Variational Diffusion Models

Diederik P. Kingma*, **Tim Salimans***, **Ben Poole, Jonathan Ho**
Google Research

Abstract

Diffusion-based generative models have demonstrated a capacity for perceptually impressive synthesis, but can they also be great likelihood-based models? We answer this in the affirmative, and introduce a family of diffusion-based generative models that obtain state-of-the-art likelihoods on standard image density estimation benchmarks. Unlike other diffusion-based models, our method allows for efficient optimization of the noise schedule jointly with the rest of the model. We show that the variational lower bound (VLB) simplifies to a remarkably short expression in terms of the signal-to-noise ratio of the diffused data, thereby improving our theoretical understanding of this model class. Using this insight, we prove an equivalence between several models proposed in the literature. In addition, we show that the continuous-time VLB is invariant to the noise schedule, except for the signal-to-noise ratio at its endpoints. This enables us to learn a noise schedule that minimizes the variance of the resulting VLB estimator, leading to faster optimization. Combining these advances with architectural improvements, we obtain state-of-the-art likelihoods on image density estimation benchmarks, outperforming autoregressive models that have dominated these benchmarks for many years, with often significantly faster optimization. In addition, we show how to use the model as part of a bits-back compression scheme, and demonstrate lossless compression rates close to the theoretical optimum.

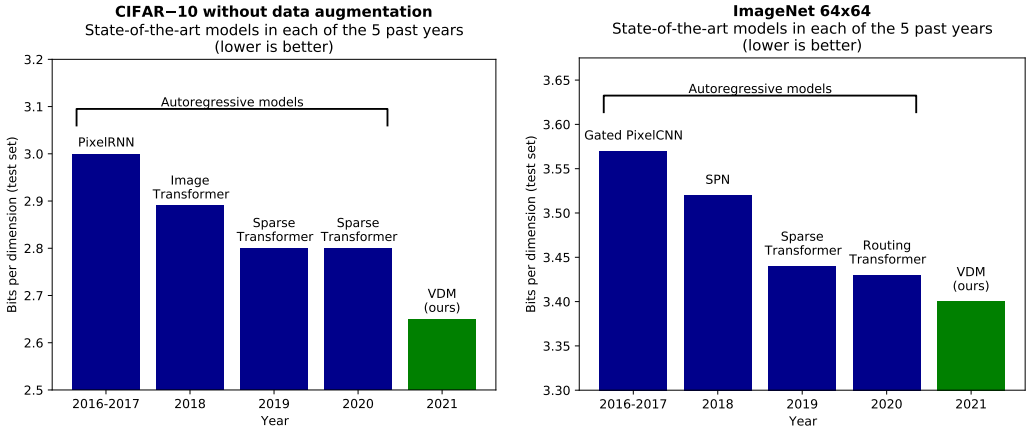
1 Introduction

Likelihood-based generative modeling is a central task in machine learning that is the basis for a wide range of applications ranging from speech synthesis [Oord et al., 2016], to translation [Sutskever et al., 2014], to compression [MacKay, 2003], to many others. Autoregressive models have long been the dominant model class on this task due to their tractable likelihood and expressivity, as shown in Figure 1. Diffusion models have recently shown impressive results in image [Ho et al., 2020, Song et al., 2021b, Nichol and Dhariwal, 2021] and audio generation [Kong et al., 2020, Chen et al., 2020] in terms of perceptual quality, but have yet to match autoregressive models on density estimation benchmarks. In this paper we make several technical contributions that allow diffusion models to challenge the dominance of autoregressive models in this domain. Our main contributions are as follows:

- We introduce a flexible family of diffusion-based generative models that achieve new state-of-the-art log-likelihoods on standard image density estimation benchmarks (CIFAR-10 and ImageNet). This is enabled by incorporating Fourier features into the diffusion model and using a learnable specification of the diffusion process, among other modeling contributions.
- We improve our theoretical understanding of density modeling using diffusion models by analyzing their variational lower bound (VLB), deriving a remarkably simple expression in terms of the signal-to-noise ratio of the diffusion process. This result delivers new insight

* Equal contribution.

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.



(a) CIFAR-10 without data augmentation

(b) ImageNet 64x64

Figure 1: Autoregressive generative models were long dominant in standard image density estimation benchmarks. In contrast, we propose a family of diffusion-based generative models, *Variational Diffusion Models* (VDMs), that outperforms contemporary autoregressive models in these benchmarks. See Table 1 for more results and comparisons.

into the model class: for the continuous-time (infinite-depth) setting we prove a novel invariance of the generative model and its VLB to the specification of the diffusion process, and we show that various diffusion models from the literature are equivalent up to a trivial time-dependent rescaling of the data.

2 Related work

Our work builds on diffusion probabilistic models (DPMs) [Sohl-Dickstein et al., 2015], or *diffusion models* in short. DPMs can be viewed as a type of variational autoencoder (VAE) [Kingma and Welling, 2013, Rezende et al., 2014], whose structure and loss function allows for efficient training of arbitrarily deep models. Interest in diffusion models has recently reignited due to their impressive image generation results [Ho et al., 2020, Song and Ermon, 2020].

Ho et al. [2020] introduced a number of model innovations to the original DPM, with impressive results on image generation quality benchmarks. They showed that the VLB objective, for a diffusion model with discrete time and diffusion variances shared across input dimensions, is equivalent to multi-scale denoising score matching, up to particular weightings per noise scale. Further improvements were proposed by Nichol and Dhariwal [2021], resulting in better log-likelihood scores. Gao et al. [2020] show how diffusion can also be used to efficiently optimize energy-based models (EBMs) towards a close approximation of the log-likelihood objective, resulting in high-fidelity samples even after long MCMC chains.

Song and Ermon [2019] first proposed learning generative models through a multi-scale denoising score matching objective, with improved methods in Song and Ermon [2020]. This was later extended to continuous-time diffusion with novel sampling algorithms based on reversing the diffusion process [Song et al., 2021b].

Concurrent to our work, Song et al. [2021a], Huang et al. [2021], and Vahdat et al. [2021] also derived variational lower bounds to the data likelihood under a continuous-time diffusion model. Where we consider the infinitely deep limit of a standard VAE, Song et al. [2021a] and Vahdat et al. [2021] present different derivations based on stochastic differential equations. Huang et al. [2021] considers both perspectives and discusses the similarities between the two approaches. An advantage of our analysis compared to these other works is that we present an intuitive expression of the VLB in terms of the signal-to-noise ratio of the diffused data, which then leads to new results on the invariance of the generative model and its VLB to the specification of the diffusion process. We empirically compare to these works, as well as others, in Table 1.

Previous approaches to diffusion probabilistic models fixed the diffusion process; in contrast optimize the diffusion process parameters jointly with the rest of the model. This turns the model into a type of VAE [Kingma and Welling, 2013, Rezende et al., 2014]. This is enabled by directly parameterizing the mean and variance of the marginal $q(\mathbf{z}_t|\mathbf{z}_0)$, where previous approaches instead parameterized the individual diffusion steps $q(\mathbf{z}_{t+\epsilon}|\mathbf{z}_t)$. In addition, our denoising models include several architecture changes, the most important of which is the use of Fourier features, which enable us to reach much better likelihoods than previous diffusion probabilistic models.

3 Model

We will focus on the most basic case of generative modeling, where we have a dataset of observations of $\mathbf{x}$, and the task is to estimate the marginal distribution $p(\mathbf{x})$. As with most generative models, the described methods can be extended to the case of multiple observed variables, and/or the task of estimating conditional densities $p(\mathbf{x}|\mathbf{y})$. The proposed latent-variable model consists of a diffusion process (Section 3.1) that we invert to obtain a hierarchical generative model (Section 3.3). As we will show, the model choices below result in a surprisingly simple variational lower bound (VLB) of the marginal likelihood, which we use for optimization of the parameters.

3.1 Forward time diffusion process

Our starting point is a Gaussian diffusion process that begins with the data $\mathbf{x}$, and defines a sequence of increasingly noisy versions of $\mathbf{x}$ which we call the *latent variables* $\mathbf{z}_t$, where t runs from $t = 0$ (least noisy) to $t = 1$ (most noisy). The distribution of latent variable $\mathbf{z}_t$ conditioned on $\mathbf{x}$, for any $t \in [0, 1]$ is given by:

$$q(\mathbf{z}_t|\mathbf{x}) = \mathcal{N}\left(\alpha_t \mathbf{x}, \sigma_t^2 \mathbf{I}\right), \tag{1}$$

where α_t and σ_t^2 are strictly positive scalar-valued functions of t. Furthermore, let us define the *signal-to-noise ratio* (SNR):

$$\mathrm{SNR}(t) = \alpha_t^2/\sigma_t^2. \tag{2}$$

We assume that the $\mathrm{SNR}(t)$ is strictly monotonically decreasing in time, i.e. that $\mathrm{SNR}(t) < \mathrm{SNR}(s)$ for any $t > s$. This formalizes the notion that the $\mathbf{z}_t$ is increasingly noisy as we go forward in time. We also assume that both α_t and σ_t^2 are smooth, such that their derivatives with respect to time t are finite. This diffusion process specification includes the *variance-preserving* diffusion process as used by [Sohl-Dickstein et al., 2015, Ho et al., 2020] as a special case, where $\alpha_t = \sqrt{1 - \sigma_t^2}$. Another special case is the *variance-exploding* diffusion process as used by [Song and Ermon, 2019, Song et al., 2021b], where $\alpha_t^2 = 1$. In experiments, we use the variance-preserving version.

The distributions $q(\mathbf{z}_t|\mathbf{z}_s)$ for any $t > s$ are also Gaussian, and given in Appendix A. The joint distribution of latent variables $(\mathbf{z}_s, \mathbf{z}_t, \mathbf{z}_u)$ at any subsequent timesteps $0 \leq s < t < u \leq 1$ is Markov: $q(\mathbf{z}_u|\mathbf{z}_t, \mathbf{z}_s) = q(\mathbf{z}_u|\mathbf{z}_t)$. Given the distributions above, it is relatively straightforward to verify through Bayes rule that $q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x})$, for any $0 \leq s < t \leq 1$, is also Gaussian. This distribution is also given in Appendix A.

3.2 Noise schedule

In previous work, the noise schedule has a fixed form (see Appendix H, Fig. 4a). In contrast, we learn this schedule through the parameterization

$$\sigma_t^2 = \mathrm{sigmoid}(\gamma_{\boldsymbol{\eta}}(t)) \tag{3}$$

where $\gamma_{\boldsymbol{\eta}}(t)$ is a monotonic neural network with parameters $\boldsymbol{\eta}$, as detailed in Appendix H.

Motivated by the equivalence discussed in Section 5.1, we use $\alpha_t = \sqrt{1 - \sigma_t^2}$ in our experiments for both the discrete-time and continuous-time models, i.e. variance-preserving diffusion processes. It is straightforward to verify that α_t^2 and $\mathrm{SNR}(t)$, as a function of $\gamma_{\boldsymbol{\eta}}(t)$, then simplify to:

$$\alpha_t^2 = \mathrm{sigmoid}(-\gamma_{\boldsymbol{\eta}}(t)) \tag{4}$$

$$\mathrm{SNR}(t) = \exp(-\gamma_{\boldsymbol{\eta}}(t)) \tag{5}$$

3.3 Reverse time generative model

We define our generative model by inverting the diffusion process of Section 3.1, yielding a hierarchical generative model that samples a sequence of latents $\mathbf{z}_t$, with time running backward from $t = 1$ to $t = 0$. We consider both the case where this sequence consists of a finite number of steps T, as well as a continuous time model corresponding to $T \to \infty$. We start by presenting the discrete-time case.

Given finite T, we discretize time uniformly into T timesteps (segments) of width $\tau = 1/T$. Defining $s(i) = (i-1)/T$ and $t(i) = i/T$, our hierarchical generative model for data $\mathbf{x}$ is then given by:

$$p(\mathbf{x}) = \int_{\mathbf{z}} p(\mathbf{z}_1) p(\mathbf{x}|\mathbf{z}_0) \prod_{i=1}^{T} p(\mathbf{z}_{s(i)}|\mathbf{z}_{t(i)}). \tag{6}$$

With the variance preserving diffusion specification and sufficiently small $\mathrm{SNR}(1)$, we have that $q(\mathbf{z}_1|\mathbf{x}) \approx \mathcal{N}(\mathbf{z}_1; 0, \mathbf{I})$. We therefore model the marginal distribution of $\mathbf{z}_1$ as a spherical Gaussian:

$$p(\mathbf{z}_1) = \mathcal{N}(\mathbf{z}_1; 0, \mathbf{I}). \tag{7}$$

We wish to choose a model $p(\mathbf{x}|\mathbf{z}_0)$ that is close to the unknown $q(\mathbf{x}|\mathbf{z}_0)$. Let x_i and $z_{0,i}$ be the i-th elements of $\mathbf{x}, \mathbf{z}_0$, respectively. We then use a factorized distribution of the form:

$$p(\mathbf{x}|\mathbf{z}_0) = \prod_i p(x_i|z_{0,i}), \tag{8}$$

where we choose $p(x_i|z_{0,i}) \propto q(z_{0,i}|x_i)$, which is normalized by summing over all possible discrete values of x_i (256 in the case of 8-bit image data). With sufficiently large $\mathrm{SNR}(0)$, this becomes a very close approximation to the true $q(\mathbf{x}|\mathbf{z}_0)$, as the influence of the unknown data distribution $q(\mathbf{x})$ is overwhelmed by the likelihood $q(\mathbf{z}_0|\mathbf{x})$. Finally, we choose the conditional model distributions as

$$p(\mathbf{z}_s|\mathbf{z}_t) = q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x} = \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)), \tag{9}$$

i.e. the same as $q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x})$, but with the original data $\mathbf{x}$ replaced by the output of a *denoising model* $\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)$ that predicts $\mathbf{x}$ from its noisy version $\mathbf{z}_t$. Note that in practice we pamaterize the denoising model as a function of a *noise prediction model* (Section 3.4), bridging the gap with previous work on diffusion models [Ho et al., 2020]. The means and variances of $p(\mathbf{z}_s|\mathbf{z}_t)$ are given in Appendix A.

