Denoising Diffusion Probabilistic Models

Jonathan Ho
UC Berkeley
jonathanho@berkeley.edu

Ajay Jain
UC Berkeley
ajayj@berkeley.edu

Pieter Abbeel
UC Berkeley
pabbeel@cs.berkeley.edu

Abstract

We present high quality image synthesis results using diffusion probabilistic models, a class of latent variable models inspired by considerations from nonequilibrium thermodynamics. Our best results are obtained by training on a weighted variational bound designed according to a novel connection between diffusion probabilistic models and denoising score matching with Langevin dynamics, and our models naturally admit a progressive lossy decompression scheme that can be interpreted as a generalization of autoregressive decoding. On the unconditional CIFAR10 dataset, we obtain an Inception score of 9.46 and a state-of-the-art FID score of 3.17. On 256x256 LSUN, we obtain sample quality similar to ProgressiveGAN. Our implementation is available at https://github.com/hojonathanho/diffusion

1 Introduction

Deep generative models of all kinds have recently exhibited high quality samples in a wide variety of data modalities. Generative adversarial networks (GANs), autoregressive models, flows, and variational autoencoders (VAEs) have synthesized striking image and audio samples [14, 27, 3, 58, 38, 25, 10, 32, 44, 57, 26, 33, 45], and there have been remarkable advances in energy-based modeling and score matching that have produced images comparable to those of GANs [11, 55].

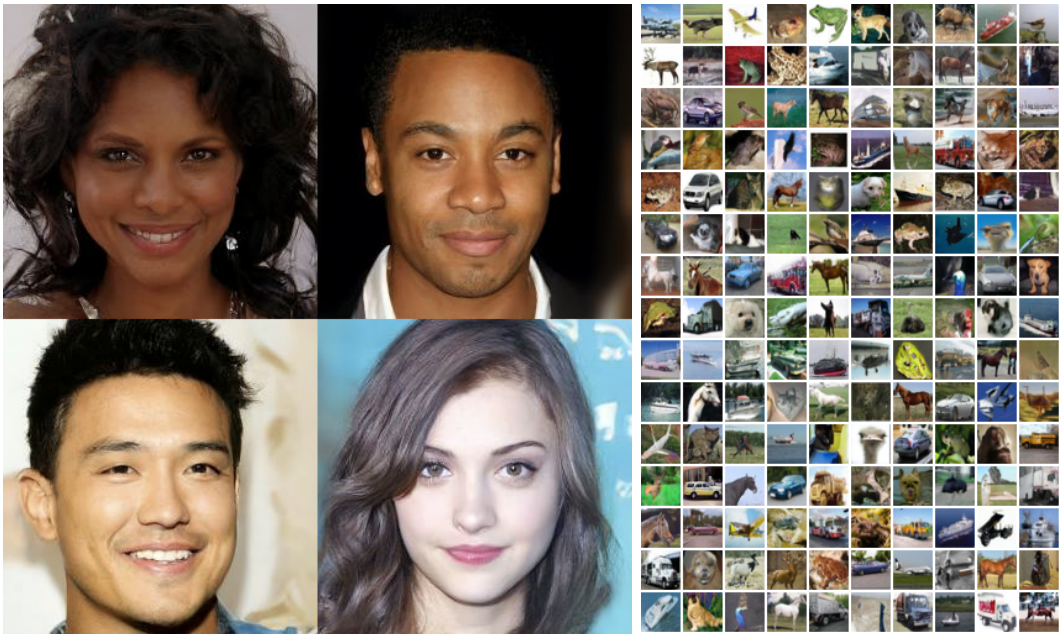


Figure 1: Generated samples on CelebA-HQ 256×256 (left) and unconditional CIFAR10 (right)

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

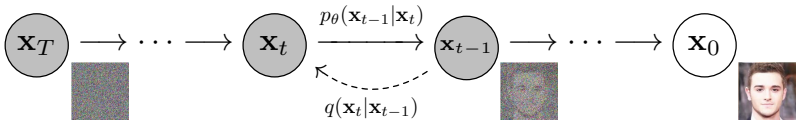


Figure 2: The directed graphical model considered in this work.

