

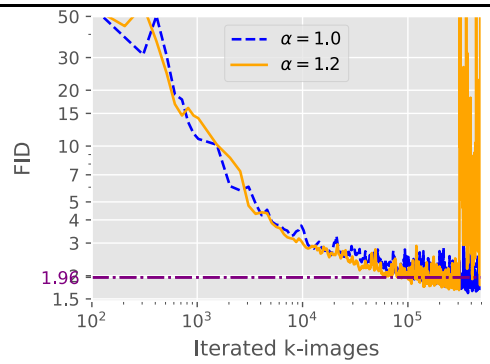

Figure 6. Ananagous plot to Fig. 2 for AFHQ-v2 64x64. The batch size is 512.

Among the single-step generators we've evaluated, CTM (Kim et al., 2023) is SiD's closest competitor in terms of generation performance. Despite the tight competition, SiD not only surpasses CTM but is also noteworthy for its independence from training data. In contrast, CTM's performance relies on access to training data and is augmented by the inclusion of an auxiliary GAN loss. This distinction significantly amplifies SiD's value, particularly in contexts where accessing the original training data is either restricted or impractical, and where data-specific GAN adjustments are undesirable.

In summary, SiD not only stands out in terms of perfor-

mance metrics but also simplifies the distillation process remarkably, operating without the need for real data. It sets itself apart by employing a notably straightforward distillation approach, unlike the complex multi-stage distillation strategy seen in Salimans & Ho (2022), the dependency on pairwise regression data in Yin et al. (2023), the use of additional GAN loss in Kim et al. (2023), or the need to access training data outlined in Song et al. (2023).

Training Iterations. In exploring SiD's performance threshold, we initially process 500 million SiD-generated synthetic images across most benchmarks. For CIFAR-10 with label conditioning, this figure increases to 800 million synthetic images for SiD with $\alpha = 1.2$. In the case of ImageNet 64x64, we extend the training for SiD with $\alpha = 1.2$ to involve 1 billion synthetic images. Through this extensive training, SiD demonstrates superior performance over the EDM teacher model across all evaluated benchmarks, with the sole exception of ImageNet 64x64, where EDM utilized 511 NFE. While we note a gradual slowing down in the rate of FID improvements, the limit of potential further reductions is not clear, indicating that with more iterations, SiD might eventually outperform EDM on ImageNet 64x64 as well.

It's noteworthy that to eclipse the achievements of rivals like Diff-instruct and DMD, SiD requires significantly fewer synthetic images than the 500 million mark, thanks to its rapid FID reduction rate. This decline often continues without evident stagnation, surpassing the teacher model's performance before the conclusion of the training. We delve into this aspect further below.

Convergence Speed. Our SiD generator, designed for distilling pretrained diffusion models, rapidly achieves the capability to generate photo-realistic images in a single step. This efficiency is showcased in Fig. 1 for the EDM model pretrained on ImageNet 64x64 and in Fig. 7 for CIFAR 32x32 (unconditional). The performance of the SiD method is further highlighted in Figs. 2-6, where the x-axis represents the thousands of images processed during training. These figures track the FID's evolution across four datasets for both $\alpha = 1$ and $\alpha = 1.2$, demonstrating a roughly linear relationship between the logarithm of the FID and the logarithm of the number of processed images. This relationship indicates that FID decreases exponentially as distillation progresses, a trend that is observed or expected to eventually slow down and approach a steady state.

For instance, on CIFAR-10 (unconditional), SiD outperforms both Diff-Instruct and DMD after processing under 20M images, achievable within fewer than 10 hours on 16 A100-40GB GPUs, or 20 hours on 8 V100-16GB GPUs. In the case of ImageNet 64x64 with a batch size of 1024 and $\alpha = 1.0$, SiD exceeds Progressive Distillation (FID 15.39) of Salimans & Ho (2022) after only around 500k generator-synthesized images (equivalent to roughly 500 iterations with a batch size of 1024), achieving FIDs lower than 5 after 7.5M images, below 4 after 13M, and under 3 after 31M images. It outperforms Diff-Instruct with fewer than 7M images processed and DMD with under 40M images. When using a larger batch size of 8192, SiD's convergence is slower, yet it attains lower FIDs: with $\alpha = 1$, it outstrips Diff-Instruct after processing less than 20M images (under 20 hours on 16 A100-40GB GPUs), and with $\alpha = 1.2$, it beats DMD after fewer than 90M images (in under 45 hours on 16 A100-40GB GPUs).

Limitations. Despite setting a new benchmark in diffusion-based generation, SiD entails the simultaneous management of three networks during the distillation process: the pretrained score network f_ϕ, the generator score network f_ψ, and the generator f_θ, which, in this study, are maintained at equal sizes. This setup demands more memory compared to traditional diffusion model training, which only necessitates retaining f_ϕ. However, the memory footprint of the two additional networks could be notably reduced by employing LoRA (Hu et al., 2022) for both f_ψ and f_θ, a possibility we aim to explore in future research.

Relative to Diff-Instruct, acknowledged for its memory and computational efficiency in distillation, as detailed in Table 6 in the Appendix for 16 A100-40GB GPUs, the memory allocation per GPU of SiD has seen a rise of around 50% for ImageNet 64x64 and about 70% for CIFAR-10, FFHQ, and AFHQ. The iteration time has increased by approximately 28% for CIFAR-10 and ImageNet 64x64, and by roughly 36% for the FFHQ and AFHQ datasets. This increase is because Diff-Instruct does not require computing score gradients, as defined in (24). By contrast, SiD necessitates computing score gradients, involving backpropagation through both the pretrained and generator score networks—a step not needed in Diff-Instruct—leading to about a one-third increase in computing time per iteration.

6. Conclusion

We present Score identity Distillation (SiD), an innovative method that transforms pretrained diffusion models into a single-step generator. By employing semi-implicit distributions, SiD aims to accomplish distillation through the minimization of a model-based score-matching loss that aligns the scores of diffused real and generative distributions across different noise intensities. Experimental outcomes underscore SiD's capability to significantly reduce the Fréchet inception distance with remarkable efficiency and outperform established generative approaches. This superiority extends across various conditions, including those using single or multiple steps, those requiring or not requiring access to training data, and those needing additional loss functions in image generation.

Acknowledgments

The authors would like to thank Dr. Zhuoran Yang and Weijian Luo for their valuable comments and suggestions. M. Zhou, H. Zheng, and Z. Wang acknowledge the support of NSF-IIS 2212418 and NIH-R37 CA271186.

Impact Statement

The positive aspect of distilled diffusion models lies in their potential to save energy and reduce costs. By simplifying and compressing large models, the deployment of distilled models often requires less computational resources, making them more energy-efficient and cost-effective. This can lead to advancements in sustainable AI practices, especially in resource-intensive applications.