3.4 Noise prediction model and Fourier features

We parameterize the denoising model in terms of a *noise prediction model* $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)$:

$$\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t) = (\mathbf{z}_t - \sigma_t \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t))/\alpha_t, \tag{10}$$

where $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)$ is parameterized as a neural network. The noise prediction models we use in experiments closely follow Ho et al. [2020], except that they process the data solely at the original resolution. The exact parameterization of the noise prediction model and noise schedule is discussed in Appendix B.

Prior work on diffusion models has mainly focused on the perceptual quality of generated samples, which emphasizes coarse scale patterns and global consistency of generated images. Here, we optimize for likelihood, which is sensitive to fine scale details and exact values of individual pixels. To capture the fine scale details of the data, we propose adding a set of *Fourier features* to the input of our noise prediction model. Let $\mathbf{x}$ be the original data, scaled to the range $[-1, 1]$, and let $\mathbf{z}$ be the resulting latent variable, with similar magnitudes. We then append channels $\sin(2^n \pi \mathbf{z})$ and $\cos(2^n \pi \mathbf{z})$, where n runs over a range of integers $\{n_{min}, ..., n_{max}\}$. These features are high frequency periodic functions that amplify small changes in the input data $\mathbf{z}_t$; see Appendix C for further details. Including these features in the input of our denoising model leads to large improvements in likelihood as demonstrated in Section 6 and Figure 5, especially when combined with a learnable SNR function. We did not observe such improvements when incorporating Fourier features into autoregressive models.

3.5 Variational lower bound

We optimize the parameters towards the variational lower bound of the marginal likelihood, also called the variational lower bound (VLB), which is given by

$$-\log p(\mathbf{x}) \leq -\mathrm{VLB}(\mathbf{x}) = \underbrace{D_{KL}(q(\mathbf{z}_1|\mathbf{x})||p(\mathbf{z}_1))}_{\text{Prior loss}} + \underbrace{\mathbb{E}_{q(\mathbf{z}_0|\mathbf{x})}\left[-\log p(\mathbf{x}|\mathbf{z}_0)\right]}_{\text{Reconstruction loss}} + \underbrace{\mathcal{L}_T(\mathbf{x})}_{\text{Diffusion loss}}. \tag{11}$$

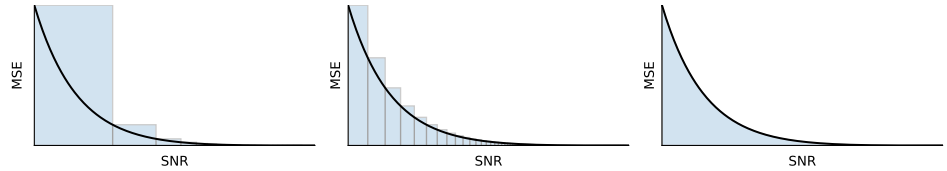


Figure 2: Illustration of the diffusion loss with few segments T (left), more segments T (middle), and infinite segments T (continuous time, right). The continuous-time loss (Equation 18) is an integral of mean squared error (MSE) over SNR, here visualized as a black curve. The black curve is strictly decreasing when the model is sufficiently well trained, so the discrete-time loss (Equation 13) is an upper bound (an upper Riemann sum approximation) of this integral that becomes better when segments are added.

The prior loss and reconstruction loss can be (stochastically and differentiably) estimated using standard techniques; see [Kingma and Welling, 2013]. The diffusion loss, $\mathcal{L}_T(\mathbf{x})$, is more complicated, and depends on the hyperparameter T that determines the depth of the generative model.

4 Discrete-time model

In the case of finite T, using $s(i) = (i-1)/T$, $t(i) = i/T$, the diffusion loss is:

$$\mathcal{L}_T(\mathbf{x}) = \sum_{i=1}^{T} \mathbb{E}_{q(\mathbf{z}_{t(i)}|\mathbf{x})} D_{KL}[q(\mathbf{z}_{s(i)}|\mathbf{z}_{t(i)}, \mathbf{x})||p(\mathbf{z}_{s(i)}|\mathbf{z}_{t(i)})]. \tag{12}$$

In appendix E we show that this expression simplifies considerably, yielding:

$$\mathcal{L}_T(\mathbf{x}) = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}), i\sim U\{1,T\}} \left[\frac{T}{2} \left(\mathrm{SNR}(s) - \mathrm{SNR}(t)\right) ||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)||_2^2 \right], \tag{13}$$

where $U\{1,T\}$ is the uniform distribution on the integers $\{1,\ldots,T\}$, and $\mathbf{z}_t = \alpha_t \mathbf{x} + \sigma_t \boldsymbol{\epsilon}$. After expanding the σ_t, α_t and $\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)$ terms with their specifications in Sections 3.2 and 3.4, this simplifies to:

$$\mathcal{L}_T(\mathbf{x}) = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}), i\sim U\{1,T\}} \left[\frac{T}{2} (\exp(\gamma_{\boldsymbol{\eta}}(t) - \gamma_{\boldsymbol{\eta}}(s)) - 1) \left\|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)\right\|_2^2 \right] \tag{14}$$

where $\mathbf{z}_t = \mathrm{sigmoid}(-\gamma_{\boldsymbol{\eta}}(t))\mathbf{x} + \mathrm{sigmoid}(\gamma_{\boldsymbol{\eta}}(t))\boldsymbol{\epsilon}$. In the discrete-time case, we simply jointly optimize $\boldsymbol{\eta}$ and $\boldsymbol{\theta}$ by maximizing the VLB through a Monte Carlo estimator of Equation 14.

This simple diffusion loss allows for a simple numerically stable implementation in 32-bit or lower-precision floating point, in contrast with previous implementations of discrete-time diffusion models (e.g. [Ho et al., 2020], which had to use 64-bit floating point. Note that $\exp(.) - 1$ has a numerically stable primitive $expm1(.)$ in common numerical computing packages.

4.1 More steps leads to a lower loss

A natural question to ask is what the number of timesteps T should be, and whether more timesteps is always better in terms of the VLB. In Appendix F we analyze the difference between the diffusion loss with T timesteps, $\mathcal{L}_T(\mathbf{x})$, and the diffusion loss with double the timesteps, $\mathcal{L}_{2T}(\mathbf{x})$, while keeping the SNR function fixed. We then find that if our trained denoising model $\hat{\mathbf{x}}_{\boldsymbol{\theta}}$ is sufficiently good, we have that $\mathcal{L}_{2T}(\mathbf{x}) < \mathcal{L}_T(\mathbf{x})$, i.e. that our VLB will be better for a larger number of timesteps. Intuitively, the discrete time diffusion loss is an upper Riemann sum approximation of an integral of a strictly decreasing function, meaning that a finer approximation yields a lower diffusion loss. This result is illustrated in Figure 2.

5 Continuous-time model: $T \to \infty$

Since taking more time steps leads to a better VLB, we now take $T \to \infty$, effectively treating time t as continuous rather than discrete. The model for $p(\mathbf{z}_t)$ can then be described as a continuous time

diffusion process [Song et al., 2021b] governed by a stochastic differential equation; see Appendix D. In Appendix E we show that in this limit the diffusion loss $\mathcal{L}_T(\mathbf{x})$ simplifies further. Letting $\mathrm{SNR}'(t) = d\mathrm{SNR}(t)/dt$, we have, with $\mathbf{z}_t = \alpha_t \mathbf{x} + \sigma_t \boldsymbol{\epsilon}$:

$$\mathcal{L}_\infty(\mathbf{x}) = -\frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})} \int_0^1 \mathrm{SNR}'(t)\, \|\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)\|_2^2\, dt, \tag{15}$$

$$= -\frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}), t\sim\mathcal{U}(0,1)} \left[\mathrm{SNR}'(t)\, \|\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)\|_2^2\right]. \tag{16}$$

After expanding the σ_t, α_t and $\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)$ terms with their specifications in Sections 3.2 and 3.4, this simplifies to:

$$\mathcal{L}_\infty(\mathbf{x}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}), t\sim\mathcal{U}(0,1)} \left[\gamma'_{\boldsymbol{\eta}}(t)\, \|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)\|_2^2\right], \tag{17}$$

where $\gamma'_{\boldsymbol{\eta}}(t) = d\gamma_{\boldsymbol{\eta}}(t)/dt$. We use the Monte Carlo estimator of this loss for evaluation and optimization.

5.1 Equivalence of diffusion models in continuous time

The signal-to-noise function $\mathrm{SNR}(t)$ is invertible due to the monotonicity assumption in Section 3.1. Due to this invertibility, we can perform a change of variables, and make everything a function of $v \equiv \mathrm{SNR}(t)$ instead of t, such that $t = \mathrm{SNR}^{-1}(v)$. Let α_v and σ_v be the functions α_t and σ_t evaluated at $t = \mathrm{SNR}^{-1}(v)$, and correspondingly let $\mathbf{z}_v = \alpha_v \mathbf{x} + \sigma_v \boldsymbol{\epsilon}$. Similarly, we rewrite our noise prediction model as $\tilde{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}, v) \equiv \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}, \mathrm{SNR}^{-1}(v))$. With this change of variables, our continuous-time loss in Equation 15 can equivalently be written as:

$$\mathcal{L}_\infty(\mathbf{x}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})} \int_{\mathrm{SNR}_{\mathrm{min}}}^{\mathrm{SNR}_{\mathrm{max}}} \|\mathbf{x} - \tilde{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_v, v)\|_2^2\, dv, \tag{18}$$

where instead of integrating w.r.t. time t we now integrate w.r.t. the signal-to-noise ratio v, and where $\mathrm{SNR}_{\mathrm{min}} = \mathrm{SNR}(1)$ and $\mathrm{SNR}_{\mathrm{max}} = \mathrm{SNR}(0)$.

What this equation shows us is that the only effect the functions $\alpha(t)$ and $\sigma(t)$ have on the diffusion loss is through the values $\mathrm{SNR}(t) = \alpha_t^2/\sigma_t^2$ at endpoints $t = 0$ and $t = 1$. Given these values $\mathrm{SNR}_{\mathrm{max}}$ and $\mathrm{SNR}_{\mathrm{min}}$, the diffusion loss is invariant to the shape of function $\mathrm{SNR}(t)$ between $t = 0$ and $t = 1$. The VLB is thus only impacted by the function $\mathrm{SNR}(t)$ through its endpoints $\mathrm{SNR}_{\mathrm{min}}$ and $\mathrm{SNR}_{\mathrm{max}}$.

Furthermore, we find that the distribution $p(\mathbf{x})$ defined by our generative model is also invariant to the specification of the diffusion process. Specifically, let $p^A(\mathbf{x})$ denote the distribution defined by the combination of a diffusion specification and denoising function $\{\alpha_v^A, \sigma_v^A, \tilde{\mathbf{x}}_{\boldsymbol{\theta}}^A\}$, and similarly let $p^B(\mathbf{x})$ be the distribution defined through a different specification $\{\alpha_v^B, \sigma_v^B, \tilde{\mathbf{x}}_{\boldsymbol{\theta}}^B\}$, where both specifications have equal $\mathrm{SNR}_{\mathrm{min}}$, $\mathrm{SNR}_{\mathrm{max}}$; as shown in Appendix G, we then have that $p^A(\mathbf{x}) = p^B(\mathbf{x})$ if $\tilde{\mathbf{x}}_{\boldsymbol{\theta}}^B(\mathbf{z}, v) \equiv \tilde{\mathbf{x}}_{\boldsymbol{\theta}}^A((\alpha_v^A/\alpha_v^B)\mathbf{z}, v)$. The distribution on all latents $\mathbf{z}_v$ is then also the same under both specifications, up to a trivial rescaling. This means that any two diffusion models satisfying the mild constraints set in 3.1 (which includes e.g. the *variance exploding* and *variance preserving* specifications considered by Song et al. [2021b]), can thus be seen as equivalent in continuous time.