This paper presents progress in diffusion probabilistic models [53]. A diffusion probabilistic model (which we will call a "diffusion model" for brevity) is a parameterized Markov chain trained using variational inference to produce samples matching the data after finite time. Transitions of this chain are learned to reverse a diffusion process, which is a Markov chain that gradually adds noise to the data in the opposite direction of sampling until signal is destroyed. When the diffusion consists of small amounts of Gaussian noise, it is sufficient to set the sampling chain transitions to conditional Gaussians too, allowing for a particularly simple neural network parameterization.

Diffusion models are straightforward to define and efficient to train, but to the best of our knowledge, there has been no demonstration that they are capable of generating high quality samples. We show that diffusion models actually are capable of generating high quality samples, sometimes better than the published results on other types of generative models (Section 4). In addition, we show that a certain parameterization of diffusion models reveals an equivalence with denoising score matching over multiple noise levels during training and with annealed Langevin dynamics during sampling (Section 3.2) [55, 61]. We obtained our best sample quality results using this parameterization (Section 4.2), so we consider this equivalence to be one of our primary contributions.

Despite their sample quality, our models do not have competitive log likelihoods compared to other likelihood-based models (our models do, however, have log likelihoods better than the large estimates annealed importance sampling has been reported to produce for energy based models and score matching [11, 55]). We find that the majority of our models' lossless codelengths are consumed to describe imperceptible image details (Section 4.3). We present a more refined analysis of this phenomenon in the language of lossy compression, and we show that the sampling procedure of diffusion models is a type of progressive decoding that resembles autoregressive decoding along a bit ordering that vastly generalizes what is normally possible with autoregressive models.

2 Background

去噪

Diffusion models [53] are latent variable models of the form $p_\theta(\mathbf{x}_0) := \int p_\theta(\mathbf{x}_{0:T})\, d\mathbf{x}_{1:T}$, where $\mathbf{x}_1, \ldots, \mathbf{x}_T$ are latents of the same dimensionality as the data $\mathbf{x}_0 \sim q(\mathbf{x}_0)$. The joint distribution $p_\theta(\mathbf{x}_{0:T})$ is called the *reverse process*, and it is defined as a Markov chain with learned Gaussian transitions starting at $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$:

$$p_\theta(\mathbf{x}_{0:T}) := p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t), \quad p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) := \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)) \tag{1}$$

1. 2.

加噪

What distinguishes diffusion models from other types of latent variable models is that the approximate posterior $q(\mathbf{x}_{1:T}|\mathbf{x}_0)$, called the *forward process* or *diffusion process*, is fixed to a Markov chain that gradually adds Gaussian noise to the data according to a variance schedule $\beta_1, \ldots, \beta_T$:

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) := \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}), \quad q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}) \tag{2}$$

3. 4.

Training is performed by optimizing the usual variational bound on negative log likelihood:

loss

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] =: L \tag{3}$$

5.

加噪过程简化

The forward process variances β_t can be learned by reparameterization [33] or held constant as hyperparameters, and expressiveness of the reverse process is ensured in part by the choice of Gaussian conditionals in $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, because both processes have the same functional form when β_t are small [53]. A notable property of the forward process is that it admits sampling $\mathbf{x}_t$ at an arbitrary timestep t in closed form: using the notation $\alpha_t := 1 - \beta_t$ and $\bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$, we have

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}) \tag{4}$$

6.

