Maximum Likelihood Training of Score-Based Diffusion Models

Anonymous Author(s)
Affiliation
Address
email

Abstract

Score-based diffusion models synthesize samples by reversing a stochastic process that diffuses data to noise, and are trained by minimizing a weighted combination of score matching losses. The log-likelihood of score-based models can be tractably computed through a connection to continuous normalizing flows, but log-likelihood is not directly optimized by the weighted combination of score matching losses. We show that for a specific weighting scheme, the objective upper bounds the negative log-likelihood, thus enabling approximate maximum likelihood training of score-based models. We empirically observe that maximum likelihood training consistently improves the likelihood of score-based models across multiple datasets, stochastic processes, and model architectures. Our best models achieve negative log-likelihoods of 2.74 and 3.76 bits/dim on CIFAR-10 and down-sampled ImageNet, outperforming all existing likelihood-based models.

1 Introduction

Score-based generative models [39, 40, 43] and diffusion probabilistic models [38, 18] have recently achieved state-of-the-art sample quality in a number of tasks, including image generation [43, 10], audio synthesis [4, 25, 33], and shape generation [2]. Both families of models perturb data with a sequence of noise distributions, and generate samples by learning to reverse this path from noise to data. Through stochastic calculus, these approaches can be unified into a single framework [43] which we refer to as score-based diffusion models in this paper.

The framework of score-based diffusion models [43] involves gradually diffusing the data distribution towards a given noise distribution using a stochastic differential equation (SDE), and learning the time reversal of this SDE for sample generation. Crucially, the reverse-time SDE has a closed-form expression which depends solely on a time-dependent gradient field (a.k.a., score) of the perturbed data distribution. This gradient field can be efficiently estimated by training a neural network (called a score-based model [39, 40]) with a weighted combination of score matching losses [22, 50, 41] as the objective. A key advantage of score-based diffusion models is that they can be transformed into continuous normalizing flows [CNFs, 5, 14], thus allowing tractable likelihood computation with numerical ODE solvers.

Compared to vanilla CNFs, score-based diffusion models are much more efficient to train. This is because the maximum likelihood objective for training CNFs requires running an expensive ODE solver for every optimization step, while the weighted combination of score matching losses for training score-based models does not. However, unlike maximum likelihood training, minimizing a combination of score matching losses does not necessarily lead to better likelihood values. Since better likelihoods are useful for applications including compression [20, 19, 45], semi-supervised learning [9], adversarial purification [42], and comparing against likelihood-based generative models,

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

we seek a training objective for score-based diffusion models that is as efficient as score matching but also promotes higher likelihoods.

We show that such an objective can be readily obtained through slight modification of the weighted combination of score matching losses. Our theory reveals that with a specific choice of weighting, which we term the *likelihood weighting*, the combination of score matching losses actually upper bounds the negative log-likelihood. We further prove that this upper bound becomes tight when our score-based model corresponds to the true time-dependent gradient field of a certain reverse-time SDE. Using likelihood weighting increases the variance of our objective, which we counteract by introducing a variance reduction technique based on importance sampling. Our bound is analogous to the classic evidence lower bound used for training latent-variable models in the variational autoencoding framework [24, 35], and can be viewed as a continuous-time generalization of [38].

With our likelihood weighting, we can minimize the weighted combination of score matching losses for approximate maximum likelihood training of score-based diffusion models. Compared to weightings in previous work [43], we consistently improve likelihood values across multiple datasets, model architectures, and SDEs, with slightly increased Fréchet Inception distances [16]. Moreover, our upper bound on negative log-likelihood allows training with variational dequantization [17], with which we reach negative log-likelihood of **2.74** bits/dim on CIFAR-10 [26] and **3.76** bits/dim on ImageNet 32×32 [49] in the absence of distribution augmentation [23]. Both results break existing records, and present the first instances of (continuous) normalizing flows which achieve better likelihood than state-of-the-art autoregressive models on these datasets.

2 Score-based diffusion models

Score-based diffusion models are deep generative models that smoothly transform data to noise with a diffusion process, and synthesize samples by learning and simulating the time reversal of this diffusion. The overall idea is illustrated in Fig. 1.

2.1 Diffusing data to noise with an SDE

Let $p(\mathbf{x})$ denote the unknown distribution of a dataset consisting of D-dimensional i.i.d. samples. Score-based diffusion models [43] employ a stochastic differential equation (SDE) to diffuse $p(\mathbf{x})$ towards a noise distribution. The SDEs are of the form

$$\mathrm{d}\mathbf{x} = \boldsymbol{f}(\mathbf{x}, t)\,\mathrm{d}t + g(t)\,\mathrm{d}\mathbf{w}, \tag{1}$$

where $\boldsymbol{f}(\cdot, t) : \mathbb{R}^D \to \mathbb{R}^D$ is the drift coefficient, $g(t) \in \mathbb{R}$ is the diffusion coefficient, and $\mathbf{w} \in \mathbb{R}^D$ denotes a standard Wiener process (a.k.a., Brownian motion). Intuitively, we can interpret $\mathrm{d}\mathbf{w}$ as infinitesimal Gaussian noise. The solution of an SDE is a diffusion process $\{\mathbf{x}(t)\}_{t\in[0,T]}$, where $T > 0$ is a fixed maximal time value. We let $p_t(\mathbf{x})$ denote the marginal distribution of $\mathbf{x}(t)$, and $p_{0t}(\mathbf{x}' \mid \mathbf{x})$ denote the transition distribution from $\mathbf{x}(0)$ to $\mathbf{x}(t)$. Note that by definition we always have $p_0 = p$ when using an SDE to perturb the data distribution.

The role of the SDE is to smooth the data distribution by adding noise, gradually removing structure until little of the original signal remains. In the framework of score-based diffusion models, we choose $\boldsymbol{f}(\mathbf{x}, t)$, $g(t)$, and T such that the diffusion process $\{\mathbf{x}(t)\}_{t\in[0,T]}$ approaches some analytically tractable prior distribution $\pi(\mathbf{x})$ at $t = T$, meaning $p_T(\mathbf{x}) \approx \pi(\mathbf{x})$. Three families of SDEs suitable for this task are outlined in [43], namely Variance Exploding (VE) SDEs, Variance Preserving (VP) SDEs, and subVP SDEs.

2.2 Generating samples with the reverse SDE

Sample generation in score-based diffusion models relies on time-reversal of the diffusion process. For well-behaved drift and diffusion coefficients, the forward diffusion described in Eq. (1) has an associated reverse-time diffusion process [1, 15] given by the following SDE

$$\mathrm{d}\mathbf{x} = \left[\boldsymbol{f}(\mathbf{x}, t) - g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right] \mathrm{d}t + g(t)\,\mathrm{d}\bar{\mathbf{w}}, \tag{2}$$

where $\bar{\mathbf{w}}$ is now a standard Wiener process in the reverse-time direction. Here $\mathrm{d}t$ represents an infinitesimal negative time step, meaning that the above SDE must be solved from $t = T$ to $t = 0$.

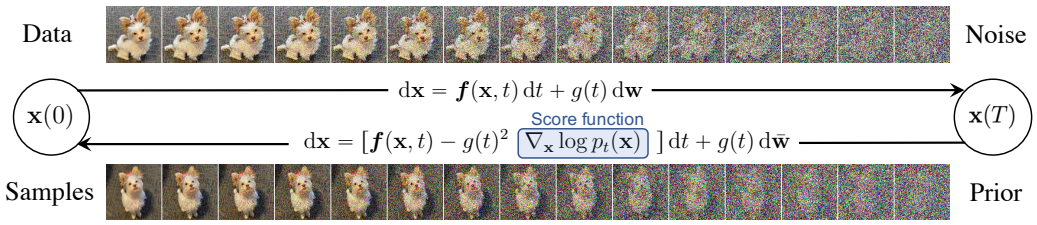

Figure 1: We can use an SDE to diffuse data to a simple noise distribution. This SDE can be reversed once we know the score of the marginal distribution at each intermediate time step, $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$.