5.2 Weighted diffusion loss

This equivalence between diffusion specifications continues to hold even if, instead of the VLB, these models optimize a *weighted* diffusion loss of the form:

$$\mathcal{L}_\infty(\mathbf{x}, w) = \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})} \int_{\mathrm{SNR}_{\mathrm{min}}}^{\mathrm{SNR}_{\mathrm{max}}} w(v)\, \|\mathbf{x} - \tilde{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_v, v)\|_2^2\, dv, \tag{19}$$

which e.g. captures all the different objectives discussed by Song et al. [2021b], see Appendix K. Here, $w(v)$ is a weighting function that generally puts increased emphasis on the noisier data compared to the VLB, and which thereby can sometimes improve perceptual generation quality as measured by certain metrics like FID and Inception Score. For the models presented in this paper, we further use $w(v) = 1$ as corresponding to the (unweighted) VLB.

Model (Bits per dim on test set)	Type	CIFAR10 no data aug.	CIFAR10 data aug.	ImageNet 32x32	ImageNet 64x64
Previous work					
ResNet VAE with IAF [Kingma et al., 2016]	VAE	3.11			
Very Deep VAE [Child, 2020]	VAE	2.87		3.80	3.52
NVAE [Vahdat and Kautz, 2020]	VAE	2.91		3.92	
Glow [Kingma and Dhariwal, 2018]	Flow		$3.35^{(B)}$	4.09	3.81
Flow++ [Ho et al., 2019a]	Flow	3.08		3.86	3.69
PixelCNN [Van Oord et al., 2016]	AR	3.03		3.83	3.57
PixelCNN++ [Salimans et al., 2017]	AR	2.92			
Image Transformer [Parmar et al., 2018]	AR	2.90		3.77	
SPN [Menick and Kalchbrenner, 2018]	AR				3.52
Sparse Transformer [Child et al., 2019]	AR	2.80			3.44
Routing Transformer [Roy et al., 2021]	AR				3.43
Sparse Transformer + DistAug [Jun et al., 2020]	AR		$2.53^{(A)}$		
DDPM [Ho et al., 2020]	Diff		$3.69^{(C)}$		
EBM-DRL [Gao et al., 2020]	Diff		$3.18^{(C)}$		
Score SDE [Song et al., 2021b]	Diff	2.99			
Improved DDPM [Nichol and Dhariwal, 2021]	Diff	2.94			3.54
Concurrent work					
CR-NVAE [Sinha and Dieng, 2021]	VAE		$2.51^{(A)}$		
LSGM [Vahdat et al., 2021]	Diff	2.87			
ScoreFlow [Song et al., 2021a] (variational bound)	Diff		$2.90^{(C)}$	3.86	
ScoreFlow [Song et al., 2021a] (cont. norm. flow)	Diff	2.83	$2.80^{(C)}$	3.76	
Our work					
VDM (variational bound)	Diff	**2.65**	**2.49**$^{(A)}$	**3.72**	**3.40**

Table 1: Summary of our findings for density modeling tasks, in terms of bits per dimension (BPD) on the test set. Model types are autoregressive (AR), normalizing flows (Flow), variational autoencoders (VAE), or diffusion models (Diff). Our results were obtained using the continuous-time formulation of our model. CIFAR-10 data augmentations are: (A) extensive, (B) small translations, or (C) horizontal flips. The numbers for VDM are variational bounds, and can likely be improved by estimating the marginal likelihood through importance sampling, or through evaluation of the corresponding continuous normalizing flow as done by Song et al. [2021a].

5.3 Variance minimization

Due to the invariance shown in Section 5.1 for the continous-time case, we therefore only optimize the endpoints of the noise schedule w.r.t. the VLB, and not the schedule $\gamma_{\boldsymbol{\eta}}(t)$ interpolating between them. The value of $\gamma_{\boldsymbol{\eta}}(t)$ for $0 < t < 1$ does impact the variance of our Monte Carlo estimate of it. We therefore propose to optimize $\gamma_{\boldsymbol{\eta}}(t)$ for $0 < t < 1$ by minimizing the variance of the Monte Carlo estimator, as detailed in Appendix I. We also found it helpful to sample t using a low-discrepancy sampler, as explained in Appendix I.1.

6 Experiments

We demonstrate our proposed class of diffusion models, which we call *Variational Diffusion Models* (VDMs), on the CIFAR-10 [Krizhevsky et al., 2009] dataset, and the downsampled ImageNet [Van Oord et al., 2016, Deng et al., 2009] dataset, where we focus on maximizing likelihood. For our result with data augmentation we used random flips, 90-degree rotations, and color channel swapping. More details of our model specifications are in Appendix B.

6.1 Likelihood and samples

Table 1 shows our results on modeling the CIFAR-10 dataset, and the downsampled ImageNet dataset. We establish a new state-of-the-art in terms of test set likelihood on all considered benchmarks, by a significant margin. Our model for CIFAR-10 without data augmentation surpasses the previous best

Figure 3: Non cherry-picked unconditional samples from our Imagenet 64x64 model, trained in continuous time and generated using $T = 1000$. The model's hyper-parameters and parameters are optimized w.r.t. the likelihood bound, so the model is not optimized for synthesis quality.

result of 2.80 about 10x faster than it takes the Sparse Transformer to reach this, in wall clock time on equivalent hardware.

Our CIFAR-10 model, whose hyper-parameters were tuned for likelihood, results in a FID (perceptual quality) score of 7.41. This would have been state-of-the-art until recently, but is worse than recent diffusion models that specifically target FID scores [Nichol and Dhariwal, 2021, Song et al., 2021b, Ho et al., 2020]. By instead using a weighted diffusion loss, with the weighting function $w(\text{SNR})$ used by Ho et al. [2020] and described in Appendix K, our FID score improves to 4.0. We did not pursue further tuning of the model to improve FID instead of likelihood.

A random sample of generated images from our model is provided in Figure 3. We provide additional samples from this model, as well as our other models for the other datasets, in Appendix M.

6.2 Ablations

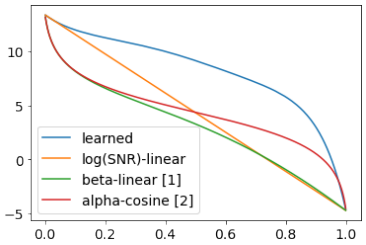
Next, we investigate the relative importance of our contributions. In Table 2 we compare our discrete-time and continuous-time specifications of the diffusion model: When evaluating our model with a small number of steps, our discretely trained models perform better by learning the diffusion schedule to optimize the VLB. However, as argued theoretically in Section 4.1, we find experimentally that more steps T indeed gives better likelihood. When T grows large, our continuously trained model performs best, helped by training its diffusion schedule to minimize variance instead.

Minimizing the variance also helps the continuous time model to train faster, as shown in Figure 5. This effect is further examined in Table 4b, where we find dramatic variance reductions compared to our baselines in continuous time. Figure 4a shows how this effect is achieved: Compared to the other schedules, our learned schedule spends much more time in the high $\text{SNR}(t)$ / low σ_t^2 range.

In Figure 5 we further show training curves for our model including and excluding the Fourier features proposed in Appendix C: with Fourier features enabled our model achieves much better likelihood. For comparison we also implemented Fourier features in a PixelCNN++ model [Salimans et al., 2017], where we do not see a benefit. In addition, we find that learning the SNR is necessary to get the most out of including Fourier features: if we fix the SNR schedule to that used by Ho et al. [2020], the maximum log-SNR is fixed to approximately 8 (see figure 6), and test set negative likelihood stays above 4 bits per dim. When learning the SNR endpoints, our maximum log-SNR ends up at 13.3, which, combined with the inclusion of Fourier features, leads to the SOTA test set likelihoods reported in Table 1.

6.3 Lossless compression

For a fixed number of evaluation timesteps T_{eval}, our diffusion model in discrete time is a hierarchical latent variable model that can be turned into a lossless compression algorithm using bits-back coding [Hinton and Van Camp, 1993]. As a proof of concept of practical lossless compression, Table 2 reports net codelengths on the CIFAR10 test set for various settings of T_{eval} using BB-ANS [Townsend et al., 2018], an implementation of bits-back coding based on asymmetric numeral systems [Duda, 2009]. Details of our implementation are given in Appendix N. We achieve state-of-the-art net codelengths, proving our model can be used as the basis of a lossless compression algorithm.



(a) log SNR vs time t

SNR(t) schedule	Var(BPD)
Learned (ours)	**0.53**
log SNR-linear	6.35
β-Linear [1]	31.6
α-Cosine [2]	31.1

(b) Variance of VLB estimate

Figure 4: Our learned continuous-time variance-minimizing noise schedule SNR(t) for CIFAR-10, compared to its log-linear initialization and to schedules from the literature: [1] The β-Linear schedule from Ho et al. [2020], [2] The α-Cosine schedule from Nichol and Dhariwal [2021]. All schedules were scaled and shifted on the log scale such that the resulting $\text{SNR}_{\text{min}}, \text{SNR}_{\text{max}}$ were the equal to our learned endpoints, resulting in the same VLB estimate of 2.66. We report the variance of our VLB estimate per data point, computed on the test set, and conditional on the data: This does not include the noise due to sampling minibatches of data.

T_{train}	T_{eval}	BPD	Bits-Back Net BPD
10	10	4.31	
100	100	2.84	
250	250	2.73	
500	500	2.68	
1000	1000	2.67	
10000	10000	2.66	
∞	10	7.54	7.54
∞	100	2.90	2.91
∞	250	2.74	2.76
∞	500	2.69	2.72
∞	1000	2.67	2.72
∞	10000	**2.65**	
∞	∞	**2.65**	

Table 2: Discrete versus continuous-time training and evaluation with CIFAR-10, in terms of bits per dimension (BPD).

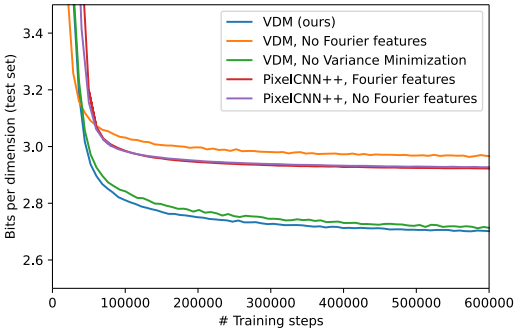


Figure 5: Test set likelihoods during training, with/without Fourier features, and with/without learning the noise schedule to minimize variance.

However, for large T_{eval} a gap remains with the theoretically optimal codelength corresponding to the negative VLB, and compression becomes computationally expensive due to the large number of neural network forward passes required. Closing this gap with more efficient implementations of bits-back coding suitable for very deep models is an interesting avenue for future work.

7 Conclusion

We presented state-of-the-art results on modeling the density of natural images using a new class of diffusion models that incorporates a learnable diffusion specification, Fourier features for fine-scale modeling, as well as other architectural innovations. In addition, we obtained new theoretical insight into likelihood-based generative modeling with diffusion models, showing a surprising invariance of the VLB to the forward time diffusion process in continuous time, as well as an equivalence between various diffusion processes from the literature previously thought to be different.

Acknowledgments

We thank Yang Song, Kevin Murphy and Mohammad Norouzi for feedback on drafts of this paper.

References

Nanxin Chen, Yu Zhang, Heiga Zen, Ron J. Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation. *arXiv preprint arXiv:2009.00713*, 2020.

Rewon Child. Very deep VAEs generalize autoregressive models and can outperform them on images. *arXiv preprint arXiv:2011.10650*, 2020.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

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

Jarek Duda. Asymmetric numeral systems. *arXiv preprint arXiv:0902.0271*, 2009.

Ruiqi Gao, Yang Song, Ben Poole, Ying Nian Wu, and Diederik P Kingma. Learning energy-based models by diffusion recovery likelihood. *arXiv preprint arXiv:2012.08125*, 2020.

Geoffrey E Hinton and Drew Van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, pages 5–13, 1993.

Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *International Conference on Machine Learning*, pages 2722–2730. PMLR, 2019a.

Jonathan Ho, Evan Lohn, and Pieter Abbeel. Compression with flows via local bits-back coding. In *Advances in Neural Information Processing Systems*, pages 3874–3883, 2019b.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239*, 2020.

Chin-Wei Huang, Jae Hyun Lim, and Aaron Courville. A variational perspective on diffusion-based generative models and score matching. *arXiv preprint arXiv:2106.02808*, 2021.

Heewoo Jun, Rewon Child, Mark Chen, John Schulman, Aditya Ramesh, Alec Radford, and Ilya Sutskever. Distribution augmentation for generative modeling. In *International Conference on Machine Learning*, pages 5006–5019. PMLR, 2020.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039*, 2018.

Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. *International Conference on Learning Representations*, 2013.

Diederik P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, pages 4743–4751, 2016.