Efficient training is therefore possible by optimizing random terms of L with stochastic gradient descent. Further improvements come from variance reduction by rewriting L (3) as:

$$\mathbb{E}_q\Big[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0)\parallel p(\mathbf{x}_T))}_{L_T}+\sum_{t>1}\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)\parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}}\underbrace{-\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\Big] \tag{5}$$

(See Appendix A for details. The labels on the terms are used in Section 3.) Equation (5) uses KL divergence to directly compare $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ against forward process posteriors, which are tractable when conditioned on $\mathbf{x}_0$:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)=\mathcal{N}(\mathbf{x}_{t-1};\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t,\mathbf{x}_0),\tilde{\beta}_t\mathbf{I}), \tag{6}$$

$$\text{where}\quad \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t,\mathbf{x}_0):=\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0+\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t \quad\text{and}\quad \tilde{\beta}_t:=\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \tag{7}$$

Consequently, all KL divergences in Eq. (5) are comparisons between Gaussians, so they can be calculated in a Rao-Blackwellized fashion with closed form expressions instead of high variance Monte Carlo estimates.

3 Diffusion models and denoising autoencoders

Diffusion models might appear to be a restricted class of latent variable models, but they allow a large number of degrees of freedom in implementation. One must choose the variances β_t of the forward process and the model architecture and Gaussian distribution parameterization of the reverse process. To guide our choices, we establish a new explicit connection between diffusion models and denoising score matching (Section 3.2) that leads to a simplified, weighted variational bound objective for diffusion models (Section 3.4). Ultimately, our model design is justified by simplicity and empirical results (Section 4). Our discussion is categorized by the terms of Eq. (5).

3.1 Forward process and L_T

We ignore the fact that the forward process variances β_t are learnable by reparameterization and instead fix them to constants (see Section 4 for details). Thus, in our implementation, the approximate posterior q has no learnable parameters, so L_T is a constant during training and can be ignored.

3.2 Reverse process and $L_{1:T-1}$

Now we discuss our choices in $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)=\mathcal{N}(\mathbf{x}_{t-1};\boldsymbol{\mu}_\theta(\mathbf{x}_t,t),\boldsymbol{\Sigma}_\theta(\mathbf{x}_t,t))$ for $1<t\le T$. First, we set $\boldsymbol{\Sigma}_\theta(\mathbf{x}_t,t)=\sigma_t^2\mathbf{I}$ to untrained time dependent constants. Experimentally, both $\sigma_t^2=\beta_t$ and $\sigma_t^2=\tilde{\beta}_t=\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$ had similar results. The first choice is optimal for $\mathbf{x}_0\sim\mathcal{N}(\mathbf{0},\mathbf{I})$, and the second is optimal for $\mathbf{x}_0$ deterministically set to one point. These are the two extreme choices corresponding to upper and lower bounds on reverse process entropy for data with coordinatewise unit variance [53].

Second, to represent the mean $\boldsymbol{\mu}_\theta(\mathbf{x}_t,t)$, we propose a specific parameterization motivated by the following analysis of L_t. With $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)=\mathcal{N}(\mathbf{x}_{t-1};\boldsymbol{\mu}_\theta(\mathbf{x}_t,t),\sigma_t^2\mathbf{I})$, we can write:

$$L_{t-1}=\mathbb{E}_q\left[\frac{1}{2\sigma_t^2}\|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t,\mathbf{x}_0)-\boldsymbol{\mu}_\theta(\mathbf{x}_t,t)\|^2\right]+C \tag{8}$$

where C is a constant that does not depend on θ. So, we see that the most straightforward parameterization of $\boldsymbol{\mu}_\theta$ is a model that predicts $\tilde{\boldsymbol{\mu}}_t$, the forward process posterior mean. However, we can expand Eq. (8) further by reparameterizing Eq. (4) as $\mathbf{x}_t(\mathbf{x}_0,\boldsymbol{\epsilon})=\sqrt{\bar{\alpha}_t}\mathbf{x}_0+\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$ for $\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})$ and applying the forward process posterior formula (7):