This reverse-time SDE results in exactly the same diffusion process $\{\mathbf{x}(t)\}_{t\in[0,T]}$ as Eq. (1), assuming it is initialized with $\mathbf{x}(T) \sim p_T(\mathbf{x})$. This result allows for the construction of diffusion-based generative models, and its functional form reveals the key target for learning: the time-dependent score function $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$. Again, see Fig. 1 for a helpful visualization of this two-part formulation.

In order to estimate $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ from a given dataset, we fit the parameters of a neural network $\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)$, termed a *score-based model*, such that $\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}, t) \approx \nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ for almost all $\mathbf{x} \in \mathbb{R}^D$ and $t \in [0, T]$. Unlike many likelihood-based generative models, a score-based model does not need to satisfy the integral constraints of a density function, and is therefore much easier to parameterize. Good score-based models should keep the following least squares loss small

$$\mathcal{J}_{\mathrm{SM}}(\boldsymbol{\theta}; \lambda(\cdot)) := \frac{1}{2} \int_0^T \mathbb{E}_{p_t(\mathbf{x})}[\lambda(t)\, \|\nabla_{\mathbf{x}} \log p_t(\mathbf{x}) - \boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)\|_2^2]\, \mathrm{d}t, \tag{3}$$

where $\lambda\colon [0, T] \to \mathbb{R}_{>0}$ is a positive weighting function. The integrand features the well-known score matching [22] objective $\mathbb{E}_{p_t(\mathbf{x})}[\|\nabla_{\mathbf{x}} \log p_t(\mathbf{x}) - \boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)\|_2^2]$. We therefore refer to Eq. (3) as a weighted combination of score matching losses.

With score matching techniques [50, 41], we can estimate Eq. (3) up to an additive constant and minimize it for training score-based models. Specifically, we can use denoising score matching [50] to transform $\mathcal{J}_{\mathrm{SM}}(\boldsymbol{\theta}; \lambda(\cdot))$ into the following, which is equivalent up to a constant independent of $\boldsymbol{\theta}$:

$$\mathcal{J}_{\mathrm{DSM}}(\boldsymbol{\theta}; \lambda(\cdot)) := \frac{1}{2} \int_0^T \mathbb{E}_{p(\mathbf{x})p_{0t}(\mathbf{x}'|\mathbf{x})}[\lambda(t)\, \left\|\nabla_{\mathbf{x}'} \log p_{0t}(\mathbf{x}' \mid \mathbf{x}) - \boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}', t)\right\|_2^2]\, \mathrm{d}t. \tag{4}$$

Whenever the drift coefficient $\boldsymbol{f}_{\boldsymbol{\theta}}(\mathbf{x}, t)$ is linear in $\mathbf{x}$ (which is true for all SDEs in [43]), the transition density $p_{0t}(\mathbf{x}' \mid \mathbf{x})$ is a tractable Gaussian distribution. We can form a Monte Carlo estimate of both the time integral and expectation in $\mathcal{J}_{\mathrm{DSM}}(\boldsymbol{\theta}; \lambda(\cdot))$ with a sample $(t, \mathbf{x}, \mathbf{x}')$, where t is uniformly drawn from $[0, T]$, $\mathbf{x} \sim p(\mathbf{x})$ is a sample from the dataset, and $\mathbf{x}' \sim p_{0t}(\mathbf{x}' \mid \mathbf{x})$. The gradient $\nabla_{\mathbf{x}'} \log p_{0t}(\mathbf{x}' \mid \mathbf{x})$ can also be computed in closed form since $p_{0t}(\mathbf{x}' \mid \mathbf{x})$ is Gaussian.

After training a score-based model $\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)$ with $\mathcal{J}_{\mathrm{DSM}}(\boldsymbol{\theta}; \lambda(\cdot))$, we can plug it into the reverse-time SDE in Eq. (2). Samples are then generated by solving this reverse-time SDE with numerical SDE solvers, given an initial sample from $\pi(\mathbf{x})$ at $t = T$. Since the forward SDE Eq. (1) is designed such that $p_T(\mathbf{x}) \approx \pi(\mathbf{x})$, the reverse-time SDE will closely trace the diffusion process given by Eq. (1) in the reverse time direction, and yield an approximate data sample at $t = 0$ (as visualized in Fig. 1).

3 Likelihood of score-based diffusion models

The forward and backward diffusion processes in score-based diffusion models induce two probabilistic models for which we can define a likelihood. The first probabilistic model, denoted as $p_{\boldsymbol{\theta}}^{\mathrm{SDE}}(\mathbf{x})$, is given by the approximate reverse-time SDE constructed from our score-based model $\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)$. In particular, suppose $\{\hat{\mathbf{x}}_{\boldsymbol{\theta}}(t)\}_{t\in[0,T]}$ is a stochastic process given by

$$\mathrm{d}\hat{\mathbf{x}} = \left[\boldsymbol{f}(\hat{\mathbf{x}}, t) - g(t)^2 \boldsymbol{s}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}, t)\right] \mathrm{d}t + g(t)\, \mathrm{d}\bar{\mathbf{w}}, \quad \hat{\mathbf{x}}_{\boldsymbol{\theta}}(T) \sim \pi. \tag{5}$$

We define $p_{\boldsymbol{\theta}}^{\mathrm{SDE}}$ as the marginal distribution of $\hat{\mathbf{x}}_{\boldsymbol{\theta}}(0)$. The probabilistic model $p_{\boldsymbol{\theta}}^{\mathrm{SDE}}$ is jointly defined by the score-based model $\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)$, the prior π, plus the drift and diffusion coefficients of the forward SDE in Eq. (1). We can obtain a sample $\hat{\mathbf{x}}_{\boldsymbol{\theta}}(0) \sim p_{\boldsymbol{\theta}}^{\mathrm{SDE}}$ by numerically solving the reverse-time SDE in Eq. (5) with an initial noise vector $\hat{\mathbf{x}}_{\boldsymbol{\theta}}(T) \sim \pi$.

The other probabilistic model, denoted as $p_{\boldsymbol{\theta}}^{\text{ODE}}(\mathbf{x})$, is derived from the SDE's associated *probability flow ODE* [28, 43]. Every SDE has a corresponding probability flow ODE whose marginal distribution at each time t matches that of the SDE, so that they share the same $p_t(\mathbf{x})$ for all time. In particular, the ODE corresponding to the SDE in Eq. (1) is given by

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \boldsymbol{f}(\mathbf{x}, t) - \frac{1}{2} g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x}). \tag{6}$$

Unlike the SDEs in Eq. (1) and Eq. (2), this ODE describes fully deterministic dynamics for the process. Notably, it still features the same time-dependent score function $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$. By approximating this score function with our model $\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)$, the probability flow ODE becomes

$$\frac{\mathrm{d}\tilde{\mathbf{x}}}{\mathrm{d}t} = \boldsymbol{f}(\tilde{\mathbf{x}}, t) - \frac{1}{2} g(t)^2 \boldsymbol{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}, t). \tag{7}$$