Friso Kingma, Pieter Abbeel, and Jonathan Ho. Bit-swap: Recursive bits-back coding for lossless compression with hierarchical latent variables. In *International Conference on Machine Learning*, pages 3408–3417. PMLR, 2019.

Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020.

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

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

Jacob Menick and Nal Kalchbrenner. Generating high fidelity images with subscale pixel networks and multidimensional upscaling. *arXiv preprint arXiv:1812.01608*, 2018.

Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. *arXiv preprint arXiv:2102.09672*, 2021.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *International Conference on Machine Learning*, pages 4055–4064. PMLR, 2018.

Danilo J Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pages 1278–1286, 2014.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. Efficient content-based sparse attention with routing transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68, 2021.

Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *arXiv preprint arXiv:1701.05517*, 2017.

Samarth Sinha and Adji B Dieng. Consistency regularization for variational auto-encoders. *arXiv preprint arXiv:2105.14859*, 2021.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265, 2015.

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

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, pages 11895–11907, 2019.

Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien 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.

Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. *arXiv e-prints*, pages arXiv–2101, 2021a.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-Based Generative Modeling Through Stochastic Differential Equations. In *International Conference on Learning Representations*, 2021b.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*, 2014.

Matthew Tancik, Pratul P Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *arXiv preprint arXiv:2006.10739*, 2020.

James Townsend, Thomas Bird, and David Barber. Practical lossless compression with latent variables using bits back coding. In *International Conference on Learning Representations*, 2018.

James Townsend, Thomas Bird, Julius Kunze, and David Barber. HiLLoC: lossless image compression with hierarchical latent variable models. In *International Conference on Learning Representations*, 2020.

Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder. *arXiv preprint arXiv:2007.03898*, 2020.

Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. *arXiv preprint arXiv:2106.05931*, 2021.

Aaron Van Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *International Conference on Machine Learning*, pages 1747–1756, 2016.

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

Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

A Distribution details

A.1 $q(\mathbf{z}_t|\mathbf{z}_s)$

The distribution of $\mathbf{z}_t$ given $\mathbf{z}_s$, for any $0 \leq s < t \leq 1$, is given by:

$$q(\mathbf{z}_t|\mathbf{z}_s) = \mathcal{N}\left(\alpha_{t|s}\mathbf{z}_s, \sigma^2_{t|s}\mathbf{I}\right), \quad \text{where} \quad \alpha_{t|s} = \alpha_t/\alpha_s, \quad \text{and} \quad \sigma^2_{t|s} = \sigma^2_t - \alpha^2_{t|s}\sigma^2_s. \tag{20}$$

A.2 $q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x})$

Given the distributions $q(\mathbf{z}_t|\mathbf{x})$ (Equation 1) and $q(\mathbf{z}_t|\mathbf{z}_s)$ (Equation 20), we apply Bayes rule to derive the distribution $q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x})$ for any $0 \leq s < t \leq 1$, yielding:

$$q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}_Q(\mathbf{z}_t, \mathbf{x}; s, t), \sigma^2_Q(s,t)\mathbf{I}) \tag{21}$$

$$\text{where} \quad \sigma^2_Q(s,t) = \sigma^2_{t|s}\sigma^2_s/\sigma^2_t, \tag{22}$$

$$\text{and} \quad \boldsymbol{\mu}_Q(\mathbf{z}_t, \mathbf{x}; s, t) = \frac{1}{\alpha_{t|s}}(\mathbf{z}_t + \sigma^2_{t|s}\nabla_{\mathbf{z}_t}\log q(\mathbf{z}_t|\mathbf{x})) = \frac{\alpha_{t|s}\sigma^2_s}{\sigma^2_t}\mathbf{z}_t + \frac{\alpha_s\sigma^2_{t|s}}{\sigma^2_t}\mathbf{x}. \tag{23}$$

A.3 $p(\mathbf{z}_s|\mathbf{z}_t)$

Finally, we choose the conditional model distributions as

$$p(\mathbf{z}_s|\mathbf{z}_t) = q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x} = \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)), \tag{24}$$

i.e. the same as $q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x})$, but with the original data $\mathbf{x}$ replaced by the output of a denoising model $\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)$ that predicts $\mathbf{x}$ from its noisy version $\mathbf{z}_t$. We then have $p(\mathbf{z}_s|\mathbf{z}_t) = \mathcal{N}(\mathbf{z}_s; \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z}_t; s, t), \sigma^2_Q(s,t)\mathbf{I})$, with

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z}_t; s, t) = \frac{\alpha_{t|s}\sigma^2_s}{\sigma^2_t}\mathbf{z}_t + \frac{\alpha_s\sigma^2_{t|s}}{\sigma^2_t}\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t) = \frac{1}{\alpha_{t|s}}\mathbf{z}_t - \frac{\sigma^2_{t|s}}{\alpha_{t|s}\sigma_t}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t) = \frac{1}{\alpha_{t|s}}\mathbf{z}_t + \frac{\sigma^2_{t|s}}{\alpha_{t|s}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t; t), \tag{25}$$

with variance $\sigma^2_Q(s,t)$ the same as in Equation 22, and with

$$\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t) = (\mathbf{z}_t - \alpha_t\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t))/\sigma_t \quad \text{and} \quad \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t; t) = (\alpha_t\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t) - \mathbf{z}_t)/\sigma^2_t. \tag{26}$$

Equation 25 shows that we can interpret our model in three different ways: 1) In terms of the *denoising model* $\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)$ that recovers $\mathbf{x}$ from its corrupted version $\mathbf{z}_t$. 2) In terms of a *noise prediction model* $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)$ that directly infers the noise $\boldsymbol{\epsilon}$ that was used to generate $\mathbf{z}_t$. 3) In terms of a *score model* $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)$, that at its optimum equals the scores of the marginal density: $\mathbf{s}^*(\mathbf{z}_t; t) = \nabla_{\mathbf{z}}\log q(\mathbf{z}_t)$; see Appendix L. These are three equally valid views on the same model class, that have been used interchangeably in the literature. We find the denoising interpretation the most intuitive, and will therefore mostly use $\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)$ in the theoretical part of this paper, although in practice we parameterize our model via $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)$ following Ho et al. [2020]. The parameterization of our model is discussed in Appendix B.

B Hyperparameters, architecture, and implementation details

In this section we provide details on the exact setup for each of our experiments. In Sections B.1 we describe the choices in common to each of our experiments. Hyperparameters specific to the individual experiments are given in Section B.2. We are currently working towards open sourcing our code.

B.1 Model and implementation

Our denoising models are parameterized in terms of noise prediction models $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t; \gamma_t)$, as explained in Section 3.4. Our noise prediction models $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ closely follow the architecture used by Ho et al. [2020], which is based on a U-Net type neural net [Ronneberger et al., 2015] that maps from the

input $\mathbf{z} \in \mathbb{R}^d$ to output of the same dimension. As compared to their publically available code at https://github.com/hojonathanho/diffusion, our implementation differs in the following ways:

- Our networks don't perform any internal downsampling or upsampling: we process all the data at the original input resolution.
- Our models are deeper than those used by Ho et al. [2020]. Specific numbers are given in Section B.2.
- Instead of taking time t as input to the noise prediction model, we use γ_t, which we rescale to have approximately the same range as t of $[0, 1]$ before using it to form 'time' embeddings in the same way as Ho et al. [2020].
- Our models calculate Fourier features on the input data $\mathbf{z}_t$ as discussed in Appendix C, which are then concatenated to $\mathbf{z}_t$ before being fed to the U-Net.
- Apart from the *middle* attention block that connects the upward and downward branches of the U-Net, we remove all other attention blocks from the model. We found that these attention blocks made it more likely for the model to overfit to the training set.
- All of our models use dropout at a rate of 0.1 in the intermediate layers, as did Ho et al. [2020]. In addition we regularize the model by using decoupled weight decay [Loshchilov and Hutter, 2017] with coefficient 0.01.
- We use the Adam optimizer with a learning rate of $2e^{-4}$ and exponential decay rates of $\beta_1 = 0.9, \beta_2 = 0.99$. We found that higher values for β_2 resulted in training instabilities.
- For evaluation, we use an exponential moving average of our parameters, calculated with an exponential decay rate of 0.9999.

We regularly evaluate the variational bound on the likelihood on the validation set and find that our models do not overfit during training, using the current settings. We therefore do not use early stopping and instead allow the network to be optimized for 10 million parameter updates for CIFAR-10, and for 2 million updates for ImageNet, before obtaining the test set numbers reported in this paper. It looks like our models keep improving even after this number of updates, in terms of likelihood, but we did not explore this systematically due to resource constraints.

All of our models are trained on TPUv3 hardware (see https://cloud.google.com/tpu) using data parallelism. We also evaluated our trained models using CPU and GPU to check for robustness of our reported numbers to possible rounding errors. We found only very small differences when evaluating on these other hardware platforms.

B.2 Settings for each dataset

Our model for CIFAR-10 with no data augmentation uses a U-Net of depth 32, consisting of 32 ResNet blocks in the forward direction and 32 ResNet blocks in the reverse direction, with a single attention layer and two additional ResNet blocks in the middle. We keep the number of channels constant throughout at 128. This model was trained on 8 TPUv3 chips, with a total batch size of 128 examples. Reaching a test-set BPD of 2.65 after 10 million updates takes 9 days, although our model already surpasses sparse transformers (the previous state-of-the-art) of 2.80 BPD after only 2.5 hours of training.

For CIFAR-10 with data augmentation we used random flips, 90-degree rotations, and color channel swapping, which were previously shown to help for density estimation by Jun et al. [2020]. Each of the three augmentations independently were given a 50% probability of being applied to each example, which means that 1 in 8 training examples was not augmented at all. For this experiment, we doubled the number of channels in our model to 256, and decreased the dropout rate from 10% to 5%. Since overfitting was less of a problem with data augmentation, we add back the attention blocks after each ResNet block, following Ho et al. [2020]. We also experimented with conditioning our model on an additional binary feature that indicates whether or not the example was augmented, which can be seen as a simplified version of the augmentation conditioning proposed by Jun et al. [2020]. Conditioning made almost no difference to our results, which may be explained by the relatively large fraction (12.5%) of clean data fed to our model during training. We trained our model for slightly over a week on 128 TPUv3 chips to obtain the reported result.

Our model for 32x32 ImageNet looks similar to that for CIFAR-10 without data augmentation, with a U-Net depth of 32, but uses double the number of channels at 256. It is trained using data parallelism on 32 TPUv3 chips, with a total batch size of 512.

Our model for 64x64 ImageNet uses double the depth at 64 ResNet layers in both the forward and backward direction in the U-Net. It also uses a constant number of channels of 256. This model is trained on 128 TPUv3 chips at a total batch size of 512 examples.

C Fourier features for improved fine scale prediction

Prior work on diffusion models has mainly focused on the perceptual quality of generated samples, which emphasizes coarse scale patterns and global consistency of generated images. Here, we optimize for likelihood, which is sensitive to fine scale details and exact values of individual pixels. Since our reconstruction model $p(\mathbf{x}|\mathbf{z}_0)$ given in Equation 8 is weak, the burden of modeling these fine scale details falls on our denoising diffusion model $\hat{\mathbf{x}}_{\boldsymbol{\theta}}$. In initial experiments, we found that the denoising model had a hard time accurately modeling these details. At larger noise levels, the latents $\mathbf{z}_t$ follow a smooth distribution due to the added Gaussian noise, but at the smallest noise levels the discrete nature of 8-bit image data leads to sharply peaked marginal distributions $q(\mathbf{z}_t)$.

To capture the fine scale details of the data, we propose adding a set of *Fourier features* to the input of our denoising model $\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)$. Such Fourier features consist of a linear projection of the original data onto a set of periodic basis functions with high frequency, which allows the network to more easily model high frequency details of the data. Previous work [Tancik et al., 2020] has used these features for input coordinates to model high frequency details across the *spatial* dimension, and for time embeddings to condition denoising networks over the *temporal* dimension [Song et al., 2021b]. Here we apply it to color channels for single pixels, in order to model fine distributional details at the level of each scalar input.

Concretely, let $z_{i,j,k}$ be the scalar value in the k-th channel in the (i, j) spatial position of network input $\mathbf{z}_t$. We then add additional channels to the input of the denoising model of the form

$$f^n_{i,j,k} = \sin(z_{i,j,k} 2^n \pi), \quad \text{and} \quad g^n_{i,j,k} = \cos(z_{i,j,k} 2^n \pi), \tag{27}$$

where n runs over a range of integers $\{n_{min}, ..., n_{max}\}$. These additional channels are then concatenated to $\mathbf{z}_t$ before being used as input in a standard convolutional denoising model similar to that used by Ho et al. [2020]. We find that the presence of these high frequency features allows our network to learn with much higher values of SNR_{max}, or conversely lower noise levels σ_0^2, than is otherwise optimal. This leads to large improvements in likelihood as demonstrated in Section 6 and Figure 5. We did not observe such improvements when incorporating Fourier features into autoregressive models.