However, the negative aspect arises when considering the ease of distilling models trained on violent or pornographic data. This poses significant ethical concerns, as deploying such models may inadvertently facilitate the generation and dissemination of harmful content. The distillation process, intended to transfer knowledge efficiently, could unintentionally amplify and perpetuate inappropriate patterns present in the original data. This not only jeopardizes user safety but also raises ethical and societal questions about the responsible use of AI technology. Striking a balance between the positive gains in energy efficiency and the potential negative consequences of distilling inappropriate content is crucial for the responsible development and deployment of AI models. Stringent ethical guidelines and oversight are essential to mitigate these risks and ensure the responsible use of distilled diffusion models.

References

Bao, F., Li, C., Cao, Y., and Zhu, J. All are worth words: A ViT backbone for score-based diffusion models. *arXiv preprint arXiv:2209.12152*, 2022.

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

Brock, A., Donahue, J., and Simonyan, K. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=B1xsqj09Fm.

Choi, Y., Uh, Y., Yoo, J., and Ha, J.-W. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8188–8197, 2020.

Chung, H., Sim, B., Ryu, D., and Ye, J. C. Improving diffusion models for inverse problems using manifold constraints. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=nJJjv0JDJju.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

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

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

Geng, Z., Pokle, A., and Kolter, J. Z. One-step diffusion distillation via deep equilibrium models. *Advances in Neural Information Processing Systems*, 36, 2023.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.

Gu, J., Zhai, S., Zhang, Y., Liu, L., and Susskind, J. M. BOOT: Data-free distillation of denoising diffusion models with bootstrapping. In *ICML 2023 Workshop on Structured Probabilistic Inference & Generative Modeling*, 2023.

Hasanzadeh, A., Hajiramezanali, E., Narayanan, K., Duffield, N., Zhou, M., and Qian, X. Semi-implicit graph variational auto-encoders. *Advances in neural information processing systems*, 32, 2019.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, pp. 6626–6637, 2017.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

Ho, J., Saharia, C., Chan, W., Fleet, D. J., Norouzi, M., and Salimans, T. Cascaded diffusion models for high fidelity image generation. *J. Mach. Learn. Res.*, 23(47):1–33, 2022.

Holmes, C. C. and Walker, S. G. Assigning a value to a power likelihood in a general Bayesian model. *Biometrika*, 104(2):497–503, 2017.

Hong, M., Wai, H.-T., Wang, Z., and Yang, Z. A two-timescale stochastic algorithm framework for bilevel optimization: Complexity analysis and application to actor-critic. *SIAM Journal on Optimization*, 33(1):147–180, 2023.

Hu, E. J., yelong shen, Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Jabri, A., Fleet, D., and Chen, T. Scalable adaptive computation for iterative generation. *arXiv preprint arXiv:2212.11972*, 2022.

Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T. Analyzing and improving the image quality of StyleGAN. In *Proc. CVPR*, 2020.

Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=k7FuTOWMOc7.

Kim, D., Lai, C.-H., Liao, W.-H., Murata, N., Takida, Y., Uesaka, T., He, Y., Mitsufuji, Y., and Ermon, S. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. *arXiv preprint arXiv:2310.02279*, 2023.

Kingma, D. P. and Welling, M. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

Kingma, D. P., Salimans, T., Poole, B., and Ho, J. Variational diffusion models. *arXiv preprint arXiv:2107.00630*, 2021.

Krizhevsky, A. et al. Learning multiple layers of features from tiny images. 2009.

Kynkäänniemi, T., Karras, T., Laine, S., Lehtinen, J., and Aila, T. Improved precision and recall metric for assessing generative models. *Advances in Neural Information Processing Systems*, 32, 2019.

Lawson, J., Tucker, G., Dai, B., and Ranganath, R. Energy-inspired models: Learning with sampler-induced distributions. *Advances in Neural Information Processing Systems*, 32, 2019.

Lipman, Y., Chen, R. T., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Liu, L., Ren, Y., Lin, Z., and Zhao, Z. Pseudo numerical methods for diffusion models on manifolds. In *International Conference on Learning Representations*, 2022a. URL https://openreview.net/forum?id=PlKWVd2yBkY.

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

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. DPM-solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=2uAaGwlP_V.

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

Luo, C. Understanding diffusion models: A unified perspective. *arXiv preprint arXiv:2208.11970*, 2022.

Luo, S., Tan, Y., Huang, L., Li, J., and Zhao, H. Latent consistency models: Synthesizing high-resolution images with few-step inference. *ArXiv*, abs/2310.04378, 2023a.

Luo, S., Tan, Y., Patil, S., Gu, D., von Platen, P., Passos, A., Huang, L., Li, J., and Zhao, H. Lcm-lora: A universal stable-diffusion acceleration module, 2023b.

Luo, W., Hu, T., Zhang, S., Sun, J., Li, Z., and Zhang, Z. Diff-instruct: A universal approach for transferring knowledge from pre-trained diffusion models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023c. URL https://openreview.net/forum?id=MLIs5iRq4w.

Lyu, S. Interpretation and generalization of score matching. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pp. 359–366, 2009.

Lyu, Z., Xu, X., Yang, C., Lin, D., and Dai, B. Accelerating diffusion models via early stop of the diffusion process. *arXiv preprint arXiv:2205.12524*, 2022.

Meng, C., Rombach, R., Gao, R., Kingma, D., Ermon, S., Ho, J., and Salimans, T. On distillation of guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14297–14306, 2023.

Moens, V., Ren, H., Maraval, A., Tutunov, R., Wang, J., and Ammar, H. Efficient semi-implicit variational inference. *arXiv preprint arXiv:2101.06070*, 2021.

Molchanov, D., Kharitonov, V., Sobolev, A., and Vetrov, D. Doubly semi-implicit variational inference. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 2593–2602. PMLR, 2019.

Nguyen, T. H. and Tran, A. SwiftBrush: One-step text-to-image diffusion model with variational score distillation. *arXiv preprint arXiv:2312.05239*, 2023.

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

Pandey, K., Mukherjee, A., Rai, P., and Kumar, A. DiffuseVAE: Efficient, controllable and high-fidelity generation from low-dimensional latents. *arXiv preprint arXiv:2201.00308*, 2022.

Peebles, W. and Xie, S. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

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

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

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning*, pp. 1278–1286, 2014.

Robbins, H. E. An empirical Bayes approach to statistics. In *Breakthroughs in Statistics: Foundations and basic theory*, pp. 388–394. Springer, 1992.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Ronneberger, O., P.Fischer, and Brox, T. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pp. 234–241. Springer, 2015. URL http://lmb.informatik.uni-freiburg.de/Publications/2015/RFB15a. (available on arXiv:1505.04597 [cs.CV]).

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E. L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., Ho, J., Fleet, D. J., and Norouzi, M. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022.

Salimans, T. and Ho, J. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TIdIXIpzhoI.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training GANs. In *Advances in Neural Information Processing Systems*, pp. 2234–2242, 2016.

Sauer, A., Chitta, K., Müller, J., and Geiger, A. Projected gans converge faster. *Advances in Neural Information Processing Systems*, 34:17480–17492, 2021.

Sauer, A., Lorenz, D., Blattmann, A., and Rombach, R. Adversarial diffusion distillation. *ArXiv*, abs/2311.17042, 2023.