In fact, this ODE is an instance of a continuous normalizing flow [CNF, 14], and we can quantify how the ODE dynamics transform volumes across time in exactly the same way as these traditional flow-based models [5]. Given a prior distribution $\pi(\mathbf{x})$, and a trajectory function $\tilde{\mathbf{x}}_{\boldsymbol{\theta}} \colon [0, T] \to \mathbb{R}^D$ satisfying the ODE in Eq. (7), we define $p_{\boldsymbol{\theta}}^{\text{ODE}}$ as the marginal distribution of $\tilde{\mathbf{x}}_{\boldsymbol{\theta}}(0)$ when $\tilde{\mathbf{x}}_{\boldsymbol{\theta}}(T) \sim \pi$. Similarly to $p_{\boldsymbol{\theta}}^{\text{SDE}}$, the model $p_{\boldsymbol{\theta}}^{\text{ODE}}$ is jointly defined by the score-based model $\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x})$, the prior π, and the forward SDE in Eq. (1). Leveraging the instantaneous change-of-variables formula [5], we can evaluate $\log p_{\boldsymbol{\theta}}^{\text{ODE}}(\mathbf{x})$ exactly with numerical ODE solvers. Since $p_{\boldsymbol{\theta}}^{\text{ODE}}$ is a CNF, we can generate a sample $\tilde{\mathbf{x}}_{\boldsymbol{\theta}}(0) \sim p_{\boldsymbol{\theta}}^{\text{ODE}}$ by numerically solving the ODE in Eq. (7) with an initial value $\tilde{\mathbf{x}}_{\boldsymbol{\theta}}(T) \sim \pi$.

Although computing $\log p_{\boldsymbol{\theta}}^{\text{ODE}}(\mathbf{x})$ is tractable, training $p_{\boldsymbol{\theta}}^{\text{ODE}}$ with maximum likelihood will require calling an ODE solver for every optimization step, which can be prohibitively expensive for large-scale score-based models. Unlike $p_{\boldsymbol{\theta}}^{\text{ODE}}$, we cannot evaluate $\log p_{\boldsymbol{\theta}}^{\text{SDE}}(\mathbf{x})$ exactly for an arbitrary data point $\mathbf{x}$. However, we have a lower bound on $\log p_{\boldsymbol{\theta}}^{\text{SDE}}(\mathbf{x})$ which allows both efficient evaluation and optimization, as will be shown in Section 4.2.

4 Bounding the likelihood of score-based diffusion models

Many applications benefit from models which achieve high likelihood. One example is lossless compression, where likelihood directly corresponds to the minimum expected number of bits needed to encode a message. Popular likelihood-based models such as variational autoencoders and normalizing flows have already found success in image compression [45, 19, 20]. Despite some known drawbacks [44], likelihood is also still one of the most popular metrics for evaluating and comparing generative models.

Maximizing the likelihood of score-based diffusion models can be accomplished by either maximizing the likelihood of $p_{\boldsymbol{\theta}}^{\text{SDE}}$ or $p_{\boldsymbol{\theta}}^{\text{ODE}}$. Although $p_{\boldsymbol{\theta}}^{\text{ODE}}$ is a continuous normalizing flow (CNF) and its log-likelihood is tractable, training with maximum likelihood is expensive. As mentioned already, it requires solving an ODE at every optimization step in order to evaluate the log-likelihood on a batch of training data. In contrast, training with a weighted combination of score matching losses is much more efficient, yet it does not directly promote high likelihood of either $p_{\boldsymbol{\theta}}^{\text{SDE}}$ or $p_{\boldsymbol{\theta}}^{\text{ODE}}$.

In what follows, we show that with a specific choice of the weighting function $\lambda(t)$, the combination of score matching losses $\mathcal{J}_{\text{SM}}(\boldsymbol{\theta}; \lambda(\cdot))$ actually becomes an upper bound on $D_{\text{KL}}(p \parallel p_{\boldsymbol{\theta}}^{\text{SDE}})$, and can therefore serve as an efficient proxy for maximum likelihood training. In addition, we provide a related lower bound on $\log p_{\boldsymbol{\theta}}^{\text{SDE}}(\mathbf{x})$ that can be evaluated efficiently on any individual datapoint $\mathbf{x}$.

4.1 Bounding the KL divergence with likelihood weighting

It is well-known that maximizing the log-likelihood of a probabilistic model is equivalent to minimizing the KL divergence from the data distribution to the model distribution. We show in the following theorem that for the model $p_{\boldsymbol{\theta}}^{\text{SDE}}$, this KL divergence can be upper bounded by $\mathcal{J}_{\text{SM}}(\boldsymbol{\theta}; \lambda(\cdot))$ when using the weighting function $\lambda(t) = g(t)^2$.

Theorem 1. *Let $p(\mathbf{x})$ be the data distribution, $\pi(\mathbf{x})$ be a known prior distribution, and $p_{\boldsymbol{\theta}}^{\text{SDE}}$ be defined as in Section 3. Suppose $\{\mathbf{x}(t)\}_{t \in [0,T]}$ is a stochastic process defined by the SDE in Eq. (1) with $\mathbf{x}(0) \sim p$, where the marginal distribution of $\mathbf{x}(t)$ is denoted as p_t. Under some regularity conditions detailed in Appendix A, we have*

$$D_{\text{KL}}(p \parallel p_{\boldsymbol{\theta}}^{\text{SDE}}) \leqslant \mathcal{J}_{\text{SM}}(\boldsymbol{\theta}; g(\cdot)^2) + D_{\text{KL}}(p_T \parallel \pi). \tag{8}$$

Table 1: SDEs and their corresponding weightings for score matching losses.

SDE	Formula	$\lambda(t)$ in [43]	likelihood weighting
VE	$\mathrm{d}\mathbf{x} = \sigma(t)\,\mathrm{d}\mathbf{w}$	$\sigma^2(t)$	$\sigma^2(t)$
VP	$\mathrm{d}\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\mathbf{w}$	$1 - e^{-\int_0^t \beta(s)\,\mathrm{d}s}$	$\beta(t)$
subVP	$\mathrm{d}\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}\,\mathrm{d}t + \sqrt{\beta(t)(1 - e^{-2\int_0^t \beta(s)\,\mathrm{d}s})}\,\mathrm{d}\mathbf{w}$	$(1 - e^{-\int_0^t \beta(s)\,\mathrm{d}s})^2$	$\beta(t)(1 - e^{-2\int_0^t \beta(s)\,\mathrm{d}s})$

Sketch of proof. Let $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ denote the path measures of SDEs in Eq. (1) and Eq. (5) respectively. Intuitively, $\boldsymbol{\mu}$ is the joint distribution of the diffusion process $\{\mathbf{x}(t)\}_{t\in[0,T]}$ given in Section 2.1, and $\boldsymbol{\nu}$ represents the joint distribution of the process $\{\mathbf{x}_{\boldsymbol{\theta}}(t)\}_{t\in[0,T]}$ defined in Section 3. Since we can marginalize $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ to obtain distributions p and $p_{\boldsymbol{\theta}}^{\text{SDE}}$, the data processing inequality gives $D_{\mathrm{KL}}(p \parallel p_{\boldsymbol{\theta}}^{\text{SDE}}) \leqslant D_{\mathrm{KL}}(\boldsymbol{\mu} \parallel \boldsymbol{\nu})$. From the chain rule for the KL divergence, we also have $D_{\mathrm{KL}}(\boldsymbol{\mu} \parallel \boldsymbol{\nu}) = D_{\mathrm{KL}}(p_T \parallel \pi) + \mathbb{E}_{p_T(\mathbf{z})}[D_{\mathrm{KL}}(\boldsymbol{\mu}(\cdot \mid \mathbf{x}(T) = \mathbf{z}) \parallel \boldsymbol{\nu}(\cdot \mid \mathbf{x}_{\boldsymbol{\theta}}(T) = \mathbf{z}))]$, where the KL divergence in the final term can be computed by applying Girsanov's theorem [30] to Eq. (5) and the reverse-time SDE of Eq. (1). □

When the prior distribution π is fixed, Theorem 1 guarantees that optimizing the weighted combination of score matching losses $\mathcal{J}_{\text{SM}}(\boldsymbol{\theta}; g(\cdot)^2)$ is equivalent to minimizing an upper bound on the KL divergence from the data distribution p to the model distribution $p_{\boldsymbol{\theta}}^{\text{SDE}}$. Due to well-known equivalence between minimizing KL divergence and maximizing likelihood, we have the following corollary.