In our expreriments, we got best results with $n_{min} = 7$ and $n_{max} = 8$, probably since Fourier features with these frequencies are most relevant; features with lower frequencies can be learned by the network from $\mathbf{z}$, and higher frequencies are not present in the data thus irrelevant for likelihood.

D As a SDE

When we take the number of steps $T \to \infty$, our model for $p(\mathbf{z}_t)$ can best be described as a continuous time diffusion process [Song et al., 2021b], governed by the stochastic differential equation

$$d\mathbf{z}_t = [f(t)\mathbf{z}_t - g^2(t) s_{\boldsymbol{\theta}}(\mathbf{z}_t; t)]dt + g(t) d\mathbf{W}_t, \tag{28}$$

with time running backwards from $t = 1$ to $t = 0$, where $\mathbf{W}$ denotes a Wiener process (standard Brownian motion), and

$$f(t) = \frac{d \log \alpha_t}{dt}, \qquad g^2(t) = \frac{d\sigma^2(t)}{dt} - 2\frac{d \log \alpha_t}{dt}\sigma^2(t). \tag{29}$$

E Derivation of the VLB estimators

E.1 Discrete-time VLB

Similar to [Sohl-Dickstein et al., 2015], we decompose the negative variational lower bound (VLB) as:

$$-\log p(\mathbf{x}) \leq -\text{VLB}(\mathbf{x}) = \underbrace{D_{KL}(q(\mathbf{z}_1|\mathbf{x})||p(\mathbf{z}_1))}_{\text{Prior loss}} + \underbrace{\mathbb{E}_{q(\mathbf{z}_0|\mathbf{x})}\left[-\log p(\mathbf{x}|\mathbf{z}_0)\right]}_{\text{Reconstruction loss}} + \underbrace{\mathcal{L}_T(\mathbf{x})}_{\text{Diffusion loss}}. \tag{30}$$

The prior loss and reconstruction loss can be (stochastically and differentiably) estimated using standard techniques. We will now derive an estimator for the diffusion loss $\mathcal{L}_T(\mathbf{x})$, the remaining and more challenging term. In the case of finite T, using $s(i) = (i-1)/T$, $t(i) = i/T$, the diffusion loss is:

$$\mathcal{L}_T(\mathbf{x}) = \sum_{i=1}^{T} \mathbb{E}_{q(\mathbf{z}_{t(i)}|\mathbf{x})} D_{KL}[q(\mathbf{z}_{s(i)}|\mathbf{z}_{t(i)}, \mathbf{x})||p(\mathbf{z}_{s(i)}|\mathbf{z}_{t(i)})]. \tag{31}$$

We will use s and t as shorthands for $s(i)$ and $t(i)$. We will first derive an expression of $D_{KL}(q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x})||p(\mathbf{z}_s|\mathbf{z}_t))$.

Recall that $p(\mathbf{z}_s|\mathbf{z}_t) = q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x} = \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t))$, and thus $q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}) = \mathcal{N}(\mathbf{z}_s; \boldsymbol{\mu}_Q(\mathbf{z}_t, \mathbf{x}; s, t), \sigma_Q^2(s,t)\mathbf{I})$ and $p(\mathbf{z}_s|\mathbf{z}_t) = \mathcal{N}(\mathbf{z}_s; \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z}_t; s, t), \sigma_Q^2(s,t)\mathbf{I})$, with

$$\boldsymbol{\mu}_Q(\mathbf{z}_t, \mathbf{x}; s, t) = \frac{\alpha_{t|s}\sigma_s^2}{\sigma_t^2}\mathbf{z}_t + \frac{\alpha_s \sigma_{t|s}^2}{\sigma_t^2}\mathbf{x} \tag{32}$$

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z}_t; s, t) = \frac{\alpha_{t|s}\sigma_s^2}{\sigma_t^2}\mathbf{z}_t + \frac{\alpha_s \sigma_{t|s}^2}{\sigma_t^2}\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t), \tag{33}$$

$$\text{and} \quad \sigma_Q^2(s,t) = \sigma_{t|s}^2 \sigma_s^2 / \sigma_t^2. \tag{34}$$

Since $q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x})$ and $p(\mathbf{z}_s|\mathbf{z}_t)$ are Gaussians, their KL divergence is available in closed form as a function of their means and variances, which due to their with equal variances simplifies as:

$$\begin{aligned}
D_{KL}(q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x})||p(\mathbf{z}_s|\mathbf{z}_t)) &= \frac{1}{2\sigma_Q^2(s,t)}||\boldsymbol{\mu}_Q - \boldsymbol{\mu}_{\boldsymbol{\theta}}||_2^2 && (35)\\
&= \frac{\sigma_t^2}{2\sigma_{t|s}^2\sigma_s^2}\frac{\alpha_s^2\sigma_{t|s}^4}{\sigma_t^4}||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)||_2^2 && (36)\\
&= \frac{1}{2\sigma_s^2}\frac{\alpha_s^2\sigma_{t|s}^2}{\sigma_t^2}||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)||_2^2 && (37)\\
&= \frac{1}{2\sigma_s^2}\frac{\alpha_s^2(\sigma_t^2 - \alpha_{t|s}^2\sigma_s^2)}{\sigma_t^2}||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)||_2^2 && (38)\\
&= \frac{1}{2}\frac{\alpha_s^2\sigma_t^2/\sigma_s^2 - \alpha_t^2}{\sigma_t^2}||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)||_2^2 && (39)\\
&= \frac{1}{2}\left(\frac{\alpha_s^2}{\sigma_s^2} - \frac{\alpha_t^2}{\sigma_t^2}\right)||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)||_2^2 && (40)\\
&= \frac{1}{2}\left(\text{SNR}(s) - \text{SNR}(t)\right)||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)||_2^2 && (41)
\end{aligned}$$

Reparameterizing $\mathbf{z}_t \sim q(\mathbf{z}_t|\mathbf{x})$ as $\mathbf{z}_t = \alpha_t\mathbf{x} + \sigma_t\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$, our diffusion loss becomes:

$$\begin{aligned}
\mathcal{L}_T(\mathbf{x}) &= \sum_{i=1}^{T}\mathbb{E}_{q(\mathbf{z}_t|\mathbf{x})}[D_{KL}(q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x})||p(\mathbf{z}_s|\mathbf{z}_t))] && (42)\\
&= \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})}[\sum_{i=1}^{T}\left(\text{SNR}(s) - \text{SNR}(t)\right)||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)||_2^2] && (43)
\end{aligned}$$

E.2 Estimator of $\mathcal{L}_T(\mathbf{x})$

To avoid having to compute all T terms when calculating the diffusion loss, we construct an unbiased estimator of $\mathcal{L}_T(\mathbf{x})$ using

$$\mathcal{L}_T(\mathbf{x}) = \frac{T}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}),i\sim U\{1,T\}}\left[(\text{SNR}(s) - \text{SNR}(t))\,||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t;t)||_2^2\right] \tag{44}$$

where $U\{1,T\}$ is the discrete uniform distribution from 1 to (and including) T, $s = (i-1)/T$, $t = i/T$ and $\mathbf{z}_t = \alpha_t\mathbf{x} + \sigma_t\boldsymbol{\epsilon}$. This trivially yields an unbiased Monte Carlo estimator, by drawing random samples $i \sim U\{1,T\}$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(0,\mathbf{I})$.

E.3 Infinite depth ($T \to \infty$)

To calculate the limit of the diffusion loss as $T \to \infty$, we express $\mathcal{L}_T(\mathbf{x})$ as a function of $\tau = 1/T$:

$$\mathcal{L}_T(\mathbf{x}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}),i\sim U\{1,T\}}\left[\frac{\text{SNR}(t-\tau) - \text{SNR}(t)}{\tau}||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t;t)||_2^2\right], \tag{45}$$

where again $t = i/T$ and $\mathbf{z}_t = \alpha_t\mathbf{x} + \sigma_t\boldsymbol{\epsilon}$.

As $\tau \to 0, T \to \infty$, and letting $\text{SNR}'(t)$ denote the derivative of the SNR function, this then gives

$$\mathcal{L}_\infty(\mathbf{x}) = -\frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}),t\sim U[0,1]}\left[\text{SNR}'(t)||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t;t)||_2^2\right] \tag{46}$$

$$= -\frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})}\int_0^1 \text{SNR}'(t)\,\|\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t;t)\|_2^2\,dt. \tag{47}$$

F Influence of the number of steps T on the VLB

Recall that the diffusion loss for our choice of model p, q, when using T timesteps, is given by

$$\mathcal{L}_T(\mathbf{x}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})}\left[\sum_{i=1}^{T}(\text{SNR}(s(i)) - \text{SNR}(t(i)))\,||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_{t(i)};t(i))||_2^2\right], \tag{48}$$

with $s(i) = (i-1)/T$, $t(i) = i/T$.

This can then be written equivalently as

$$\mathcal{L}_T(\mathbf{x}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})}\left[\sum_{i=1}^{T}(\text{SNR}(s) - \text{SNR}(t') + \text{SNR}(t') - \text{SNR}(t))\,||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t;t)||_2^2\right], \tag{49}$$

with $t' = t - 0.5/T$.

In contrast, the diffusion loss with 2T timesteps can be written as

$$\mathcal{L}_{2T}(\mathbf{x}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})}\sum_{i=1}^{T}(\text{SNR}(s) - \text{SNR}(t'))\,||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_{t'};t')||_2^2 + \sum_{i=1}^{T}(\text{SNR}(t') - \text{SNR}(t))\,||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t;t)||_2^2. \tag{50}$$

Subtracting the two results, we get

$$\mathcal{L}_{2T}(\mathbf{x}) - \mathcal{L}_T(\mathbf{x}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})}\left[\sum_{i=1}^{T}(\text{SNR}(s) - \text{SNR}(t'))\left(||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_{t'};t')||_2^2 - ||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t;t)||_2^2\right)\right]. \tag{51}$$

Since $t' < t$, $\mathbf{z}_{t'}$ is a less noisy version of the data from earlier in the diffusion process compared to $\mathbf{z}_t$. Predicting the original data $\mathbf{x}$ from $\mathbf{z}_{t'}$ is thus strictly easier than from $\mathbf{z}_t$, leading to lower mean squared error if our model $\hat{\mathbf{x}}_{\boldsymbol{\theta}}$ is good enough. We thus have that $\mathcal{L}_{2T}(\mathbf{x}) - \mathcal{L}_T(\mathbf{x}) < 0$, which means that doubling the number of timesteps always improves our diffusion loss. For this reason we argue for using the continuous-time VLB corresponding to $T \to \infty$ in this paper.

G Equivalence of diffusion specifications

(Continuation of Section 5.1.)

Let $p^A(\mathbf{x})$ denote the distribution on observed data $\mathbf{x}$ as defined by the combination of a diffusion specification $\{\alpha_v^A, \sigma_v^A\}$ and denoising function $\tilde{\mathbf{x}}_{\boldsymbol{\theta}}^A$, and let $\mathbf{z}_v^A$ denote the latents of this diffusion process. Similarly, let $p^B(\mathbf{x}), \mathbf{z}_v^B$ be defined through a different specification $\{\alpha_v^B, \sigma_v^B, \tilde{\mathbf{x}}_{\boldsymbol{\theta}}^B\}$.

Since $v \equiv \alpha_v^2/\sigma_v^2$, we have that $\sigma_v = \alpha_v/\sqrt{v}$, which means that $\mathbf{z}_v(\mathbf{x}, \boldsymbol{\epsilon}) = \alpha_v \mathbf{x} + \sigma_v \boldsymbol{\epsilon} = \alpha_v(\mathbf{x} + \boldsymbol{\epsilon}/\sqrt{v})$. This holds for any diffusion specification by definition, and therefore we have $\mathbf{z}_v^A(\mathbf{x}, \boldsymbol{\epsilon}) = (\alpha_v^A/\alpha_v^B)\mathbf{z}_v^B(\mathbf{x}, \boldsymbol{\epsilon})$. The latents $\mathbf{z}_v$ for different diffusion specifications are thus identical, up to a trivial rescaling, and their information content only depends on the signal-to-noise ratio v, not on α_v, σ_v separately.

For the purpose of denoising from a latent $\mathbf{z}_v^B$, this means we can simply define the denoising model as $\tilde{\mathbf{x}}_{\boldsymbol{\theta}}^B(\mathbf{z}_v^B, v) \equiv \tilde{\mathbf{x}}_{\boldsymbol{\theta}}^A((\alpha_v^A/\alpha_v^B)\mathbf{z}_v^B, v)$, and we will then get the same reconstruction loss as when denoising from $\mathbf{z}_v^A$ using model $\tilde{\mathbf{x}}_{\boldsymbol{\theta}}^A$. Assuming endpoints SNR_{max} and SNR_{min} are equal for both specifications, Equation 18 then tells us that $\mathcal{L}_\infty^A(\mathbf{x}) = \mathcal{L}_\infty^B(\mathbf{x})$, i.e. they both produce the same diffusion loss in continuous time.