$$L_{t-1}-C=\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\frac{1}{2\sigma_t^2}\left\|\tilde{\boldsymbol{\mu}}_t\left(\mathbf{x}_t(\mathbf{x}_0,\boldsymbol{\epsilon}),\frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t(\mathbf{x}_0,\boldsymbol{\epsilon})-\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon})\right)-\boldsymbol{\mu}_\theta(\mathbf{x}_t(\mathbf{x}_0,\boldsymbol{\epsilon}),t)\right\|^2\right] \tag{9}$$

$$=\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\frac{1}{2\sigma_t^2}\left\|\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t(\mathbf{x}_0,\boldsymbol{\epsilon})-\frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}\right)-\boldsymbol{\mu}_\theta(\mathbf{x}_t(\mathbf{x}_0,\boldsymbol{\epsilon}),t)\right\|^2\right] \tag{10}$$

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

Equation (10) reveals that $\boldsymbol{\mu}_\theta$ must predict $\frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon} \right)$ given $\mathbf{x}_t$. Since $\mathbf{x}_t$ is available as input to the model, we may choose the parameterization

$$\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \tilde{\boldsymbol{\mu}}_t \left(\mathbf{x}_t, \frac{1}{\sqrt{\bar{\alpha}_t}} (\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t)) \right) = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) \tag{11}$$

where $\boldsymbol{\epsilon}_\theta$ is a function approximator intended to predict $\boldsymbol{\epsilon}$ from $\mathbf{x}_t$. To sample $\mathbf{x}_{t-1} \sim p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is to compute $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$, where $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The complete sampling procedure, Algorithm 2, resembles Langevin dynamics with $\boldsymbol{\epsilon}_\theta$ as a learned gradient of the data density. Furthermore, with the parameterization (11), Eq. (10) simplifies to:

$$\mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}} \left[\frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1-\bar{\alpha}_t)} \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2 \right] \tag{12}$$

which resembles denoising score matching over multiple noise scales indexed by t [55]. As Eq. (12) is equal to (one term of) the variational bound for the Langevin-like reverse process (11), we see that optimizing an objective resembling denoising score matching is equivalent to using variational inference to fit the finite-time marginal of a sampling chain resembling Langevin dynamics.

To summarize, we can train the reverse process mean function approximator $\boldsymbol{\mu}_\theta$ to predict $\tilde{\boldsymbol{\mu}}_t$, or by modifying its parameterization, we can train it to predict $\boldsymbol{\epsilon}$. (There is also the possibility of predicting $\mathbf{x}_0$, but we found this to lead to worse sample quality early in our experiments.) We have shown that the $\boldsymbol{\epsilon}$-prediction parameterization both resembles Langevin dynamics and simplifies the diffusion model's variational bound to an objective that resembles denoising score matching. Nonetheless, it is just another parameterization of $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, so we verify its effectiveness in Section 4 in an ablation where we compare predicting $\boldsymbol{\epsilon}$ against predicting $\tilde{\boldsymbol{\mu}}_t$.

3.3 Data scaling, reverse process decoder, and L_0

We assume that image data consists of integers in $\{0, 1, \dots, 255\}$ scaled linearly to $[-1, 1]$. This ensures that the neural network reverse process operates on consistently scaled inputs starting from the standard normal prior $p(\mathbf{x}_T)$. To obtain discrete log likelihoods, we set the last term of the reverse process to an independent discrete decoder derived from the Gaussian $\mathcal{N}(\mathbf{x}_0; \boldsymbol{\mu}_\theta(\mathbf{x}_1, 1), \sigma_1^2 \mathbf{I})$:

$$\begin{aligned} p_\theta(\mathbf{x}_0|\mathbf{x}_1) &= \prod_{i=1}^{D} \int_{\delta_-(x_0^i)}^{\delta_+(x_0^i)} \mathcal{N}(x; \mu_\theta^i(\mathbf{x}_1, 1), \sigma_1^2)\, dx \\ \delta_+(x) &= \begin{cases} \infty & \text{if } x = 1 \\ x + \frac{1}{255} & \text{if } x < 1 \end{cases} \qquad \delta_-(x) = \begin{cases} -\infty & \text{if } x = -1 \\ x - \frac{1}{255} & \text{if } x > -1 \end{cases} \end{aligned} \tag{13}$$