Corollary 1. *Consider the same conditions and notations in Theorem 1. When π is a fixed prior distribution that does not depend on $\boldsymbol{\theta}$, we have*

$$-\mathbb{E}_{p(\mathbf{x})}[\log p_{\boldsymbol{\theta}}^{\text{SDE}}(\mathbf{x})] \leqslant \mathcal{J}_{\text{SM}}(\boldsymbol{\theta}; g(\cdot)^2) + C_1 = \mathcal{J}_{\text{DSM}}(\boldsymbol{\theta}; g(\cdot)^2) + C_2,$$

where C_1 and C_2 are constants independent of $\boldsymbol{\theta}$.

In light of the result in Corollary 1, we henceforth term $\lambda(t) = g(t)^2$ the *likelihood weighting*. The original weighting functions in [43] are inspired from earlier work such as [39, 40] and [18], which are motivated by balancing different score matching losses in the combination, and justified by empirical performance. In contrast, likelihood weighting is motivated from maximizing the likelihood of a probabilistic model induced by the diffusion process, and derived by theoretical analysis. There are three types of SDEs considered in [43]: the Variance Exploding (VE) SDE, the Variance Preserving (VP) SDE, and the subVP SDE. In Table 1, we summarize all these SDEs and contrast their original weighting functions with our likelihood weighting. For VE SDE, our likelihood weighting incidentally coincides with the original weighting used in [43], whereas for VP and subVP SDEs they differ from one another.

Theorem 1 leaves two questions unanswered. First, what are the conditions for the bound to be tight (become an equality)? Second, is there any connection between $p_{\boldsymbol{\theta}}^{\text{SDE}}$ and $p_{\boldsymbol{\theta}}^{\text{ODE}}$ under some conditions? We provide both answers in the following theorem.

Theorem 2. *Consider two continuous distributions p and q over $\mathbb{R}^D$. Let $\{\mathbf{x}(t)\}_{t\in[0,T]}$ be the diffusion process defined by the SDE in Eq. (1). We use p_t and q_t to denote the distributions of $\mathbf{x}(t)$ when $\mathbf{x}(0) \sim p$ and $\mathbf{x}(0) \sim q$ respectively. Under the conditions $q_T = \pi$ and $\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}, t) \equiv \nabla_{\mathbf{x}} \log q_t(\mathbf{x})$ for all $t \in [0, T]$, we have the following equivalence in distributions*

$$p_{\boldsymbol{\theta}}^{\text{SDE}} = p_{\boldsymbol{\theta}}^{\text{ODE}} = q. \tag{9}$$

Moreover, assuming $\log p_t(\mathbf{x})$ and $\log q_t(\mathbf{x})$ are smooth functions with at most polynomial growth at infinity, we have

$$D_{\mathrm{KL}}(p \parallel p_{\boldsymbol{\theta}}^{\text{SDE}}) = \mathcal{J}_{\text{SM}}(\boldsymbol{\theta}; g(\cdot)^2) + D_{\mathrm{KL}}(p_T \parallel \pi). \tag{10}$$

Sketch of proof. When $\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)$ matches $\nabla_{\mathbf{x}} \log q_t(\mathbf{x})$, they both represent the time-dependent score of the same stochastic process so we immediately have $p_{\boldsymbol{\theta}}^{\text{SDE}} = q$. According to the theory of probability flow ODEs, we also have $p_{\boldsymbol{\theta}}^{\text{ODE}} = q = p_{\boldsymbol{\theta}}^{\text{SDE}}$. To prove Eq. (10), we note that $D_{\mathrm{KL}}(p \parallel p_{\boldsymbol{\theta}}^{\text{SDE}}) = D_{\mathrm{KL}}(p \| q) = D_{\mathrm{KL}}(p_T \| q_T) - \int_0^T \frac{\mathrm{d}}{\mathrm{d}t} D_{\mathrm{KL}}(p_t \parallel q_t)\,\mathrm{d}t = D_{\mathrm{KL}}(p_T \| \pi) - \int_0^T \frac{\mathrm{d}}{\mathrm{d}t} D_{\mathrm{KL}}(p_t \parallel q_t)\,\mathrm{d}t$. We can now complete the proof by simplifying the integrand using the Fokker–Planck equation of p_t and q_t followed by integration by parts. □

In practice, the conditions of Theorem 2 are often violated since our score-based model $\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)$ does not exactly match the score function $\nabla_{\mathbf{x}} \log q_t(\mathbf{x})$ of some reverse-time diffusion process with initial distribution $q_T = \pi$. Therefore, although score matching with likelihood weighting performs approximate maximum likelihood training for $p_{\boldsymbol{\theta}}^{\text{SDE}}$, we emphasize that it is not theoretically guaranteed to make the likelihood of $p_{\boldsymbol{\theta}}^{\text{ODE}}$ better. That said, $p_{\boldsymbol{\theta}}^{\text{SDE}}$ will closely match $p_{\boldsymbol{\theta}}^{\text{ODE}}$ if our score-based model well-approximates the true score such that $\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}, t) \approx \nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ for all $\mathbf{x}$ and $t \in [0, T]$. Moreover, we empirically observe in our experiments (see Table 2) that training with the likelihood weighting consistently improves the likelihood of $p_{\boldsymbol{\theta}}^{\text{ODE}}$ across multiple datasets, SDEs, and model architectures.

4.2 Bounding the log-likelihood on individual datapoints

The bound in Theorem 1 is for the entire distributions of p and $p_{\boldsymbol{\theta}}^{\text{SDE}}$, but we often seek to bound the log-likelihood for an individual data point $\mathbf{x}$. In addition, $\mathcal{J}_{\text{SM}}(\boldsymbol{\theta}; \lambda(\cdot))$ in the bound is not directly computable due to the unknown quantity $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$, and can only be evaluated up to an additive constant through $\mathcal{J}_{\text{DSM}}(\boldsymbol{\theta}; \lambda(\cdot))$ (as we already discussed in Section 2.2). Therefore, the bound in Theorem 1 is only suitable for training purposes. To address these issues, we provide the following bounds for individual data points.