Similarly, the conditional model distributions over the latents $\mathbf{z}_v$ in our generative model are functions of the denoising model $\tilde{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_v, v)$ (see Equation 24), and we therefore have that the specification $\{\alpha_v^A, \sigma_v^A, \tilde{\mathbf{x}}_{\boldsymbol{\theta}}^A\}$ defines the same generative model over $\mathbf{z}_v, \mathbf{x}$ as the specification $\{\alpha_v^B, \sigma_v^B, \tilde{\mathbf{x}}_{\boldsymbol{\theta}}^B\}$, again up to a rescaling of the latents by α_v^A/α_v^B.

H Implementation of monotonic neural net noise schedule $\gamma_\eta(t)$

To learn the signal-to-noise ratio $\text{SNR}(t)$, we parameterize it as $\text{SNR}(t) = \exp(-\gamma_{\boldsymbol{\eta}}(t))$ with $\gamma_{\boldsymbol{\eta}}(t)$ a monotonic neural network. This network consists of 3 linear layers with weights that are restricted to be positive, l_1, l_2, l_3, which are composed as $\tilde{\gamma}_{\boldsymbol{\eta}}(t) = l_1(t) + l_3(\phi(l_2(l_1(t))))$, with ϕ the sigmoid function. Here, the l_2 layer has 1024 outputs, and the other layers have a single output.

In case of the continuous-time model, for the purpose of variance minimization, we postprocess the noise schedule as detailed in Appendix I.

I Variance minimization

Reduction of the variance diffusion loss estimator can lead to faster optimization. Note that this can be omitted if one aims for a simple implementation of our methods.

For the continuous-time model, we reduce the variance of the diffusion loss estimator through two methods: (1) optimizing the noise schedule w.r.t. the variance of the diffusion loss, and (2) using a low-discrepency sampler.

I.1 Low-discrepency sampler

When processing a minibatch of k examples $\mathbf{x}^i$, $i \in \{1, \ldots, k\}$, we require k timesteps t^i sampled from a uniform distribution. Instead of sampling these timesteps independently, we sample a single uniform random number $u_0 \sim U[0, 1]$ and then set $t^i = \text{mod}(u_0 + i/k, 1)$. Each t^i now has the correct uniform marginal distribution, but the minibatch of timesteps covers the space in $[0, 1]$ more equally than when sampling independently, which we find to reduce the variance in our VLB estimate.

I.2 Optimizing the noise schedule w.r.t. the variance of the diffusion loss

In case of the continuous-time model, for the purpose of variance minimization, we postprocess the noise schedule as follows. At this point, the range of $\tilde{\gamma}_{\boldsymbol{\eta}}(t)$ is unbounded and so the resulting SNR is not yet restricted to $[\text{SNR}_{\text{min}}, \text{SNR}_{\text{max}}]$. We therefore postprocess the monotonic neural network as

$$\gamma_{\boldsymbol{\eta}}(t) = \gamma_0 + (\gamma_1 - \gamma_0)\frac{\tilde{\gamma}_{\boldsymbol{\eta}}(t) - \tilde{\gamma}_{\boldsymbol{\eta}}(0)}{\tilde{\gamma}_{\boldsymbol{\eta}}(1) - \tilde{\gamma}_{\boldsymbol{\eta}}(0)}, \tag{52}$$

with $\gamma_0 = -\log(\text{SNR}_{\text{max}}), \gamma_1 = -\log(\text{SNR}_{\text{min}})$. Now $\text{SNR}(t) = \exp(-\gamma_{\boldsymbol{\eta}}(t))$ has the correct range and interpolates exactly between SNR_{min} and SNR_{max}. We treat γ_0, γ_1 as free parameters that we optimize directly w.r.t. the VLB. The remaining parameters $\boldsymbol{\eta}$ are instead learned by minimizing the variance of the stochastic estimate of the VLB.

We minimize the variance by performing stochastic gradient descent on our squared diffusion loss $\mathcal{L}_\infty^{MC}(\mathbf{x}, w, \gamma)^2$. We have that $\mathbb{E}_{t,\boldsymbol{\epsilon}}[\mathcal{L}_\infty^{MC}(\mathbf{x}, w, \gamma)^2] = \mathcal{L}_\infty(\mathbf{x}, w)^2 + \text{Var}_{t,\boldsymbol{\epsilon}}[\mathcal{L}_\infty^{MC}(\mathbf{x}, w, \gamma)]$, where the first part is independent of $\gamma_{\boldsymbol{\eta}}(t)$, and hence that

$$\mathbb{E}_{t,\boldsymbol{\epsilon}}[\nabla_{\boldsymbol{\eta}} \mathcal{L}_\infty^{MC}(\mathbf{x}, w, \gamma_{\boldsymbol{\eta}})^2] = \nabla_{\boldsymbol{\eta}} \text{Var}_{t,\boldsymbol{\epsilon}}[\mathcal{L}_\infty^{MC}(\mathbf{x}, w, \gamma_{\boldsymbol{\eta}})]. \tag{53}$$

We can calculate this gradient with negligible computational overhead as a by-product of calculating the gradient of the VLB, details of which are given in Appendix H.

We wish to calculate $\nabla_{\boldsymbol{\eta}}[\mathcal{L}_\infty^{MC}(\mathbf{x}, \gamma_{\boldsymbol{\eta}})^2]$ without performing a second backpropagation pass through the denoising model due to this objective being different than for the other parameters. To do this, we decompose the gradient as

$$\frac{d}{d\boldsymbol{\eta}}[\mathcal{L}_\infty^{MC}(\mathbf{x}, \gamma_{\boldsymbol{\eta}})^2] = \frac{d}{d\text{SNR}} \left[\mathcal{L}_\infty^{MC}(\mathbf{x}, \text{SNR})^2\right] \frac{d}{d\boldsymbol{\eta}} \left[\text{SNR}(\boldsymbol{\eta})\right], \tag{54}$$

$$\text{and} \quad \frac{d}{d\text{SNR}} \left[\mathcal{L}_\infty^{MC}(\mathbf{x}, \text{SNR})^2\right] = 2\frac{d}{d\text{SNR}} \left[\mathcal{L}_\infty^{MC}(\mathbf{x}, \text{SNR})\right] \odot \mathcal{L}_\infty^{MC}(\mathbf{x}, \text{SNR}), \tag{55}$$

where $\odot$ denotes elementwise multiplication. Here $\frac{d}{d\text{SNR}} \left[\mathcal{L}_\infty^{MC}(\mathbf{x}, \text{SNR})\right]$ is computed along with the other gradients when performing the single backpropagation pass for calculating $\nabla_{\boldsymbol{\theta}}[\mathcal{L}_\infty^{MC}]$. The remaining operations required to get $\nabla_{\boldsymbol{\eta}}[\mathcal{L}_\infty^{MC}(\mathbf{x}, \gamma_{\boldsymbol{\eta}})^2]$ have negligible computational cost.

This strategy of minimizing the variance of our diffusion loss estimate remains valid for weighted diffusion losses, $w(v) \neq 1$, not corresponding to the VLB, and we therefore expect it to be useful beyond the goal of optimizing for likelihood that we consider in this paper.

J Numerical stability

Floating point numbers are much worse at representing numbers close to 1, than at representing numbers close to 0. Since a naïve implementation of our model and its discrete-time loss function requires computing intermediate values that are close to 1, those numbers are erroneously rounded to 1, leading to numerical issues and incorrect results. Note that previous implementations of discrete-time diffusion models (e.g. [Ho et al., 2020]) used 64-bit floating point numbers to avoid numerical problems. We found this unnecessary in our model.

A numerically problematic term, for example, is $\boldsymbol{\sigma}_{t|s}^2$ which is used for sampling. It is straightforward to verify that:

$$\boldsymbol{\sigma}_{t|s}^2 = -\text{expm1}(\text{softplus}(\gamma(s)) - \text{softplus}(\gamma(t))), \tag{56}$$

where $\text{expm1}(x) \equiv \exp(x) - 1$ and $\text{softplus}(x) \equiv \log(1 + \exp(x))$ are functions with numerically stable primitives in common numerical computing packages.

K Comparison to DDPM and NCSN objectives

Previous works using denoising diffusion models [Ho et al., 2020, Song and Ermon, 2019, Nichol and Dhariwal, 2021] used a training objective that can be understood as a *weighted* diffusion loss of the form given in Equation 19:

$$\mathcal{L}_\infty(\mathbf{x}, w) = \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})} \int_{\text{SNR}_{\text{min}}}^{\text{SNR}_{\text{max}}} w(v) \left\|\mathbf{x} - \tilde{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_v, v)\right\|_2^2 dv \tag{57}$$

$$= -\frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})} \int_0^1 \text{SNR}'(t) w(\text{SNR}(t)) \left\|\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)\right\|_2^2 dt \tag{58}$$

$$= \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})} \int_0^1 \gamma'(t) w(\exp(-\gamma(t))) \left\|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)\right\|_2^2 dt, \tag{59}$$

where $\gamma(t) = -\log \text{SNR}(t)$.

When using the loss in Equation 57, we set $w(v) = 1$, corresponding to optimization of a variational bound on the likelihood of the data. Ho et al. [2020], Song and Ermon [2019], Nichol and Dhariwal [2021] instead choose to minimize the *simple objective* defined as

$$L_{\text{simple}}(\mathbf{x}) \equiv \int_0^1 \|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t, t)\|_2^2 dt, \tag{60}$$

or a discrete-time version of this.

Comparing Equation 60 with Equation 59, we can see that the loss used by Ho et al. [2020], Song and Ermon [2019], Nichol and Dhariwal [2021] corresponds to a weighting function $w(\text{SNR}(t)) = 1/\gamma'(t)$. Below, we derive the $\gamma(t)$, and thus the weighting function $w(v)$, corresponding to the diffusion processes used by Ho et al. [2020], Song and Ermon [2019], Nichol and Dhariwal [2021]. We visualize these weighting functions in Figure 6.

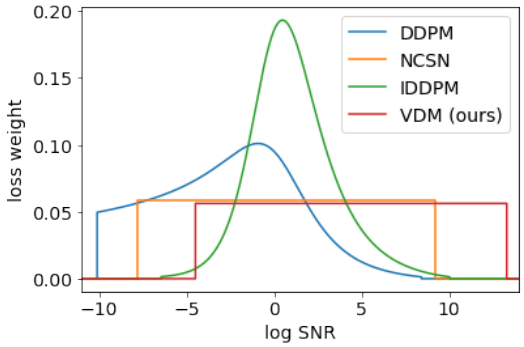


Figure 6: Implied weighting functions corresponding to the losses used by Ho et al. [2020], Song et al. [2020], and Nichol and Dhariwal [2021], as well as our proposed loss. NCSN [Song et al., 2020] uses a constant implied weighting function, and is thus consistent with maximization of the variational bound like we propose in this paper. However, unlike Song and Ermon [2019] we also learn the endpoints $\text{SNR}_{\text{min}}, \text{SNR}_{\text{max}}$, which results in a better optimized VLB value. DDPM [Ho et al., 2020] and improved DDPM [Nichol and Dhariwal, 2021] instead use implied weighting functions that put relatively more weight on the noisy data with low to medium signal-to-noise ratio. The latter two works report better FID and Inception Score than Song et al. [2020] and the current paper, which we hypothesize is due to their loss emphasizing the global consistence and coarse level patterns more than the fine scale features of the data.

For DDPM, Ho et al. [2020] use a diffusion process in discrete time with $\alpha_i = \sqrt{\prod_{j=1}^{i}(1-\beta_j)}$, $\sigma_i^2 = 1 - \alpha_i^2$, where β_i linearly interpolates between $\beta_1 = 1e^{-4}$ and $\beta_T = 0.02$ in $T = 1000$ discrete steps. When defining time $t = i/T$, this can be closely approximated as $\alpha_t^2 = \exp(-1e^{-4} - 10t^2)$, and correspondingly with $\text{SNR}(t) = 1/\text{expm1}(1e^{-4} + 10t^2)$ or $\gamma(t) = \log[\text{expm1}(1e^{-4} + 10t^2)]$, where $\text{expm1}(x) = \exp(x) - 1$. This approximation is shown in Figure 7.

For NCSNv2, Song and Ermon [2020] instead use $\alpha_t = 1$ and let σ_t be a geometric series interpolating between 0.01 and 50, i.e. $\sigma_t^2 = \exp(\gamma(t))$ with $\gamma(t) = 2\log[0.01] + 2\log[5000]t$. This means that $\gamma'(t) = 2\log[5000]$ and thus that $w(v)$ is a constant. The procedure proposed by Song and Ermon [2020] is thus consistent with maximization of the VLB like we propose here. The same holds for [Song and Ermon, 2019].

For IDDPM, Nichol and Dhariwal [2021] use $\tilde{\alpha}_t = \cos(\frac{t+0.008}{1.008}\frac{\pi}{2}) / \cos(\frac{0.008}{1.008}\frac{\pi}{2})$. The values for $\tilde{\alpha}_t$ are then translated into value for β_t, which are then clipped to 0.999. Subsequently we can then derive the $\alpha_t, \sigma_t, \gamma(t)$ corresponding to those β_t. Due to the clipping these expressions do not simplify, but we include their numerical results in Figure 6.