Shen, H., Xiao, Q., and Chen, T. On penalty-based bilevel gradient descent method. *arXiv preprint arXiv:2302.05185*, 2023.

Sobolev, A. and Vetrov, D. P. Importance weighted hierarchical variational inference. *Advances in Neural Information Processing Systems*, 32, 2019.

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

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2020.

Song, Y. and Dhariwal, P. Improved techniques for training consistency models. *arXiv preprint arXiv:2310.14189*, 2023.

Song, Y. and Ermon, S. Generative Modeling by Estimating Gradients of the Data Distribution. In *Advances in Neural Information Processing Systems*, pp. 11918–11930, 2019.

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

Tee, J. T. J., Zhang, K., Kim, C., Gowda, D. N., Yoon, H. S., and Yoo, C. D. Physics informed distillation for diffusion models, 2024. URL https://openreview.net/forum?id=a24gfxA7jD.

Titsias, M. K. and Ruiz, F. Unbiased implicit variational inference. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 167–176. PMLR, 2019.

Vahdat, A. and Kautz, J. NVAE: A deep hierarchical variational autoencoder. In *Advances in neural information processing systems*, 2020.

Vincent, P. A Connection Between Score Matching and Denoising Autoencoders. *Neural Computation*, 23(7): 1661–1674, 2011.

Wang, Z., Jiang, Y., Zheng, H., Wang, P., He, P., Wang, Z., Chen, W., and Zhou, M. Patch diffusion: Faster and more data-efficient training of diffusion models. *arXiv preprint arXiv:2304.12526*, 2023a.

Wang, Z., Lu, C., Wang, Y., Bao, F., Li, C., Su, H., and Zhu, J. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation, 2023b.

Wang, Z., Zheng, H., He, P., Chen, W., and Zhou, M. Diffusion-GAN: Training GANs with diffusion. In *The Eleventh International Conference on Learning Representations*, 2023c. URL https://openreview.net/forum?id=HZf7UbpWHuA.

Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 681–688. Citeseer, 2011.

Xiao, Z., Kreis, K., and Vahdat, A. Tackling the generative learning trilemma with denoising diffusion GANs. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=JprM0p-q0Co.

Xu, Y., Zhao, Y., Xiao, Z., and Hou, T. UFOGen: You forward once large scale text-to-image generation via diffusion GANs. *ArXiv*, abs/2311.09257, 2023.

Xue, S., Yi, M., Luo, W., Zhang, S., Sun, J., Li, Z., and Ma, Z.-M. SA-Solver: Stochastic Adams solver for fast sampling of diffusion models. *Advances in Neural Information Processing Systems*, 36, 2023.

Yang, Y., Martin, R., and Bondell, H. Variational approximations using Fisher divergence. *arXiv preprint arXiv:1905.05284*, 2019.

Ye, J., Zhu, D., and Zhu, Q. J. Exact penalization and necessary optimality conditions for generalized bilevel programming problems. *SIAM Journal on optimization*, 7(2):481–507, 1997.

Ye, S. and Liu, F. Score mismatching for generative modeling. *arXiv preprint arXiv:2309.11043*, 2023.

Yin, M. and Zhou, M. Semi-implicit variational inference. In *International Conference on Machine Learning*, pp. 5660–5669, 2018.

Yin, T., Gharbi, M., Zhang, R., Shechtman, E., Durand, F., Freeman, W. T., and Park, T. One-step diffusion with distribution matching distillation, 2023.

Yu, L. and Zhang, C. Semi-implicit variational inference via score matching. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=sd90a2ytrt.

Yu, L., Xie, T., Zhu, Y., Yang, T., Zhang, X., and Zhang, C. Hierarchical semi-implicit variational inference with application to diffusion model acceleration. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=ghIBaprxsV.

Zhang, Q. and Chen, Y. Fast sampling of diffusion models with exponential integrator. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Loek7hfb46P.

Zhao, S., Liu, Z., Lin, J., Zhu, J.-Y., and Han, S. Differentiable augmentation for data-efficient gan training. *Advances in Neural Information Processing Systems*, 33: 7559–7570, 2020.

Zheng, H. and Zhou, M. Exploiting chain rule and Bayes' theorem to compare probability distributions. *Advances in Neural Information Processing Systems*, 34:14993–15006, 2021.

Zheng, H., He, P., Chen, W., and Zhou, M. Truncated diffusion probabilistic models and diffusion-based adversarial auto-encoders. In *The Eleventh International Conference on Learning Representations*, 2023a. URL https://openreview.net/forum?id=HDxgaKk956l.

Zheng, H., Nie, W., Vahdat, A., Azizzadenesheli, K., and Anandkumar, A. Fast sampling of diffusion models via operator learning. In *International Conference on Machine Learning*, pp. 42390–42402. PMLR, 2023b.

Zheng, H., Wang, Z., Yuan, J., Ning, G., He, P., You, Q., Yang, H., and Zhou, M. Learning stackable and skippable LEGO bricks for efficient, reconfigurable, and variable-resolution diffusion modeling, 2023c.

Appendix for Score identity Distillation

A. Ablation Study and Parameter Settings

Impact of α. We conduct an ablation study to examine the impact of α on SiD. In Fig. 7, we investigate a range of α values $[-0.25, 0.0, 0.5, 0.75, 1.0, 1.2, 1.5]$ during SiD training on CIFAR10-unconditional and visualize the changes in generation as training progresses under each α. For instance, the last row illustrates the generation results when 10.24 million images (equivalent to 40,000 iterations with a batch size of 256) are processed by SiD. In Fig. 8, we illustrate the evolution of the FID and IS from iterations 0 to 8000 (corresponding to 0 to 1.024 million images), where the first plot depicts the IS evolution, while the second plot shows the trajectory of FID.

The results indicate a stable performance of the model when α varies from 0 to 1.2. A negative value of α results in large FIDs. This observation supports our analysis in Section 4.2 that directly optimizing $\mathcal{L}_\theta^{(1)}$ given by (13) may not lead to meaningful improvement, as our loss, shown in (22), is $\mathcal{L}_\theta^{(2)} - \alpha\mathcal{L}_\theta^{(1)}$. As α increases within the tested range, we observe a gradual improvement in IS and FID performance, peaking at $\alpha = 1$ or $\alpha = 1.2$. Based on these findings, we select $\alpha = 1$ or $\alpha = 1.2$ for all our experiments, although a more refined grid search on α might reveal even better performance outcomes.

Setting of β_1. We investigate the β_1 parameter of the Adam optimizer for the generator score network f_ψ and the generator G_θ by setting it as either $\beta_1 = 0$, the value used in StyleGAN2 (Karras et al., 2020) and Diff-Instruct (Luo et al., 2023c), or $\beta_1 = 0.9$, a commonly used value. We find that setting $\beta_1 = 0.9$ for f_ψ often does not result in convergence, so we retain $\beta_1 = 0$ for f_ψ for all datasets. For learning G_θ, we did not observe significant benefits between setting $\beta_1 = 0$ and $\beta_1 = 0.9$, except for the FFHQ dataset, where the FID improved by more than 0.15 when changing from $\beta_1 = 0$ to $\beta_1 = 0.9$. Therefore, we set $\beta_1 = 0.9$ for G_θ in FFHQ while retaining $\beta_1 = 0$ for all other datasets.