Theorem 3. *Let $p_{0t}(\mathbf{x}' \mid \mathbf{x})$ denote the transition distribution from $p_0(\mathbf{x})$ to $p_t(\mathbf{x})$ for the SDE in Eq. (1). With the same notations and conditions in Theorem 1, we have*

$$-\log p_{\boldsymbol{\theta}}^{\text{SDE}}(\mathbf{x}) \leqslant \mathcal{L}_{\boldsymbol{\theta}}^{SM}(\mathbf{x}) = \mathcal{L}_{\boldsymbol{\theta}}^{DSM}(\mathbf{x}), \tag{11}$$

where $\mathcal{L}_{\boldsymbol{\theta}}^{SM}(\mathbf{x})$ is defined as

$$-\mathbb{E}_{p_{0T}(\mathbf{x}'|\mathbf{x})}[\log \pi(\mathbf{x}')] + \frac{1}{2}\int_0^T \mathbb{E}_{\mathbf{x}' \sim p_{0t}(\mathbf{x}'|\mathbf{x})}\left[2g(t)^2 \nabla_{\mathbf{x}'} \cdot \boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}', t) + g(t)^2 \left\|\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}', t)\right\|_2^2 - 2\nabla_{\mathbf{x}'} \cdot \boldsymbol{f}(\mathbf{x}', t)\right] \mathrm{d}t,$$

and $\mathcal{L}_{\boldsymbol{\theta}}^{DSM}(\mathbf{x})$ is given by

$$\begin{aligned} -\mathbb{E}_{p_{0T}(\mathbf{x}'|\mathbf{x})}[\log \pi(\mathbf{x}')] &+ \frac{1}{2}\int_0^T \mathbb{E}_{p_{0t}(\mathbf{x}'|\mathbf{x})}\left[g(t)^2 \left\|\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}', t) - \nabla_{\mathbf{x}'} \log p_{0t}(\mathbf{x}' \mid \mathbf{x})\right\|_2^2\right] \mathrm{d}t \\ &- \frac{1}{2}\int_0^T \mathbb{E}_{p_{0t}(\mathbf{x}'|\mathbf{x})}\left[g(t)^2 \left\|\nabla_{\mathbf{x}'} \log p_{0t}(\mathbf{x}' \mid \mathbf{x})\right\|_2^2 - 2\nabla_{\mathbf{x}'} \cdot \boldsymbol{f}(\mathbf{x}', t)\right] \mathrm{d}t. \end{aligned}$$

Sketch of proof. For any continuous data distribution p, we have $-\mathbb{E}_{p(\mathbf{x})}[\log p_{\boldsymbol{\theta}}^{\text{SDE}}(\mathbf{x})] = D_{\text{KL}}(p \parallel p_{\boldsymbol{\theta}}^{\text{SDE}}) + \mathcal{H}(p)$, where $\mathcal{H}(p)$ denotes the differential entropy of p. The KL term can be bounded according to Theorem 1, while the differential entropy has an identity similar to Theorem 2 (see Theorem 4 in Appendix A). Combining the bound of $D_{\text{KL}}(p \parallel p_{\boldsymbol{\theta}}^{\text{SDE}})$ and the identity of $\mathcal{H}(p)$, we obtain a bound on $-\mathbb{E}_{p(\mathbf{x})}[\log p_{\boldsymbol{\theta}}^{\text{SDE}}(\mathbf{x})]$ that holds for all continuous distribution p. Removing the expectation over p on both sides then gives us a bound on $-\log p_{\boldsymbol{\theta}}^{\text{SDE}}(\mathbf{x})$ for an individual datapoint $\mathbf{x}$. We can simplify the bound to $\mathcal{L}_{\boldsymbol{\theta}}^{\text{SM}}(\mathbf{x})$ and $\mathcal{L}_{\boldsymbol{\theta}}^{\text{DSM}}(\mathbf{x})$ with similar techniques to [22] and [50]. □

We provide two equivalent bounds $\mathcal{L}_{\boldsymbol{\theta}}^{\text{SM}}(\mathbf{x})$ and $\mathcal{L}_{\boldsymbol{\theta}}^{\text{DSM}}(\mathbf{x})$. The former bears resemblance to score matching while the second resembles denoising score matching. Both admit efficient unbiased estimators when $\boldsymbol{f}(\cdot, t)$ is linear, as the time integrals and expectations in $\mathcal{L}_{\boldsymbol{\theta}}^{\text{SM}}(\mathbf{x})$ and $\mathcal{L}_{\boldsymbol{\theta}}^{\text{DSM}}(\mathbf{x})$ can be estimated by samples of the form $(t, \mathbf{x}')$, where t is uniformly sampled over $[0, T]$, and $\mathbf{x}' \sim p_{0t}(\mathbf{x}' \mid \mathbf{x})$. Since the transition distribution $p_{0t}(\mathbf{x}' \mid \mathbf{x})$ is a tractable Gaussian when $\boldsymbol{f}(\cdot, t)$ is linear, we can easily sample from it as well as evaluating $\nabla_{\mathbf{x}'} \log p_{0t}(\mathbf{x}' \mid \mathbf{x})$ for computing $\mathcal{L}_{\boldsymbol{\theta}}^{\text{DSM}}(\mathbf{x})$. Moreover, the divergences $\nabla_{\mathbf{x}} \cdot \boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)$ and $\nabla_{\mathbf{x}} \cdot \boldsymbol{f}(\mathbf{x}, t)$ in $\mathcal{L}_{\boldsymbol{\theta}}^{\text{SM}}(\mathbf{x})$ and $\mathcal{L}_{\boldsymbol{\theta}}^{\text{DSM}}(\mathbf{x})$ have efficient unbiased estimators via the Skilling–Hutchinson trick [37, 21].

We can view $\mathcal{L}_{\boldsymbol{\theta}}^{\text{DSM}}(\mathbf{x})$ as a continuous-time generalization of the evidence lower bound (ELBO) in diffusion probabilistic models [38, 18]. Our bounds in Theorem 3 are not only useful for optimizing and estimating $\log p_{\boldsymbol{\theta}}^{\text{SDE}}(\mathbf{x})$, but also for training the drift and diffusion coefficients $\boldsymbol{f}(\mathbf{x}, t)$ and $g(t)$ jointly with the score-based model $\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)$; we leave this avenue of research for future work. In addition, we can plug the bounds in Theorem 3 into any objective that involves minimizing $-\log p_{\boldsymbol{\theta}}^{\text{SDE}}(\mathbf{x})$ to obtain an efficient surrogate. Section 5.2 provides an example, where we perform variational dequantization to further improve the likelihood of score-based diffusion models.

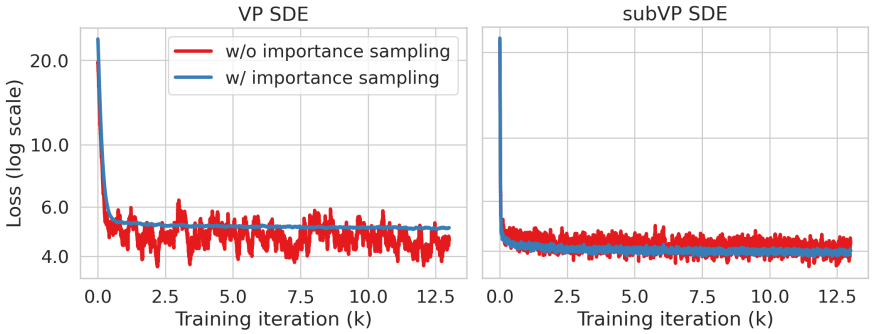

Figure 2: Learning curves with the likelihood weighting on the CIFAR-10 dataset (smoothed with exponential moving average). Importance sampling significantly reduces the loss variance.