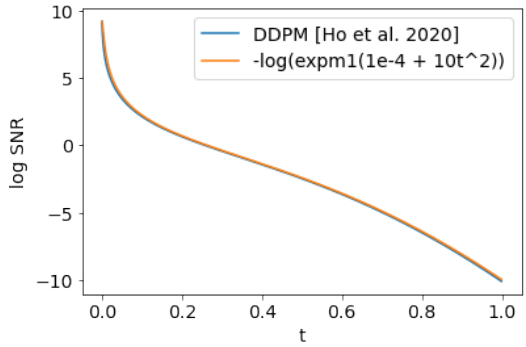


Figure 7: Log signal-to-noise ratio for the discrete-time diffusion process in Ho et al. [2020] and our continous-time approximation.

L Consistency

Let $q(\mathbf{x})$ denote the marginal distribution of data $\mathbf{x}$, and let:

$$q(\mathbf{z}_t) = \int q(\mathbf{z}_t|\mathbf{x})q(\mathbf{x})d\mathbf{x} \tag{61}$$

Here we will show that derived estimators are *consistent* estimators, in the sense that with infinite data, the optimal score model $\mathbf{s}^*_{\boldsymbol{\theta}}(\mathbf{z}_t; t)$ is such that:

$$\mathbf{s}^*_{\boldsymbol{\theta}}(\mathbf{z}_t; t) = \nabla_{\mathbf{z}} \log q(\mathbf{z}_t) \tag{62}$$

Note that $\nabla_{\mathbf{z}_t} \log q(\mathbf{z}_t|\mathbf{x}) = -\boldsymbol{\epsilon}/\boldsymbol{\sigma}_t$. We can rewrite the diffusion loss (discrete time or continuous time) for timestep t as:

$$\mathcal{L}_T(\mathbf{x}; t) = \frac{1}{2}\mathbb{E}_{q(\mathbf{x},\mathbf{z}_t)}\left[\left\|\sqrt{\mathbf{c}(t)}\left(\nabla_{\mathbf{z}_t} \log q(\mathbf{z}_t|\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)\right)\right\|_2^2\right] \tag{63}$$

where $\sqrt{\mathbf{c}(t)}$ is a time-dependent weighting factor.

In [Ho et al., 2020], it is noted that the discrete-time VLB, when using equal variances across dimensions, is equivalent to a Denoising Score Matching (DSM) objective [Vincent, 2011]. This is interesting, since it implies consistency. We generalize this original consistency proof of DSM to a more general case of different noises schedules per dimension, and arbitrary multipliers $\sqrt{\mathbf{c}_1}$ and $\sqrt{\mathbf{c}_2}$ in front of the scores, i.e. where the dimensions of $\mathbf{z}$ are differently weighted. Note, however that we'll only need the special case where $\sqrt{\mathbf{c}} = \sqrt{\mathbf{c}_1} = \sqrt{\mathbf{c}_2}$. First, note that:

$$\begin{aligned}\tfrac{1}{2}\mathbb{E}_{q(\mathbf{z}_t)}\left[||\sqrt{\mathbf{c}_1}\nabla_{\mathbf{z}_t} \log q(\mathbf{z}_t) - \sqrt{\mathbf{c}_2}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t)||_2^2\right] &= \tfrac{1}{2}\mathbb{E}_{q(\mathbf{z}_t)}\left[||\sqrt{\mathbf{c}_1}\nabla_{\mathbf{z}_t} \log q(\mathbf{z}_t)||_2^2\right] \\ &+ \tfrac{1}{2}\mathbb{E}_{q(\mathbf{z}_t)}\left[||\sqrt{\mathbf{c}_2}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t)||_2^2\right] \\ &- \mathbb{E}_{q(\mathbf{z}_t)}\left[\langle\sqrt{\mathbf{c}_1}\nabla_{\mathbf{z}_t} \log q(\mathbf{z}_t), \sqrt{\mathbf{c}_2}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t)\rangle\right]\end{aligned} \tag{64}$$

Where $\langle ., . \rangle$ denotes a dot product. Similarly:

$$\begin{aligned}\tfrac{1}{2}\mathbb{E}_{q(\mathbf{z}_t|\mathbf{x})}\left[||\sqrt{\mathbf{c}_1}\nabla_{\mathbf{z}_t} \log q(\mathbf{z}_t|\mathbf{x}) - \sqrt{\mathbf{c}_2}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t)||_2^2\right] &= \tfrac{1}{2}\mathbb{E}_{q(\mathbf{z}_t|\mathbf{x})}\left[||\sqrt{\mathbf{c}_1}\nabla_{\mathbf{z}_t} \log q(\mathbf{z}_t|\mathbf{x})||_2^2\right] \\ &+ \tfrac{1}{2}\mathbb{E}_{q(\mathbf{z}_t|\mathbf{x})}\left[||\sqrt{\mathbf{c}_2}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t)||_2^2\right] \\ &- \mathbb{E}_{q(\mathbf{z}_t|\mathbf{x})}\left[\langle\sqrt{\mathbf{c}_1}\nabla_{\mathbf{z}_t} \log q(\mathbf{z}_t|\mathbf{x}), \sqrt{\mathbf{c}_2}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t)\rangle\right]\end{aligned} \tag{65}$$

The second terms of the right-hand sides of Equation 64 and Equation 65 are equal. The third terms of the right-hand sides of Equation 64 and Equation 65 are also equal:

$$\mathbb{E}_{q(\mathbf{z}_t)}\left[\langle\sqrt{\mathbf{c}_2}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t), \sqrt{\mathbf{c}_1}\nabla_{\mathbf{z}_t}\log q(\mathbf{z}_t)\rangle\right] \tag{66}$$

$$= \mathbb{E}_{q(\mathbf{z}_t)}\left[\left\langle\sqrt{\mathbf{c}_2}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t), \sqrt{\mathbf{c}_1}\frac{\nabla_{\mathbf{z}_t}q(\mathbf{z}_t)}{q(\mathbf{z}_t)}\right\rangle\right] \tag{67}$$

$$= \int_{\mathbf{z}_t}\langle\sqrt{\mathbf{c}_2}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t), \sqrt{\mathbf{c}_1}\nabla_{\mathbf{z}_t}q(\mathbf{z}_t)\rangle\, d\mathbf{z}_t \tag{68}$$

$$= \int_{\mathbf{z}_t}\left\langle\sqrt{\mathbf{c}_2}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t), \sqrt{\mathbf{c}_1}\nabla_{\mathbf{z}_t}\int_{\mathbf{x}}q(\mathbf{x})q(\mathbf{z}_t|\mathbf{x})d\mathbf{x}\right\rangle d\mathbf{z}_t \tag{69}$$

$$= \int_{\mathbf{z}_t}\left\langle\sqrt{\mathbf{c}_2}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t), \sqrt{\mathbf{c}_1}\int_{\mathbf{x}}q(\mathbf{x})\nabla_{\mathbf{z}_t}q(\mathbf{z}_t|\mathbf{x})d\mathbf{x}\right\rangle d\mathbf{z}_t \tag{70}$$

$$= \int_{\mathbf{z}_t}\left\langle\sqrt{\mathbf{c}_2}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t), \sqrt{\mathbf{c}_1}\int_{\mathbf{x}}q(\mathbf{x})q(\mathbf{z}_t|\mathbf{x})\nabla_{\mathbf{z}_t}\log q(\mathbf{z}_t|\mathbf{x})d\mathbf{x}\right\rangle d\mathbf{z}_t \tag{71}$$

$$= \int_{\mathbf{x}}\int_{\mathbf{z}_t}q(\mathbf{x})q(\mathbf{z}_t|\mathbf{x})\,\langle\sqrt{\mathbf{c}_2}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t), \sqrt{\mathbf{c}_1}\nabla_{\mathbf{z}_t}\log q(\mathbf{z}_t|\mathbf{x})\rangle\, d\mathbf{z}_t d\mathbf{x} \tag{72}$$

$$= \mathbb{E}_{q(\mathbf{x},\mathbf{z}_t)}\left[\langle\sqrt{\mathbf{c}_2}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t), \sqrt{\mathbf{c}_1}\nabla_{\mathbf{z}_t}\log q(\mathbf{z}_t|\mathbf{x})\rangle\right] \tag{73}$$

So, only the first term of the right-hand sides of Equation 64 and Equation 65 are not equal. It follows that:

$$\tfrac{1}{2}\mathbb{E}_{q(\mathbf{z}_t)}\left[||\sqrt{\mathbf{c}_1}\nabla_{\mathbf{z}_t}\log q(\mathbf{z}_t) - \sqrt{\mathbf{c}_2}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t)||_2^2\right] = \tfrac{1}{2}\mathbb{E}_{q(\mathbf{x},\mathbf{z}_t)}\left[||\sqrt{\mathbf{c}_1}\nabla_{\mathbf{z}_t}\log q(\mathbf{z}_t|\mathbf{x}) - \sqrt{\mathbf{c}_2}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t)||_2^2\right] + \text{constant} \tag{74}$$

where constant $= \frac{1}{2}\mathbb{E}_{q(\mathbf{z}_t)}\left[||\sqrt{\mathbf{c}_1}\nabla_{\mathbf{z}_t}\log q(\mathbf{z}_t)||_2^2\right] - \frac{1}{2}\mathbb{E}_{q(\mathbf{x},\mathbf{z}_t)}\left[||\sqrt{\mathbf{c}_1}\nabla_{\mathbf{z}_t}\log q(\mathbf{z}_t|\mathbf{x})||_2^2\right]$ is constant w.r.t. the energy-based model (EBM) $E()$. In the special case where $\sqrt{\mathbf{c}_1} = \sqrt{\mathbf{c}_2}$, we have:

$$\tfrac{1}{2}\mathbb{E}_{q(\mathbf{z}_t)}\left[||\sqrt{\mathbf{c}_1}(\nabla_{\mathbf{z}_t}\log q(\mathbf{z}_t) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t))||_2^2\right] = \tfrac{1}{2}\mathbb{E}_{q(\mathbf{x},\mathbf{z}_t)}\left[||\sqrt{\mathbf{c}_1}(\nabla_{\mathbf{z}_t}\log q(\mathbf{z}_t|\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{z}_t))||_2^2\right] + \text{constant} \tag{75}$$

Therefore, minimizing the first term on the right-hand side of Equation 75 w.r.t. $E()$ (a denoising score matching objective with differently weighted dimensions) is equivalent to minimizing the left-hand side of Equation 75 w.r.t. $E()$. From this equation, it is clear that at the optimum of this DSM objective, for any positive $\mathbf{c}_1$:

$$\mathbf{s}^*_{\boldsymbol{\theta}}(\mathbf{z}_t) = \nabla_{\mathbf{z}_t}\log q(\mathbf{z}_t) \tag{76}$$

If the score model is parameterized as the gradient of an EBM $E()$, then this implies that for all $t \in [0, 1]$:

$$\exp(-E^*(\mathbf{z}_t; t)) \propto q(\mathbf{z}_t) \tag{77}$$

So, when optimizing for the diffusion loss, the EBM $E(.; t)$ will approximate the correct marginals corresponding the inference model.

M Additional samples from our models

We include additional uncurated random samples from our unconditional models trained on CIFAR-10, 32x32 Imagenet, and 64x64 Imagenet. See Figures 8, 9, and 10.