Batch Size for ImageNet 64x64. For ImageNet 64x64, we initially set the batch size to 1024 and observed an exponential decline in FID until it suddenly diverged upon reaching or surpassing 2.62, the FID obtained by DMD (Yin et al., 2023). The exact reason for this divergence is still unclear, but we suspect it may be related to the FP16 precision used during optimization. While switching to FP32 could potentially address the issue, we have not explored this option due to its much higher computational and memory costs.

Instead, we increased the overall batch size from 1024 to 8192 (while keeping the batch per GPU unchanged at 16, requiring more gradient accumulation rounds) and reduced the learning rate from 5e-6 to 4e-6. Under $\alpha = 1$, we observed stable performance, while under $\alpha = 1.2$, we observed occasional spikes in FID. Upon examining the generations corresponding to these spikes, as shown in the fifth image of Fig. 11 and Fig. 12 in the Appendix, we found interesting patterns where certain uncommon features, such as nests containing birds, were exaggerated. However, with a batch size as large as 8192, these occasional spikes did not seem to significantly impact the overall declining trend, which was roughly log-log linear initially and gradually leveled off. With that said, when the batch size was reduced to 1024, the sudden divergence could potentially be caused by such a spike, as observed in Fig. 9.

The drawback of using a larger batch size in this case is that it takes SiD longer to outperform Diff-instruct and DMD, as clearly shown by comparing the FID trajectories in Figs. 4 and 9. Although it's feasible to develop more advanced strategies, including progressively increasing the batch size, annealing the learning rate, and implementing gradient clipping, we'll reserve these for future study.

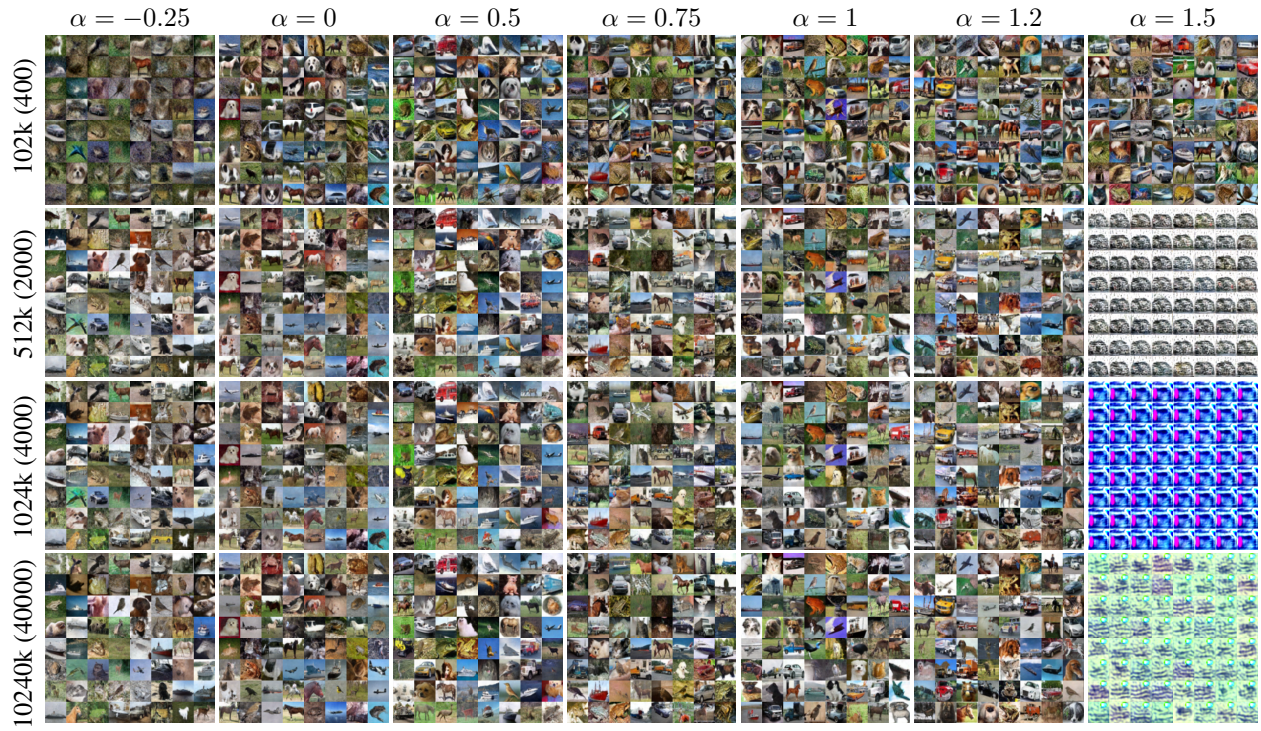

Figure 7. Ablation Study of α: The SiD generator, configured with various α values, was trained with its own synthesized images at a batch size of 256. The results, sorted by specific α values, are displayed in columns. Sequentially from top to bottom, the rows are labeled with both the total number of training images and the corresponding number of iterations, denoted as "number of images (iterations)." This labeling approach indicates the cumulative count of fake images utilized during training, corresponding to iterations of 400, 2,000, 4,000, and 40,000, progressing from the first row to the last. Across the α values of 0.5, 0.75, 1, and 1.2, minor differences are noted in both the Inception Score (IS) and visual quality, yet the Fréchet Inception Distance (FID) shows notable variations, as detailed in Fig. 8.

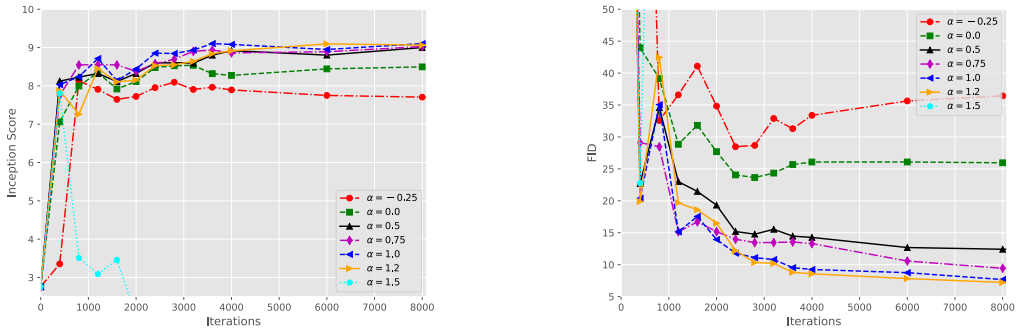

Figure 8. Ablation Study of α: Each plot illustrates the relation between the performance, measured by Inception Score and FID *vs*. the number of training iterations during the distillation of the EDM model pretrained on CIFAR-10 (unconditional), across varying values of α. The study underscores the impact of α on both training efficiency and generative fidelity, leading us to select $\alpha \in \{1.0, 1.2\}$ for all subsequent experiments.