Similar to the observation in Section 4.1, $\mathcal{L}_{\boldsymbol{\theta}}^{\mathrm{SM}}(\mathbf{x})$ and $\mathcal{L}_{\boldsymbol{\theta}}^{\mathrm{DSM}}(\mathbf{x})$ are not guaranteed to upper bound $-\log p_{\boldsymbol{\theta}}^{\mathrm{ODE}}(\mathbf{x})$. However, they should become approximate upper bounds when $\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)$ is trained sufficiently close to the ground truth. In fact, we empirically observe that $-\log p_{\boldsymbol{\theta}}^{\mathrm{ODE}}(\mathbf{x}) \leqslant \mathcal{L}_{\boldsymbol{\theta}}^{\mathrm{SM}}(\mathbf{x}) = \mathcal{L}_{\boldsymbol{\theta}}^{\mathrm{DSM}}(\mathbf{x})$ holds true for $\mathbf{x}$ sampled from the dataset in all experiments.

5 Improving the likelihood of score-based diffusion models

Our theoretical analysis implies that training with the likelihood weighting should improve the likelihood of score-based diffusion models. To verify this empirically, we test likelihood weighting with different model architectures, SDEs, and datasets. We observe that switching to likelihood weighting increases the variance of the training objective and propose to counteract it with importance sampling. We additionally combine our bound with variational dequantization [17] which narrows the gap between the likelihood of continuous and discrete probability models. All combined, we observe consistent improvement of likelihoods for both $p_{\boldsymbol{\theta}}^{\mathrm{SDE}}$ and $p_{\boldsymbol{\theta}}^{\mathrm{ODE}}$ across all settings. We term the model $p_{\boldsymbol{\theta}}^{\mathrm{ODE}}$ trained in this way *ScoreFlow*, and show that it achieves new state-of-the-art likelihoods on CIFAR-10 [26] and ImageNet 32×32 [49], outperforming even cutting-edge autoregressive models.

5.1 Variance reduction via importance sampling

As mentioned in Section 2.2, we typically use Monte Carlo sampling to approximate the time integral in $\mathcal{J}_{\mathrm{DSM}}(\boldsymbol{\theta}; \lambda(\cdot))$ during training. In particular, we first uniformly sample a time step $t \sim \mathcal{U}[0, T]$, and then use the denoising score matching loss at t as an estimate for the whole time integral. This Monte Carlo approximation is much faster than computing the time integral accurately, but introduces additional variance to the training loss.

We empirically observe that this Monte Carlo approximation suffers from a larger variance when using our likelihood weighting instead of the original weightings in [43]. Leveraging importance sampling, we propose a new Monte Carlo approximation that significantly reduces the variance of learning curves under likelihood weighting, as demonstrated in Fig. 2.

Let $\lambda(t) = \alpha(t)^2$ denote the weightings in [43] (reproduced in Table 1), and recall that our likelihood weighting is $\lambda(t) = g(t)^2$. Since $\alpha(t)^2$ empirically leads to lower variance, we can use a proposal distribution $p(t) := {}^{g(t)^2}/_{\alpha(t)^2 Z}$ to change the weighting in $\mathcal{J}_{\mathrm{DSM}}(\boldsymbol{\theta}; g(\cdot)^2)$ from $g(t)^2$ to $\alpha(t)^2$ with importance sampling, where Z is a normalizing constant that ensures $\int p(t)\,\mathrm{d}t = 1$. Specifically, for any function $h(t)$, we estimate the time integral $\int_0^T g(t)^2 h(t)\,\mathrm{d}t$ with

$$\int_0^T g(t)^2 h(t)\,\mathrm{d}t = Z \int_0^T p(t)\alpha(t)^2 h(t)\,\mathrm{d}t \approx TZ\alpha(\tilde{t})^2 h(\tilde{t}), \tag{12}$$

where $\tilde{t}$ is a sample from $p(t)$. When training score-based models with likelihood weighting, $h(t)$ corresponds to the denoising score matching loss at time t.

Ref. [29] also observes that optimizing the ELBO for diffusion probabilistic models has large variance, and propose to reduce it with importance sampling. They build their proposal distribution based on

Table 2: Negative log-likelihood (bits/dim) and sample quality (FID scores) on CIFAR-10 and ImageNet 32×32. Abbreviations: "NLL" for "negative log-likelihood"; "Uni. deq." for "Uniform dequantization"; "Var. deq." for "Variational dequantization"; "LW" for "likelihood weighting"; and "IS" for "importance sampling". Bold indicates best result in the corresponding column. Shaded rows represent models trained with both likelihood weighting and importance sampling.

Model	SDE	CIFAR-10					ImageNet 32×32				
		Uni. deq.		Var. deq.		FID↓	Uni. deq.		Var. deq.		FID↓
		NLL↓	Bound↓	NLL↓	Bound↓		NLL↓	Bound↓	NLL↓	Bound↓	
Baseline	VP	3.16	3.32	2.99	3.10	3.83	3.89	3.99	3.83	3.89	**8.30**
Baseline + LW	VP	3.06	3.23	2.88	2.99	5.39	3.90	4.00	3.83	3.89	16.25
Baseline + LW + IS	VP	2.94	3.13	2.77	2.90	6.05	3.85	3.95	**3.76**	**3.84**	10.84
Deep	VP	3.13	3.29	2.95	3.07	3.16	3.89	3.98	3.81	3.87	8.44
Deep + LW	VP	3.05	3.22	2.88	2.99	9.68	3.91	4.00	3.84	3.90	18.09
Deep + LW + IS	VP	**2.91**	3.10	**2.74**	**2.87**	5.70	3.84	3.95	3.81	3.88	11.74
Baseline	subVP	2.99	3.08	2.87	3.01	3.13	3.87	3.89	3.81	3.87	8.73
Baseline + LW	subVP	2.97	3.05	2.85	2.99	5.99	3.88	3.90	3.82	3.88	14.19
Baseline + LW + IS	subVP	2.94	3.04	2.82	2.97	5.42	3.84	**3.88**	3.78	3.86	10.43
Deep	subVP	2.96	3.05	2.84	2.98	**2.86**	3.86	**3.88**	3.80	3.86	8.85
Deep + LW	subVP	2.95	3.02	2.83	2.97	6.82	3.88	3.90	3.82	3.87	15.28
Deep + LW + IS	subVP	2.93	**3.01**	2.79	2.94	5.39	**3.82**	**3.88**	3.77	3.86	11.26

historical loss values stored at thousands of discrete time steps. Despite this similarity, our method is easier to implement without needing to maintain history, can be used for evaluation, and is particularly suited to the continuous-time setting.

We empirically test the performance of likelihood weighting and importance sampling across multiple architectures of score-based models, SDEs, and datasets. In particular, we consider DDPM++ ("Baseline" in Table 2) and DDPM++ (deep) ("Deep" in Table 2) models with VP and subVP SDEs [43] on CIFAR-10 [26] and ImageNet 32×32 [49] datasets. We omit experiments on the VE SDE since under this SDE our likelihood weighting is the same as the original weighting in [43]. For each experiment, we report $-\mathbb{E}[\log p_{\boldsymbol{\theta}}^{\text{ODE}}(\mathbf{x})]$ ("Negative log-likelihood" in Table 2), and the upper bound $\mathbb{E}[\mathcal{L}_{\boldsymbol{\theta}}^{\text{DSM}}(\mathbf{x})]$ on $-\mathbb{E}[\log p_{\boldsymbol{\theta}}^{\text{SDE}}(\mathbf{x})]$ ("Bound" in Table 2). In addition, we report FID scores [16] for samples from $p_{\boldsymbol{\theta}}^{\text{ODE}}$, produced by solving the corresponding ODE with the Dormand–Prince RK45 [13] solver. Detailed description of all our experiments can be found in Appendix B.