where D is the data dimensionality and the i superscript indicates extraction of one coordinate. (It would be straightforward to instead incorporate a more powerful decoder like a conditional autoregressive model, but we leave that to future work.) Similar to the discretized continuous distributions used in VAE decoders and autoregressive models [34, 52], our choice here ensures that the variational bound is a lossless codelength of discrete data, without need of adding noise to the data or incorporating the Jacobian of the scaling operation into the log likelihood. At the end of sampling, we display $\boldsymbol{\mu}_\theta(\mathbf{x}_1, 1)$ noiselessly.

3.4 Simplified training objective

With the reverse process and decoder defined above, the variational bound, consisting of terms derived from Eqs. (12) and (13), is clearly differentiable with respect to θ and is ready to be employed for

Table 1: CIFAR10 results. NLL measured in bits/dim.

Model	IS	FID	NLL Test (Train)
Conditional			
EBM [11]	8.30	37.9	
JEM [17]	8.76	38.4	
BigGAN [3]	9.22	14.73	
StyleGAN2 + ADA (v1) [29]	**10.06**	**2.67**	
Unconditional			
Diffusion (original) [53]			≤ 5.40
Gated PixelCNN [59]	4.60	65.93	3.03 (2.90)
Sparse Transformer [7]			**2.80**
PixelIQN [43]	5.29	49.46	
EBM [11]	6.78	38.2	
NCSNv2 [56]		31.75	
NCSN [55]	8.87 ± 0.12	25.32	
SNGAN [39]	8.22 ± 0.05	21.7	
SNGAN-DDLS [4]	9.09 ± 0.10	15.42	
StyleGAN2 + ADA (v1) [29]	$\mathbf{9.74} \pm 0.05$	3.26	
Ours (L, fixed isotropic $\mathbf{\Sigma}$)	7.67 ± 0.13	13.51	≤ 3.70 (3.69)
Ours (L_{simple})	9.46 ± 0.11	**3.17**	≤ 3.75 (3.72)

Table 2: Unconditional CIFAR10 reverse process parameterization and training objective ablation. Blank entries were unstable to train and generated poor samples with out-of-range scores.

Objective	IS	FID
$\tilde{\boldsymbol{\mu}}$ prediction (baseline)		
L, learned diagonal $\mathbf{\Sigma}$	7.28 ± 0.10	23.69
L, fixed isotropic $\mathbf{\Sigma}$	8.06 ± 0.09	13.22
$\|\tilde{\boldsymbol{\mu}} - \tilde{\boldsymbol{\mu}}_\theta\|^2$	–	–
$\boldsymbol{\epsilon}$ prediction (ours)		
L, learned diagonal $\mathbf{\Sigma}$	–	–
L, fixed isotropic $\mathbf{\Sigma}$	7.67 ± 0.13	13.51
$\|\tilde{\boldsymbol{\epsilon}} - \boldsymbol{\epsilon}_\theta\|^2$ (L_{simple})	$\mathbf{9.46 \pm 0.11}$	**3.17**

training. However, we found it beneficial to sample quality (and simpler to implement) to train on the following variant of the variational bound:

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t, \mathbf{x}_0, \boldsymbol{\epsilon}} \left[\left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2 \right] \tag{14}$$

where t is uniform between 1 and T. The $t = 1$ case corresponds to L_0 with the integral in the discrete decoder definition (13) approximated by the Gaussian probability density function times the bin width, ignoring σ_1^2 and edge effects. The $t > 1$ cases correspond to an unweighted version of Eq. (12), analogous to the loss weighting used by the NCSN denoising score matching model [55]. (L_T does not appear because the forward process variances β_t are fixed.) Algorithm 1 displays the complete training procedure with this simplified objective.