B. Algorithm Box

Algorithm 1 Score identity Distillation (SiD)

Input: Pretrained score network f_ϕ, generator G_θ, generator score network f_ψ, $\sigma_{\text{init}} = 2.5$, $t_{\max} = 800$, $\alpha = 1.2$
Initialization $\theta \leftarrow \phi, \psi \leftarrow \phi$
repeat
Sample $\boldsymbol{z} \sim \mathcal{N}(0, \mathbf{I})$ and let $\boldsymbol{x}_g = G_\theta(\sigma_{\text{init}}\boldsymbol{z})$; Sample $t \sim p(t)$ and $\boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$, and let $\boldsymbol{x}_t = \boldsymbol{x}_g + \sigma_t\boldsymbol{\epsilon}_t$; Update ψ with Equation (10):
$\hat{\mathcal{L}}_\psi = \gamma(t)\|f_\psi(\boldsymbol{x}_t, t) - \boldsymbol{x}_g\|_2^2$
$\psi = \psi - \eta\nabla_\psi\hat{\mathcal{L}}_\psi$
where the timestep distribution $t \sim p(t)$, noise level σ_t, and weighting function $\gamma(t)$ are defined as in Karras et al. (2022).
Sample $\boldsymbol{z} \sim \mathcal{N}(0, \mathbf{I})$ and let $\boldsymbol{x}_g = G_\theta(\sigma_{\text{init}}\boldsymbol{z})$; Sample $t \sim \text{Unif}[0, t_{\max}/1000]$, compute σ_t with Equation (26), compute ω_t with Equation (25), and let $\boldsymbol{x}_t = \boldsymbol{x}_g + \sigma_t\boldsymbol{\epsilon}_t$; Update G_θ with Equation (23):
$\tilde{\mathcal{L}}_\theta = (1-\alpha)\frac{\omega(t)}{\sigma_t^4}\|f_\phi(\boldsymbol{x}_t, t) - f_\psi(\boldsymbol{x}_t, t)\|_2^2$
$\quad + \frac{\omega(t)}{\sigma_t^4}(f_\phi(\boldsymbol{x}_t, t) - f_\psi(\boldsymbol{x}_t, t))^T(f_\psi(\boldsymbol{x}_t, t) - \boldsymbol{x}_g)$
$\theta = \theta - \eta\nabla_\theta\tilde{\mathcal{L}}_\theta$
until the FID plateaus or the training budget is exhausted
Output: G_θ

C. Proofs

Proof of Tweedie's formula. For Gaussian diffusion, we have (2), which we explore to derive the identity shown below. While $p_\theta(\boldsymbol{x}_t)$ often does not have an analytic form, exploiting its semi-implicit construction, its score can be expressed as

$$\begin{aligned}
\nabla_{\boldsymbol{x}_t}\ln p_\theta(\boldsymbol{x}_t) &= \frac{\int \nabla_{\boldsymbol{x}_t} q(\boldsymbol{x}_t \mid \boldsymbol{x}_g)p_\theta(\boldsymbol{x}_g)\mathrm{d}\boldsymbol{x}_g}{p_\theta(\boldsymbol{x}_t)} \\
&= \frac{\int q(\boldsymbol{x}_t \mid \boldsymbol{x}_g)\nabla_{\boldsymbol{x}_t}\ln q(\boldsymbol{x}_t \mid \boldsymbol{x}_g)p_\theta(\boldsymbol{x}_g)\mathrm{d}\boldsymbol{x}_g}{p_\theta(\boldsymbol{x}_t)} \\
&= -\frac{\int q(\boldsymbol{x}_t \mid \boldsymbol{x}_g)\frac{\boldsymbol{x}_t - a_t\boldsymbol{x}_g}{\sigma_t^2}p_\theta(\boldsymbol{x}_g)\mathrm{d}\boldsymbol{x}_g}{p_\theta(\boldsymbol{x}_t)} \\
&= -\frac{\boldsymbol{x}_t}{\sigma_t^2} + \frac{a_t}{\sigma_t^2}\frac{\int \boldsymbol{x}_g q(\boldsymbol{x}_t \mid \boldsymbol{x}_g)p_\theta(\boldsymbol{x}_g)\mathrm{d}\boldsymbol{x}_g}{p_\theta(\boldsymbol{x}_t)} \\
&= -\frac{\boldsymbol{x}_t}{\sigma_t^2} + \frac{a_t}{\sigma^2}\int \boldsymbol{x}_g q(\boldsymbol{x}_g \mid \boldsymbol{x}_t)\mathrm{d}\boldsymbol{x}_g \\
&= -\frac{\boldsymbol{x}_t}{\sigma_t^2} + \frac{a_t}{\sigma_t^2}\mathbb{E}[\boldsymbol{x}_g \mid \boldsymbol{x}_t]
\end{aligned} \tag{27}$$

Therefore, we have

$$\mathbb{E}[\boldsymbol{x}_g \mid \boldsymbol{x}_t] = \frac{\boldsymbol{x}_t + \sigma_t^2\nabla_{\boldsymbol{x}_t}\ln q_g(\boldsymbol{x}_t)}{a_t} \tag{28}$$

which is known as the Tweedie's formula. Setting $a_t = 1$ recovers the identity presented in the main body of the paper. □

Proof of Identity 3.

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{x}_t\sim p_\theta(\boldsymbol{x}_t)}[u^T(\boldsymbol{x}_t)\nabla_{\boldsymbol{x}_t}\ln p_\theta(\boldsymbol{x}_t)]\\
&=\mathbb{E}_{\boldsymbol{x}_t\sim p_\theta(\boldsymbol{x}_t)}\left[u^T(\boldsymbol{x}_t)\frac{\nabla_{\boldsymbol{x}_t}p_\theta(\boldsymbol{x}_t)}{p_\theta(\boldsymbol{x}_t)}\right]\\
&=\int u^T(\boldsymbol{x}_t)\nabla_{\boldsymbol{x}_t}p_\theta(\boldsymbol{x}_t)\mathrm{d}\boldsymbol{x}_t\\
&=\int u^T(\boldsymbol{x}_t)\int\nabla_{\boldsymbol{x}_t}q(\boldsymbol{x}_t\,|\,\boldsymbol{x}_g)p_\theta(\boldsymbol{x}_g)\mathrm{d}\boldsymbol{x}_g\mathrm{d}\boldsymbol{x}_t\\
&=\int u^T(\boldsymbol{x}_t)\int q(\boldsymbol{x}_t\,|\,\boldsymbol{x}_g)\nabla_{\boldsymbol{x}_t}\ln q(\boldsymbol{x}_t\,|\,\boldsymbol{x}_g)p_\theta(\boldsymbol{x}_g)\mathrm{d}\boldsymbol{x}_g\mathrm{d}\boldsymbol{x}_t\\
&=\mathbb{E}_{(\boldsymbol{x}_t,\boldsymbol{x}_g)\sim q(\boldsymbol{x}_t\,|\,\boldsymbol{x}_g)p_\theta(\boldsymbol{x}_g)}[u^T(\boldsymbol{x}_t)\nabla_{\boldsymbol{x}_t}\ln q(\boldsymbol{x}_t\,|\,\boldsymbol{x}_g)].
\end{aligned}\tag{29}$$