N Lossless compression

For a fixed number of evaluation timesteps T_{eval}, our diffusion model in discrete time is a hierarchical latent variable model that can be turned into a lossless compression algorithm using bits-back coding [Hinton and Van Camp, 1993]. Assuming a source of auxiliary random bits is available alongside the data, bits-back coding encodes a latent and data together, with the latent sampled from the approximate posterior using the auxiliary random bits. The net coding cost of bits-back coding is

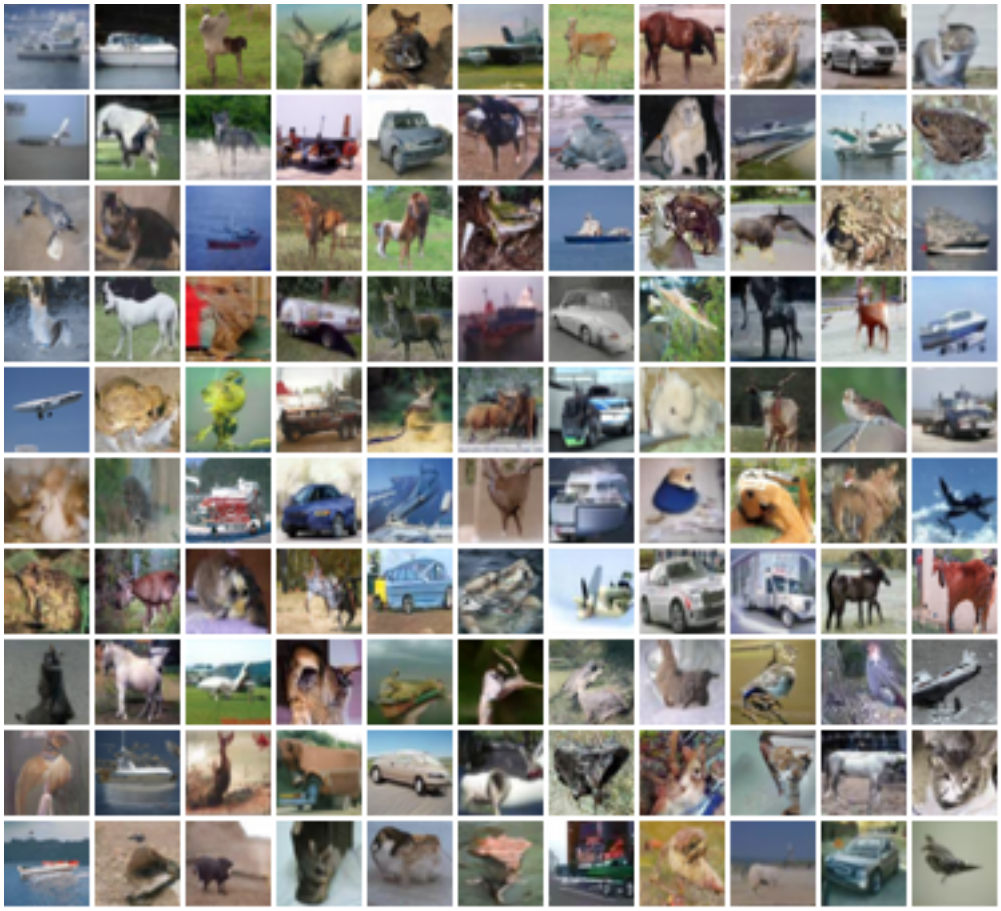


Figure 8: Random samples from an unconditional diffusion model trained on CIFAR-10 for 2 million parameter updates. The model was trained in continuous-time, and sampled using $T = 1000$.

given by subtracting the number of bits needed to sample the latent from the number of bits needed to encode the latent and data using the reverse process, so the negative VLB of our discrete time model is the theoretical expected coding cost for bits-back coding.

As a proof of concept for lossless compression using our model, Table 2 reports net codelengths on the CIFAR10 test set for various settings of T_{eval} using BB-ANS [Townsend et al., 2018], a practical implementation of bits-back coding based on asymmetric numeral systems [Duda, 2009]. Since diffusion models have Markov forward and reverse processes, we use the Bit-Swap implementation of BB-ANS [Kingma et al., 2019]. Practical implementations of bits-back coding must discretize continuous latent variables and their associated continuous probability distributions; for simplicity, our implementation uses a uniform discretization of the continuous latents and their associated Gaussian conditionals from the forward and reverse processes. Additionally, we found it crucial to encrypt the ANS bitstream before each decoding operation to ensure clean bits for sampling from the approximate posterior; we did so by applying the XOR operation to the ANS bitstream with pseudorandom bits from a fixed sequence of seeds. For example, without cleaning the bitstream using encryption, compressing a batch of 100 examples using $T_{eval} = 250$ costs 2.74 bits per byte, but with encryption, the cost improves to 2.68 bits per dimension.

For a small number of timesteps T_{eval}, our bits-back implementation attains net codelengths that agree closely with the negative VLB, but there is some discrepancy for large T_{eval}. This is due to inaccuracies in the compression algorithm to represent discretized Gaussians with small standard deviations, and small discrepancies in codelength compound into a gap of up to 0.05 bits per dimension when T is large. (In prior work, e.g. [Kingma et al., 2019, Ho et al., 2019b, Townsend et al., 2020], practical implementations of bits-back coding have been tested on latent variable models

with only tens of layers, not hundreds.) In addition, a large number of timesteps makes compression computationally expensive, because a neural network forward pass must be run for each timestep. Closing the codelength gap with an efficient implementation of bits-back coding for a large number of timesteps is an interesting avenue for future work.

O Density estimation on additional data sets

At the request of one of the reviewers we also ran our model on additional data sets of higher resolution, less diverse, images. Specifically, we obtain a test set likelihood of 2.14 bits per dim on CelebA-HQ [Karras et al., 2017], and 1.44 on LSUN bedrooms [Yu et al., 2015], both at 128×128 resolution. Since these are not established benchmarks in density estimation, and since downsampling methods in the literature are not consistent, we don't compare against previous methods for these data sets. Our results are provided purely to give a ballpark estimate of how well our proposed method scales to higher resolution images.

The model used for these data sets is based on that used for Imagenet 64×64, with an additional level in the UNet at resolution 128×128, consisting of 16 residual layers using 128 channels. Our model downsamples between the 128×128 and 64×64 resolutions, similar to e.g. Ho et al. [2020], but unlike the models we used for the other data sets we considered.

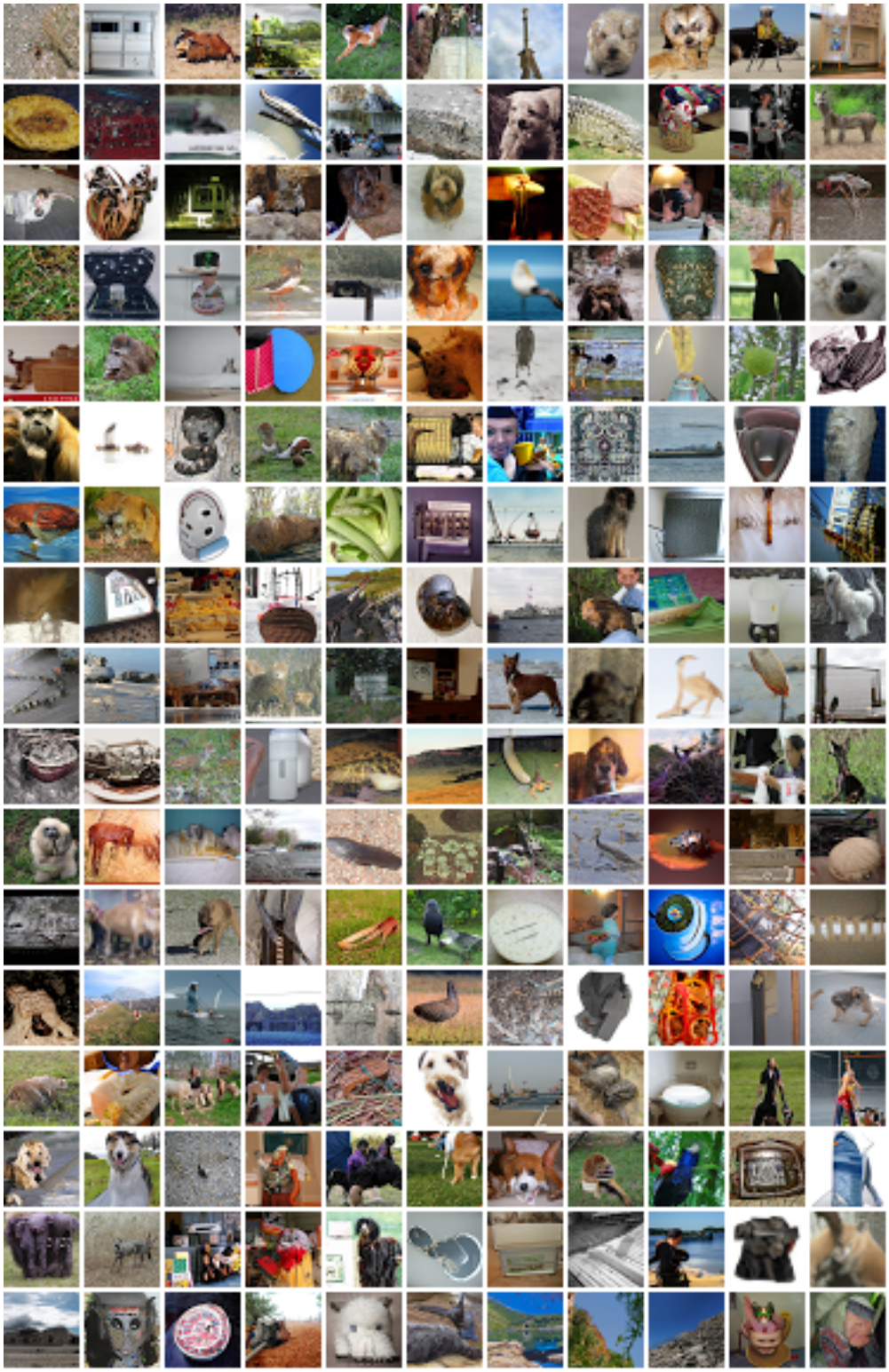


Figure 9: Random samples from an unconditional diffusion model trained on 32x32 ImageNet for 3.7 million parameter updates. The model was trained in continuous-time, and sampled using $T = 1000$.

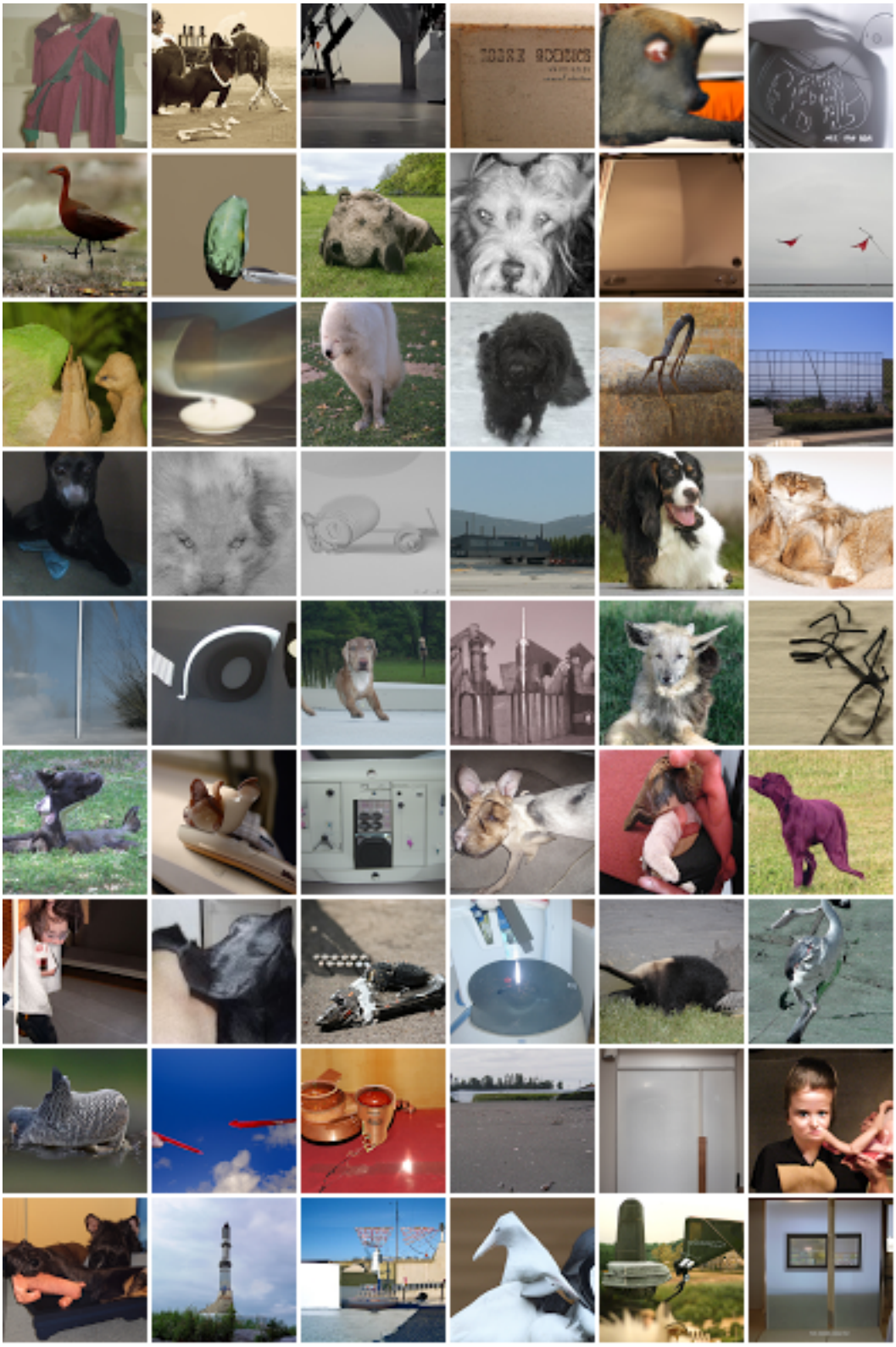


Figure 10: Random samples from an unconditional diffusion model trained on 64x64 ImageNet for 2 million parameter updates. The model was trained in continuous-time, and sampled using $T = 1000$.