Since our simplified objective (14) discards the weighting in Eq. (12), it is a weighted variational bound that emphasizes different aspects of reconstruction compared to the standard variational bound [18, 22]. In particular, our diffusion process setup in Section 4 causes the simplified objective to down-weight loss terms corresponding to small t. These terms train the network to denoise data with very small amounts of noise, so it is beneficial to down-weight them so that the network can focus on more difficult denoising tasks at larger t terms. We will see in our experiments that this reweighting leads to better sample quality.

4 Experiments

We set $T = 1000$ for all experiments so that the number of neural network evaluations needed during sampling matches previous work [53, 55]. We set the forward process variances to constants increasing linearly from $\beta_1 = 10^{-4}$ to $\beta_T = 0.02$. These constants were chosen to be small relative to data scaled to $[-1, 1]$, ensuring that reverse and forward processes have approximately the same functional form while keeping the signal-to-noise ratio at $\mathbf{x}_T$ as small as possible ($L_T = D_{\text{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel \mathcal{N}(\mathbf{0}, \mathbf{I})) \approx 10^{-5}$ bits per dimension in our experiments).

To represent the reverse process, we use a U-Net backbone similar to an unmasked PixelCNN++ [52, 48] with group normalization throughout [66]. Parameters are shared across time, which is specified to the network using the Transformer sinusoidal position embedding [60]. We use self-attention at the 16×16 feature map resolution [63, 60]. Details are in Appendix B.

4.1 Sample quality

Table 1 shows Inception scores, FID scores, and negative log likelihoods (lossless codelengths) on CIFAR10. With our FID score of 3.17, our unconditional model achieves better sample quality than most models in the literature, including class conditional models. Our FID score is computed with respect to the training set, as is standard practice; when we compute it with respect to the test set, the score is 5.24, which is still better than many of the training set FID scores in the literature.

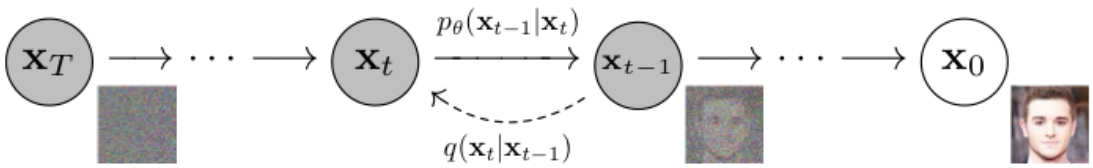


Figure 2: The directed graphical model considered in this work.

* 关键词：去噪

* 核心假设：

1. 数据服从概率分布 $\vec{q}$（未知）
2. 噪声服从高斯/正态分布

在推理阶段：输入纯噪声 $\vec{x}_T \sim N(0, I)$，

模型在 T 个时刻均预测一次噪声的标准差，

$\vec{x}_0$ 即为去噪生成的样本

① 训练阶段 1. 加噪

given: $\vec{x}_0$ (图片)

β_t: 超参

[4] $q(\vec{x}_t \mid \vec{x}_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}\,\vec{x}_{t-1}, \beta_t I)$ [4] 单步加噪

[3]
$$q(\vec{x}_{1:T} \mid \vec{x}_0) = q(\vec{x}_1, \vec{x}_2, \cdots, \vec{x}_T \mid \vec{x}_0)$$ 条件概率

$$= q(\vec{x}_1 \mid \vec{x}_0) \cdot q(\vec{x}_2, \vec{x}_3 \cdots, \vec{x}_T \mid \vec{x}_0, \vec{x}_1)$$

$$= q(\vec{x}_1 \mid \vec{x}_0) \cdot q(\vec{x}_{2:T} \mid \vec{x}_1)$$ 马尔可夫性质

$$= \cdots$$ (归纳)

$$= \prod_{t=1}^{T} q(x_t \mid x_{t-1})$$ [3].