□

Proof of Theorem 5. Expanding the L_2 norm, we have

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{x}_t\sim p_\theta(\boldsymbol{x}_t)}\|S(\boldsymbol{x}_t)-\nabla_{\boldsymbol{x}_t}\ln p_\theta(\boldsymbol{x}_t)\|_2^2\\
&=\frac{1}{\sigma_t^2}\mathbb{E}_{\boldsymbol{x}_t\sim p_\theta(\boldsymbol{x}_t)}[(\mathbb{E}[\boldsymbol{x}_0\,|\,\boldsymbol{x}_t]-\mathbb{E}[\boldsymbol{x}_g\,|\,\boldsymbol{x}_t])^T(S(\boldsymbol{x}_t)-\nabla_{\boldsymbol{x}_t}\ln p_\theta(\boldsymbol{x}_t))]\\
&=\underbrace{\frac{1}{\sigma_t^2}\mathbb{E}_{\boldsymbol{x}_t\sim p_\theta(\boldsymbol{x}_t)}\left[(\mathbb{E}[\boldsymbol{x}_0\,|\,\boldsymbol{x}_t]-\mathbb{E}[\boldsymbol{x}_g\,|\,\boldsymbol{x}_t])^TS(\boldsymbol{x}_t)\right]}_{①}-\underbrace{\frac{1}{\sigma_t^2}\mathbb{E}_{\boldsymbol{x}_t\sim p_\theta(\boldsymbol{x}_t)}[(\mathbb{E}[\boldsymbol{x}_0\,|\,\boldsymbol{x}_t]-\mathbb{E}[\boldsymbol{x}_g\,|\,\boldsymbol{x}_t])^T\nabla_{\boldsymbol{x}_t}\ln p_\theta(\boldsymbol{x}_t)]}_{②}
\end{aligned}\tag{30}$$

denote

$$\begin{aligned}
①&=\frac{1}{\sigma_t^2}\mathbb{E}_{\boldsymbol{x}_t\sim p_\theta(\boldsymbol{x}_t)}\left[\delta_{\phi,\psi^*(\theta)}(\boldsymbol{x}_t)^T(\mathbb{E}[\boldsymbol{x}_0\,|\,\boldsymbol{x}_t]-\boldsymbol{x}_t)\right]\\
&=\frac{1}{\sigma_t^2}\mathbb{E}_{\boldsymbol{x}_t\sim p_\theta(\boldsymbol{x}_t)}[\delta_{\phi,\psi^*(\theta)}(\boldsymbol{x}_t)^T\mathbb{E}[\boldsymbol{x}_0\,|\,\boldsymbol{x}_t]]-\frac{1}{\sigma_t^2}\mathbb{E}_{\boldsymbol{x}_t\sim p_\theta(\boldsymbol{x}_t)}[\delta_{\phi,\psi^*(\theta)}(\boldsymbol{x}_t)^T\boldsymbol{x}_t)]
\end{aligned}\tag{31}$$

$$\begin{aligned}
②&=\mathbb{E}_{\boldsymbol{x}_g\sim p_\theta(\boldsymbol{x}_g)}\mathbb{E}_{\boldsymbol{x}_t\sim q(\boldsymbol{x}_t\,|\,\boldsymbol{x}_g,t)}[\delta_{\phi,\psi^*(\theta)}(\boldsymbol{x}_t)^T\nabla_{\boldsymbol{x}_t}\ln q(\boldsymbol{x}_t\,|\,\boldsymbol{x}_g))]\\
&=\frac{1}{\sigma_t^2}\mathbb{E}_{\boldsymbol{x}_g\sim p_\theta(\boldsymbol{x}_g)}\mathbb{E}_{\boldsymbol{x}_t\sim q(\boldsymbol{x}_t\,|\,\boldsymbol{x}_g,t)}[\delta_{\phi,\psi^*(\theta)}(\boldsymbol{x}_t)^T(\boldsymbol{x}_g-\boldsymbol{x}_t)]\\
&=\frac{1}{\sigma_t^2}\mathbb{E}_{\boldsymbol{x}_g\sim p_\theta(\boldsymbol{x}_g)}\mathbb{E}_{\boldsymbol{x}_t\sim q(\boldsymbol{x}_t\,|\,\boldsymbol{x}_g,t)}[\delta_{\phi,\psi^*(\theta)}(\boldsymbol{x}_t)^T\boldsymbol{x}_g]-\frac{1}{\sigma_t^2}\mathbb{E}_{\boldsymbol{x}_t\sim p_\theta(\boldsymbol{x}_t)}[\delta_{\phi,\psi^*(\theta)}(\boldsymbol{x}_t)^T\boldsymbol{x}_t]
\end{aligned}\tag{32}$$

Therefore we have

$$L=①-②=\frac{1}{\sigma_t^2}\mathbb{E}_{\boldsymbol{x}_g\sim p_\theta(\boldsymbol{x}_g)}\mathbb{E}_{\boldsymbol{x}_t\sim q(\boldsymbol{x}_t\,|\,\boldsymbol{x}_g,t)}[\delta_{\phi,\psi^*(\theta)}(\boldsymbol{x}_t)^T(\mathbb{E}[\boldsymbol{x}_0\,|\,\boldsymbol{x}_t]-\boldsymbol{x}_g)]\tag{33}$$

□

D. Analytic study of the toy example

We prove the conclusions in Propositions 4 and 6. Given $p_{\text{data}}(\boldsymbol{x}_0)=\mathcal{N}(\mathbf{0},\mathbf{I})$, $p_\theta(\boldsymbol{x}_g)=\mathcal{N}(\theta,\mathbf{I})$, $q(\boldsymbol{x}_t\,|\,\boldsymbol{x}_0)=\mathcal{N}(\boldsymbol{x}_t;\boldsymbol{x}_0,\sigma_t^2\mathbf{I})$, and $q(\boldsymbol{x}_t\,|\,\boldsymbol{x}_g)=\mathcal{N}(\boldsymbol{x}_t;\boldsymbol{x}_g,\sigma_t^2\mathbf{I})$, we have $p_{\text{data}}(\boldsymbol{x}_t)=\mathcal{N}(0,(1+\sigma_t^2)\mathbf{I})$ and $p_\theta(\boldsymbol{x}_t)=\mathcal{N}(\theta,(1+\sigma_t^2)\mathbf{I})$. The optimal value of θ would be $\theta^*=0$. The score can be expressed as

$$\begin{aligned}
S(\boldsymbol{x}_t)&=\nabla_{\boldsymbol{x}_t}\ln p_{\text{data}}(\boldsymbol{x}_t)=-\frac{\boldsymbol{x}_t}{1+\sigma_t^2}\\
\nabla_{\boldsymbol{x}_t}\ln p_\theta(\boldsymbol{x}_t)&=-\frac{\boldsymbol{x}_t-\theta}{1+\sigma_t^2}.
\end{aligned}$$