We summarize all results in Table 2. Our key observations are as follows:

1. Although Theorem 3 only guarantees $\mathbb{E}[\mathcal{L}_{\boldsymbol{\theta}}^{\text{DSM}}(\mathbf{x})] \geqslant -\mathbb{E}[\log p_{\boldsymbol{\theta}}^{\text{SDE}}(\mathbf{x})]$, and in general we have $p_{\boldsymbol{\theta}}^{\text{SDE}} \neq p_{\boldsymbol{\theta}}^{\text{ODE}}$, we still find that $\mathbb{E}[\mathcal{L}_{\boldsymbol{\theta}}^{\text{DSM}}(\mathbf{x})]$ ("Bound" in Table 2) is larger than $-\mathbb{E}[\log p_{\boldsymbol{\theta}}^{\text{ODE}}(\mathbf{x})]$ ("NLL" in Table 2) in all our settings.
2. When all conditions are fixed except for the weighting in the training objective, having a lower value of the bound for $p_{\boldsymbol{\theta}}^{\text{SDE}}$ always lead to a lower negative log-likelihood for $p_{\boldsymbol{\theta}}^{\text{ODE}}$.
3. With likelihood weighting, we can uniformly improve the likelihood of $p_{\boldsymbol{\theta}}^{\text{ODE}}$ and the bound of $p_{\boldsymbol{\theta}}^{\text{SDE}}$ on CIFAR-10 across model architectures and SDEs, but not on ImageNet 32×32.
4. By combining importance sampling and likelihood weighting, we are able to achieve uniformly better likelihood for $p_{\boldsymbol{\theta}}^{\text{ODE}}$ and bounds for $p_{\boldsymbol{\theta}}^{\text{SDE}}$ across model architectures, SDEs, and datasets, with only slight degradation of sample quality as measured by FID.

In summary, our experiments confirm that with importance sampling, likelihood weighting is not only effective for maximizing the lower bound for the log-likelihood of $p_{\boldsymbol{\theta}}^{\text{SDE}}$, but also improving the log-likelihood of $p_{\boldsymbol{\theta}}^{\text{ODE}}$.

5.2 Variational dequantization

Digital images are discrete data, and must be dequantized when training continuous density models like normalizing flows [11, 12] and score-based diffusion models. One popular approach to this is uniform dequantization [47, 44], where we add small uniform noise over $[0, 1)^D$ to images taking values in $\{0, 1, \cdots, 255\}^D$. As shown in [44], training a continuous model $p_{\boldsymbol{\theta}}(\mathbf{x})$ on uniformly

dequantized data implicitly maximizes a lower bound on the log-likelihood of a certain discrete model $P_{\boldsymbol{\theta}}(\mathbf{x})$. Due to the gap between $p_{\boldsymbol{\theta}}(\mathbf{x})$ and $P_{\boldsymbol{\theta}}(\mathbf{x})$, comparing the likelihood of continuous density models to models which fit discrete data directly, such as autoregressive models [49] or variational autoencoders, naturally puts the former at a disadvantage.

To minimize the gap between $p_{\boldsymbol{\theta}}(\mathbf{x})$ and $P_{\boldsymbol{\theta}}(\mathbf{x})$, ref. [17] proposes variational dequantization, where a separate normalizing flow model $q_{\boldsymbol{\phi}}(\mathbf{u} \mid \mathbf{x})$ is trained to produce the dequantization noise by optimizing the following objective

$$\max_{\boldsymbol{\phi}} \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}\mathbb{E}_{\mathbf{u}\sim q_{\boldsymbol{\phi}}(\cdot|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}+\mathbf{u}) - \log q_{\boldsymbol{\phi}}(\mathbf{u} \mid \mathbf{x})]. \tag{13}$$

This approach improved the likelihoods of normalizing flows. Since score-based diffusion models are also continuous normalizing flows, variational dequantization may improve their likelihoods.

Leveraging the bound $-\log p_{\boldsymbol{\theta}}^{\text{SDE}}(\mathbf{x}) \leqslant \mathcal{L}_{\boldsymbol{\theta}}^{\text{DSM}}(\mathbf{x})$, we can train a variational dequantization model $q_{\boldsymbol{\phi}}(\mathbf{u} \mid \mathbf{x})$ for improved maximum likelihood training of score-based diffusion models. The objective is obtained from a simple modification to Eq. (13), given by

$$\max_{\boldsymbol{\phi}} \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}\mathbb{E}_{\mathbf{u}\sim q_{\boldsymbol{\phi}}(\cdot|\mathbf{x})}[-\mathcal{L}_{\boldsymbol{\theta}}^{\text{DSM}}(\mathbf{x}+\mathbf{u}) - \log q_{\boldsymbol{\phi}}(\mathbf{u} \mid \mathbf{x})]. \tag{14}$$

Following [17], we use a modified Flow++ as the dequantization model, and optimize $\boldsymbol{\phi}$ via the reparameterization trick [24]. During training, we estimate the time integrals in $\mathcal{L}_{\boldsymbol{\theta}}^{\text{DSM}}(\mathbf{x}+\mathbf{u})$ with importance sampling (Section 5.1). All experimental details are given in Appendix B.

As shown in Table 2, training a variational dequantizer $q_{\boldsymbol{\phi}}(\mathbf{u} \mid \mathbf{x})$ by Eq. (14) uniformly improves both the bound for $p_{\boldsymbol{\theta}}^{\text{SDE}}$ and the negative log-likelihood (NLL) of $p_{\boldsymbol{\theta}}^{\text{ODE}}$ in all settings, regardless of how the original score-based models are trained. We term $p_{\boldsymbol{\theta}}^{\text{ODE}}$ a *ScoreFlow* when its corresponding score-based model $\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)$ is trained with likelihood weighting, importance sampling, and variational dequantization combined. It is a continuous normalizing flow, but is parameterized by a score-based model and trained in a more efficient way. With variational dequantization, we show ScoreFlows obtain state-of-the-art NLLs of 2.74 bits/dim on CIFAR-10 and 3.76 bits/dim on ImageNet 32×32. This establishes the new state-of-the-art on both datasets, as shown in Table 3, and is also the first time that a flow-based model outperforms the best-performing autoregressive models in terms of log-likelihood.

Table 3: NLLs on CIFAR-10 and ImageNet 32x32.

Model	CIFAR-10	ImageNet
FFJORD [14]	3.40	-
Flow++ [17]	3.08	3.86
Gated PixelCNN [31]	3.03	3.83
VFlow [3]	2.98	3.83
PixelCNN++ [36]	2.92	-
NVAE [48]	2.91	3.92
Image Transformer [32]	2.90	3.77
Very Deep VAE [7]	2.87	3.80
PixelSNAIL [6]	2.85	3.80
δ-VAE [34]	2.83	3.77
Sparse Transformer [8]	2.80	-
ScoreFlow (Ours)	**2.74**	**3.76**

6 Conclusion

We propose an efficient training objective for approximate maximum likelihood training of score-based diffusion models. Our theoretical analysis shows that the weighted combination of score matching losses upper bounds the negative log-likelihood when using a particular weighting function which we term the likelihood weighting. By minimizing this upper bound, we consistently improve the likelihood of score-based diffusion models across multiple model architectures, SDEs, and datasets. When combined with variational dequantization, we achieve new state-of-the-art likelihoods on CIFAR-10 and ImageNet 32×32, outperforming all existing likelihood-based models.

Our upper bound is analogous to the evidence lower bound commonly used for training variational autoencoders. Aside from promoting higher likelihood, the bound can be combined with other objectives that depend on the negative log-likelihood, and also enables joint training of the forward and backward SDEs, which we leave as a future research direction. Our results suggest that score-based diffusion models are competitive alternatives to continuous normalizing flows which enjoy the same tractable likelihood computation but with more efficient maximum likelihood training.