定理1. 重参化 (reparameterization)

if $Z \in N(\mu, \sigma^2)$ $Z = \mu + \sigma\varepsilon$. where $\varepsilon \in N(0, I)$

核心: 采样标准正态分布即可达到采样任意正态分布的效果

定理2. 两正态分布之和

Let $X \in N(\mu_x, \sigma_x^2)$

$Y \in N(\mu_y, \sigma_y^2)$

Then $X + Y \in N(\mu_x + \mu_y, \sigma_x^2 + \sigma_y^2)$

6 * 用重参化的方式表达扩散 Forward Process：

$$q(\vec{x}_t \mid \vec{x}_{t-1}) = N(\sqrt{1-\beta_t}\,\vec{x}_{t-1},\ \beta_t I)$$

⇓ 重参化

$$\vec{x}_t = \sqrt{1-\beta_t}\,\vec{x}_{t-1} + \sqrt{\beta_t}\,\varepsilon_t, \quad \text{where } \varepsilon_t \in N(0, I)$$

$$= \sqrt{1-\beta_t}\cdot(\sqrt{1-\beta_{t-1}}\,\vec{x}_{t-2} + \sqrt{\beta_{t-1}}\,\varepsilon_{t-1}) + \sqrt{\beta_t}\,\varepsilon_t$$

$$= \sqrt{1-\beta_t}\cdot\sqrt{1-\beta_{t-1}}\,\vec{x}_{t-2} + \left(\sqrt{1-\beta_t}\sqrt{\beta_{t-1}}\,\varepsilon_{t-1} + \sqrt{\beta_t}\,\varepsilon_t\right)$$

→ 两正态分布之和！

$\sqrt{1-\beta_t}\sqrt{\beta_{t-1}}\,\varepsilon_{t-1} + \sqrt{\beta_t}\,\varepsilon_t \sim N(0,\ 1-(1-\beta_t)(1-\beta_{t-1}))$

* 简写：$\alpha_t = 1-\beta_t$ $=$ $\sqrt{1-\alpha_t\alpha_{t-1}}\,\bar{\varepsilon}_{t-1}$（重参）

$$= \sqrt{\alpha_t\alpha_{t-1}}\,\vec{x}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\,\bar{\varepsilon}_{t-1}$$

$$= \cdots \quad \text{（归纳）}$$

$$= \sqrt{\prod_{i=1}^{t}\alpha_i}\,\vec{x}_0 + \sqrt{1-\prod_{i=1}^{t}\alpha_i}\,\bar{\varepsilon}_0$$

* 简写：$\bar{\alpha}_t = \prod_{i=1}^{t}\alpha_i$

$$(\vec{x}_t) \quad = \sqrt{\bar{\alpha}_t}\,\vec{x}_0 + \sqrt{1-\bar{\alpha}_t}\,\vec{\varepsilon}.$$

Hence $q(\vec{x}_t \mid \vec{x}_0) = N(\sqrt{\bar{\alpha}_t}\,\vec{x}_0,\ (1-\bar{\alpha}_t)I)$

结论：Given $\vec{x}_0$，到任意一步 $\vec{x}_t$ 都一步到位，
无需采样 t 次（加速训练）

假设：一共加噪 T 次

$$q(\vec{x}_T \mid \vec{x}_0) = N(\bar{\alpha}_T \vec{x}_0 + (1-\bar{\alpha}_T)I)$$

$$\bar{\alpha}_T = \prod_{i=1}^{T} \alpha_i = \prod_{i=1}^{T} (1-\beta_i)$$

β_i：超参，0 ~ 1 取值

$\bar{\alpha}_T \to 0$ as $T \to \infty$

Hence $q(\vec{x}_T \mid \vec{x}_0) \to N(0, I)$ as $T \to \infty$

这就是为什么推理所谓（无加噪过程），
降噪过程开局从 $N(0, I)$ 标准正态分布采样！

---

② 训练阶段 2. 降噪过程（Reverse Process）

输入：$\vec{x}_t$（在 T 时刻推理，则 $\vec{x}_T$ 为完全噪声）
时刻 t

输出：去噪分布（模型预测！）

2 $p_\theta(\vec{x}_{t-1} \mid \vec{x}_t) \sim N(\mu_\theta(\vec{x}_t, t), \Sigma_\theta^2(\vec{x}_t, t))$