Hence, the difference between the scores is $\delta_{\phi,\psi^*(\theta)}(\boldsymbol{x}_t) = -\frac{\theta}{1+\sigma_t^2}$. By applying Tweedie's formula as described in Identities 1 and 2, we obtain

$$f_\phi(\boldsymbol{x}_t, t) = \mathbb{E}[\boldsymbol{x}_0 \,|\, \boldsymbol{x}_t] = \frac{\boldsymbol{x}_t}{1+\sigma_t^2}$$
$$\mathbb{E}[\boldsymbol{x}_g \,|\, \boldsymbol{x}_t] = \boldsymbol{x}_t \frac{1}{1+\sigma_t^2} + \theta \frac{\sigma_t^2}{1+\sigma_t^2}$$

By assumption we have

$$f_\psi(\boldsymbol{x}_t, t) = \boldsymbol{x}_t \frac{1}{1+\sigma_t^2} + \psi \frac{\sigma_t^2}{1+\sigma_t^2},$$

which means $\psi^*(\theta) = \theta$, then by Equation (12) we have

$$\delta_{\phi,\psi}(\boldsymbol{x}_t) = \sigma_t^{-2}(f_\phi(\boldsymbol{x}_t, t) - f_\psi(\boldsymbol{x}_t, t)) = -\frac{\psi}{1+\sigma_t^2}. \tag{34}$$

Accordingly,

$$\hat{L}_\theta^{(1)} = \delta_{\phi,\psi}(\boldsymbol{x}_t)^T \delta_{\phi,\psi}(\boldsymbol{x}_t) = \frac{\psi^2}{(1+\sigma_t^2)^2} \tag{35}$$

Therefore, while $\hat{L}_\theta = \delta_{\phi,\psi^*(\theta)}(\boldsymbol{x}_t)^T \delta_{\phi,\psi^*(\theta)}(\boldsymbol{x}_t) = \frac{\theta^2}{(1+\sigma_t^2)^2}$ would provide useful gradient to learn θ, its naive approximation $\hat{L}_\theta^{(1)}$ could fail to provide meaningful gradient.

We can further compute

$$\begin{aligned}
\hat{L}_\theta^{(2)} &= \hat{L}_\theta^{(1)} + \frac{\delta_{\phi,\psi}(\boldsymbol{x}_t)^T (f_\psi(\boldsymbol{x}_t, t) - \boldsymbol{x}_g)}{\sigma_t^2} \\
&= \frac{\psi^2}{(1+\sigma_t^2)^2} - \frac{\psi}{\sigma_t^2(1+\sigma_t^2)} (\boldsymbol{x}_t \frac{1}{1+\sigma_t^2} + \psi \frac{\sigma_t^2}{1+\sigma_t^2} - \boldsymbol{x}_g) \\
&= \frac{\psi}{(1+\sigma_t^2)^2} \left[\boldsymbol{x}_g - \frac{\epsilon_t}{\sigma_t} \right].
\end{aligned}$$

Thus

$$\begin{aligned}
\nabla_\theta \hat{L}_\theta^{(2)} &= \frac{\psi}{(1+\sigma_t^2)^2} \nabla_\theta G_\theta(z) \\
&= -\frac{1}{1+\sigma_t^2} \delta_{\phi,\psi}(\boldsymbol{x}_t) \nabla_\theta G_\theta(z) \\
&\approx \frac{1}{1+\sigma_t^2} [\nabla_{\boldsymbol{x}_t} \ln p_\theta(\boldsymbol{x}_t) - S_\phi(\boldsymbol{x}_t)] \nabla_\theta G_\theta(z).
\end{aligned}$$

E. Training and Evaluation Details and Additional Results.

The hyperparameters tailored for our study are outlined in Table 6, with all remaining settings consistent with those in the EDM code (Karras et al., 2022). The initial development of the SiD algorithm utilized a cluster with 8 Nvidia RTX A5000 GPUs. To support a mini-batch size up to 8192 for ImageNet 64x64, we adopted the gradient accumulation strategy. Extensive evaluations across four diverse datasets were conducted using cloud computation nodes equipped with either 16 Nvidia A100-40GB GPUs, 8 Nvidia V100-16GB GPUs, or 8 Nvidia H100-80GB GPUs, with most experiments performed on Nvidia A100-40GB GPUs.

Comparisons of memory usage and per-iteration computation costs between SiD and Diff-Instruct, utilizing 16 Nvidia A100-40GB GPUs, are detailed in Table 6.

We note the time and memory costs reported in Table 6 do not include these used to evaluate the Fréchet Inception Distance (FID) of the single-step generator during the distillation process. The FID for the SiD generator, utilizing exponential

moving average (ema), was evaluated after processing each batch of 500k generator-synthesized fake images. We preserve the SiD generator that achieves the lowest FID, and to ensure accuracy, we re-evaluate it across 10 independent runs to calculate the corresponding metrics. It's worth noting that some prior studies have reported the best metric obtained across multiple independent random runs, a practice that raises concerns about reliability and reproducibility. We consciously avoid this approach in our work to ensure a more robust and credible evaluation.

Table 6. Hyperparameter settings and comparison of distillation time and memory usage between Diff-Instruct and SiD on 16 NVIDIA A100 GPUs with 40 GB of memory each.

Method	Hyperparameters	CIFAR-10 32x32	ImageNet 64x64	FFHQ 64x64	AFHQ-v2 64x64
	Batch size	256	8192	512	512
	Batch size per GPU	16	16	32	32
	# of GPUs (40G A100)	16	16	16	16
	Gradient accumulation round	1	32	1	1
	Learning rate of (ψ, θ)	1e-5	4e-6	1e-5	5e-6
	Loss scaling of (ψ, θ)	(1,100)			
	ema	0.5	2	0.5	0.5
	fp16	False	True	True	True
	Optimizer Adam (eps)	1e-8	1e-6	1e-6	1e-6
	Optimizer Adam (β_1) of θ	0	0	0.9	0
	Optimizer Adam (β_1) of ψ	0			
	Optimizer Adam (β_2)	0.999			
	α	1.0 and 1.2			
	σ_{init}	2.5			
	$t_{\max}$	800			
	augment, dropout, cres	The same as in EDM for each corresponding dataset			
Diff-Instruct	max memory in GB allocated per GPU	4.4	20.4	8.1	8.1
	max memory in GB reserved per GPU	4.7	23.0	10.8	10.8
	∼seconds per 1k images	1.4	2.8	1.1	1.1
SiD	max memory in GB allocated per GPU	7.8	31.3	17.0	17.0
	max memory in GB reserved per GPU	8.1	31.9	17.2	17.2
	∼seconds per 1k images	1.6	3.6	1.3	1.3
	∼hours per 10M (10^4k) images	4.4	10.0	3.6	3.6
	∼days per 100M (10^5k) images	1.9	4.2	1.5	1.5
	∼days per 500M (5×10^5k) images	9.3	20.8	7.5	7.5