References

[1] B. D. Anderson. Reverse-Time Diffusion Equation Models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

[2] R. Cai, G. Yang, H. Averbuch-Elor, Z. Hao, S. Belongie, N. Snavely, and B. Hariharan. Learning Gradient Fields for Shape Generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[3] J. Chen, C. Lu, B. Chenli, J. Zhu, and T. Tian. Vflow: More expressive generative flows with variational data augmentation. In *International Conference on Machine Learning*, pages 1660–1669. PMLR, 2020.

[4] N. Chen, Y. Zhang, H. Zen, R. J. Weiss, M. Norouzi, and W. Chan. WaveGrad: Estimating Gradients for Waveform Generation. *arXiv preprint arXiv:2009.00713*, 2020.

[5] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural Ordinary Differential Equations. In *Advances in neural information processing systems*, pages 6571–6583, 2018.

[6] X. Chen, N. Mishra, M. Rohaninejad, and P. Abbeel. Pixelsnail: An improved autoregressive generative model. In *International Conference on Machine Learning*, pages 864–872. PMLR, 2018.

[7] R. Child. Very deep VAEs generalize autoregressive models and can outperform them on images. In *International Conference on Learning Representations*, 2021.

[8] R. Child, S. Gray, A. Radford, and I. Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

[9] Z. Dai, Z. Yang, F. Yang, W. W. Cohen, and R. Salakhutdinov. Good semi-supervised learning that requires a bad GAN. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6513–6523, 2017.

[10] P. Dhariwal and A. Nichol. Diffusion models beat GANs on image synthesis. *arXiv preprint arXiv:2105.05233*, 2021.

[11] L. Dinh, D. Krueger, and Y. Bengio. NICE: Non-Linear Independent Components Estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[12] L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using real NVP. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[13] J. R. Dormand and P. J. Prince. A family of embedded runge-kutta formulae. *Journal of computational and applied mathematics*, 6(1):19–26, 1980.

[14] W. Grathwohl, R. T. Q. Chen, J. Bettencourt, I. Sutskever, and D. Duvenaud. FFJORD: Free-form Continuous Dynamics for Scalable Reversible Generative Models. In *International Conference on Learning Representations*, 2019.

[15] U. G. Haussmann and E. Pardoux. Time reversal of diffusions. *The Annals of Probability*, pages 1188–1205, 1986.

[16] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6626–6637, 2017.

[17] J. Ho, X. Chen, A. Srinivas, Y. Duan, and P. Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *International Conference on Machine Learning*, pages 2722–2730. PMLR, 2019.

[18] J. Ho, A. Jain, and P. Abbeel. Denoising Diffusion Probabilistic Models. *Advances in Neural Information Processing Systems*, 33, 2020.

[19] J. Ho, E. Lohn, and P. Abbeel. Compression with flows via local bits-back coding. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3874–3883, 2019.

[20] E. Hoogeboom, J. W. T. Peters, R. van den Berg, and M. Welling. Integer discrete flows and lossless compression. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 12134–12144, 2019.

[21] M. F. Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989.

[22] A. Hyvärinen. Estimation of Non-Normalized Statistical Models by Score Matching. *Journal of Machine Learning Research*, 6(Apr):695–709, 2005.

[23] H. Jun, R. Child, M. Chen, J. Schulman, A. Ramesh, A. Radford, and I. Sutskever. Distribution augmentation for generative modeling. In *International Conference on Machine Learning*, pages 5006–5019. PMLR, 2020.

[24] D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations*, 2014.

[25] Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro. DiffWave: A Versatile Diffusion Model for Audio Synthesis. *arXiv preprint arXiv:2009.09761*, 2020.

[26] A. Krizhevsky, V. Nair, and G. Hinton. The CIFAR-10 Dataset. *online: http://www. cs. toronto. edu/kriz/cifar. html*, 55, 2014.

[27] X. Li, T.-K. L. Wong, R. T. Q. Chen, and D. Duvenaud. Scalable Gradients for Stochastic Differential Equations. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, 2020.

[28] D. Maoutsa, S. Reich, and M. Opper. Interacting particle solutions of Fokker–Planck equations through gradient-log-density estimation. *arXiv preprint arXiv:2006.00702*, 2020.

[29] A. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. *arXiv preprint arXiv:2102.09672*, 2021.

[30] B. Oksendal. *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013.

[31] A. v. d. Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu. Conditional image generation with pixelcnn decoders. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 4797–4805, 2016.

[32] N. Parmar, A. Vaswani, J. Uszkoreit, L. Kaiser, N. Shazeer, A. Ku, and D. Tran. Image transformer. In *International Conference on Machine Learning*, pages 4055–4064. PMLR, 2018.

[33] V. Popov, I. Vovk, V. Gogoryan, T. Sadekova, and M. Kudinov. Grad-TTS: A diffusion probabilistic model for text-to-speech. *arXiv preprint arXiv:2105.06337*, 2021.

[34] A. Razavi, A. van den Oord, B. Poole, and O. Vinyals. Preventing posterior collapse with delta-vaes. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[35] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32, 2014.

[36] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[37] J. Skilling. The eigenvalues of mega-dimensional matrices. In *Maximum Entropy and Bayesian Methods*, pages 455–466. Springer, 1989.

[38] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep Unsupervised Learning Using Nonequilibrium Thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265, 2015.

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

[40] Y. Song and S. Ermon. Improved Techniques for Training Score-Based Generative Models. *Advances in Neural Information Processing Systems*, 33, 2020.

[41] Y. Song, S. Garg, J. Shi, and S. Ermon. Sliced Score Matching: A Scalable Approach to Density and Score Estimation. In *Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence*, page 204, 2019.

[42] Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. In *International Conference on Learning Representations*, 2018.

[43] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-Based Generative Modeling Through Stochastic Differential Equations. In *International Conference on Learning Representations*, 2021.

[44] L. Theis, A. van den Oord, and M. Bethge. A note on the evaluation of generative models. In Y. Bengio and Y. LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[45] J. Townsend, T. Bird, and D. Barber. Practical lossless compression with latent variables using bits back coding. In *International Conference on Learning Representations*, 2019.

[46] B. Tzen and M. Raginsky. Neural Stochastic Differential Equations: Deep Latent Gaussian Models in the Diffusion Limit. *arXiv:1905.09883*, 2019.

[47] B. Uria, I. Murray, and H. Larochelle. Rnade: the real-valued neural autoregressive density-estimator. In *Proceedings of the 26th International Conference on Neural Information Processing Systems-Volume 2*, pages 2175–2183, 2013.

[48] A. Vahdat and J. Kautz. NVAE: A deep hierarchical variational autoencoder. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[49] A. Van den Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. In *International Conference on Machine Learning*, pages 1747–1756. PMLR, 2016.

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

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] The limitations of our theory are discussed inline in Section 4.
   (c) Did you discuss any potential negative societal impacts of your work? [No] We tackle a common task in machine learning and our methods *per se* do not add new negative societal impacts.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] The full set of assumptions are in Appendix A.
   (b) Did you include complete proofs of all theoretical results? [Yes] The full proofs are all provided in Appendix A.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] Code will be released upon publication.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] All experimental details are in Appendix B
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Training our models is expensive and we do not have enough resource/time for multiple repetitions of our experiments.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Compute and resource types are given in Appendix B
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We cited the creators of CIFAR-10 and down-sampled ImageNet.
   (b) Did you mention the license of the assets? [No] Licenses are standard and can be found online.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No] Code and checkpoint will be released upon publication.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] All datasets used in our work are publicly available.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]