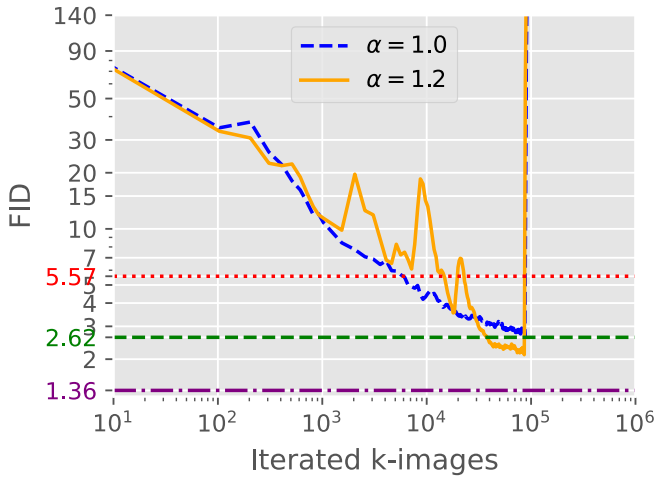

Figure 9. Analogous plot to Fig. 4 for ImageNet 64x64, where the batch size for SiD is 1024 and learning rate is 5e-6. The FID declines fast until it suddenly diverges. Increasing the batch size to 8192 and lowering the learning rate to 4e-6, as shown in Fig. 4, has alleviated the issue of sudden divergence.

Figure 10. Similar to Fig. 1, this plot showcases the SiD method's efficacy with $\alpha = 1.0$, a batch size of 8192, and a learning rate of 4e-6. The images are created using a consistent set of random noises after training the SiD generator with differing numbers of synthesized images, specifically 0, 0.2, 1, 5, 10, 20, and 50 million images. These correspond to approximately 0, 20, 120, 600, 1.2K, 2.5K, and 6.1K training iterations, respectively, displayed sequentially from left to right. The corresponding FIDs at these stages are 153.73, 54.63, 46.07, 11.02, 6.93, 4.68, and 3.34. The progression of FIDs is illustrated by the dashed blue curve in Fig. 4.

Figure 11. Analogous plot to Fig. 10 for SiD with an adjusted parameter $\alpha = 1.2$. The corresponding FIDs are 154.05, 57.63, 43.55, 16.89, 78.92, 7.45, and 3.22. The progression of FIDs is illustrated by the solid orange curve in Fig. 4.

Figure 12. All but the last subplot consist of example SiD generated images corresponding to the spikes of the solid orange curve in Fig. 4, which depicts the evolution of FIDs of SiD with $\alpha = 1.2$. These spikes are observed after processing around 10, 55, 17, 23, 73, and 88 million images. The last subplot displays SiD generated images using the generator with the lowest FID.

Figure 13. Uncoditional CIFAR-10 32X32 random images generated with SiD (FID: 1.923).

Figure 14. Label conditioning CIFAR-10 32X32 random images generated with SiD (FID: 1.710)

Figure 15. Label conditioning ImageNet 64x64 random images generated with SiD (FID: 1.524)

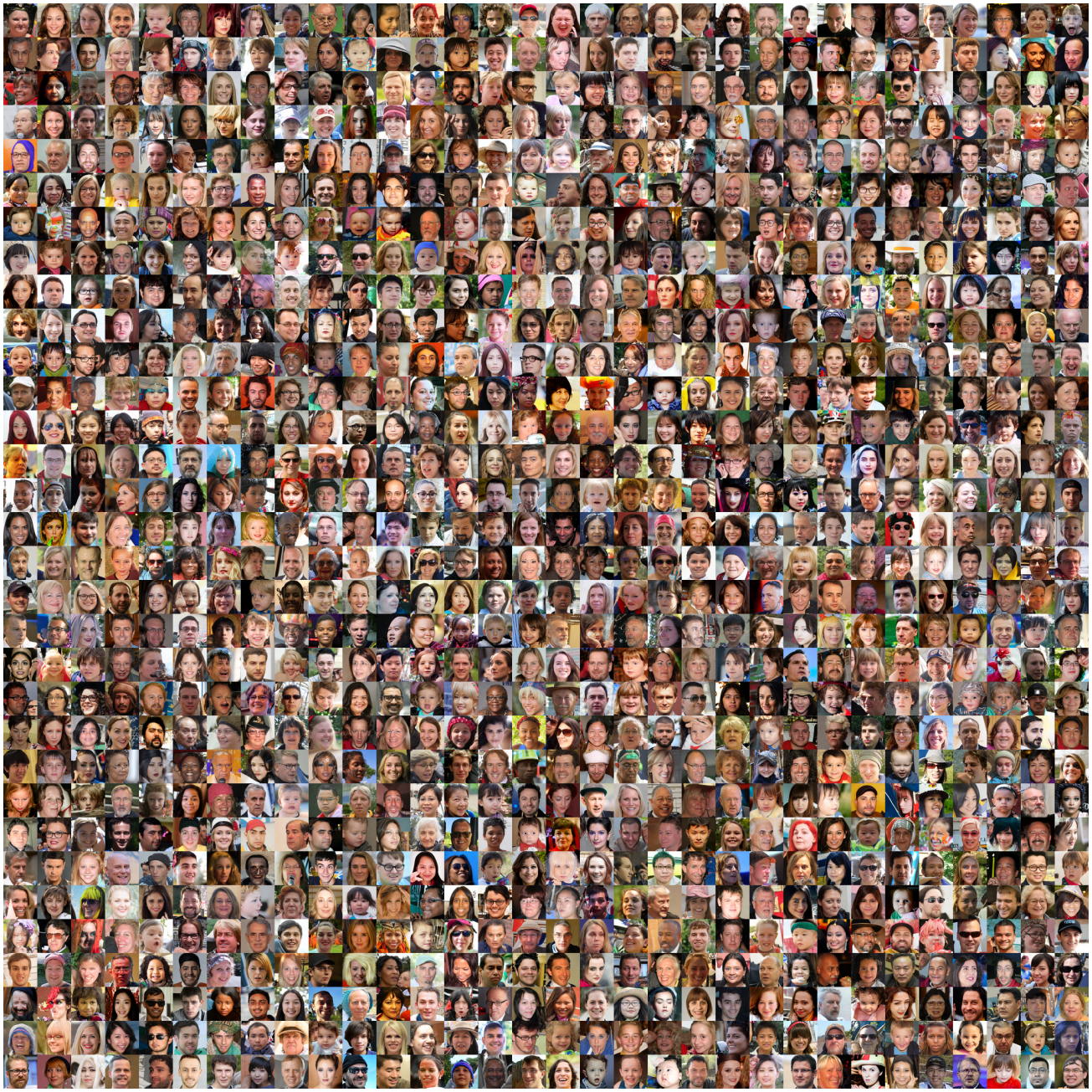

Figure 16. FFHQ 64X64 random images generated with SiD (FID: 1.550)

Figure 17. AFHQ-V2 64X64 random images generated with SiD (FID: 1.628)