③. Loss 设计

核心：
- a. 最大似然估计 (MLE: maximum likely hood estimation)
- b. 证据/变分下界 (ELBO: evidence lower bound)
- c. 化点简化

注：对 diffusion loss 推导感兴趣的同学，
非常推荐从 VAE (Variational Autoencoder 变分自编码器) 开始入门，VAE基本可以理解为一层 diffusion

a. 最大似然估计 (maximum likelyhood estimation)

背景：真实数据服从未知分布 q，(扩散中即 $p(\vec{x}_0)$)

我们有：q 的采样 (例：n张图片的数据集)

目标：最大化 概率模型 p_θ 下观测/采样数据的可能性

* Logistic Regression 逻辑回归用的就是 MLE

* 扩散模型：$\vec{x}_0 \sim q(\vec{x}_0)$ 假设 $\vec{d}_0 \cdots \vec{d}_{n-1}$ 是 n 个采样

MLE：

$$\arg\max_\theta \prod_{i=0}^{n-1} p_\theta(\vec{d}_i)$$

$$= \arg\max_\theta \log \prod_{i=0}^{n-1} p_\theta(\vec{d}_i)$$

$$= \arg\max_\theta \sum_{i=0}^{n-1} \log p_\theta(\vec{d}_i)$$

* 此处 $p_\theta(\vec{d}_i)$ 指的是 p_θ 模型采样 $\vec{d}_i$ 的可能性

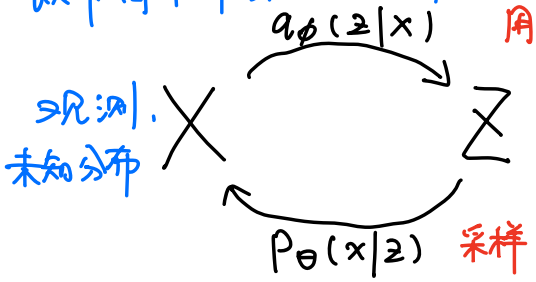
⑤式左

$$\Leftrightarrow \arg\min_\theta E_{\vec{x}_0 \sim q}[-\log p_\theta(\vec{x}_0)] \quad (\text{当 } n \text{ 够大，大数定律})$$

b. 证据/变分下界 (Evidence Lower Bound)

$$\ln p_\theta(x) \geq E_{z\sim q_\phi(\cdot|x)}\left[\ln \frac{p_\theta(x,z)}{q_\phi(z|x)}\right]$$

以下简单介绍 VAE 来辅助理解 ELBO



用 q_ϕ 估计 p_θ 后验概率

隐变量，先验 $z = N(0, I)$，目的：生成 X

采样 z，用 $p_\theta(x|z)$ 得到 z 条件下 x 的分布

$$p_\theta(x) = \int p_\theta(x|z)\, p(z)\, dz \quad \text{边缘积分}$$

$$= \int p_\theta(x,z) \cdot \frac{q_\phi(z|x)}{q_\phi(z|x)} \cdot dz$$

$$= E_{z\sim q_\phi(\cdot|x)}\left[\frac{p_\theta(x,z)}{q_\phi(z|x)}\right]$$

$$\ln p_\theta(x) = \ln E_{z\sim q_\phi(\cdot|x)}\left[\frac{p_\theta(x,z)}{q_\phi(z|x)}\right]$$

$$\geq E_{z\sim q_\phi(\cdot|x)}\left[\ln \frac{p_\theta(x,z)}{q_\phi(z|x)}\right] \circ$$

琴生不等式：$\ln E[x] \geq E[\ln x]$

核心：用 $q_\phi(z|x)$ 估计 $p_\theta(z|x)$ 后验概率

在扩散模型，变分下界推导（KL散度角度）

核心：max变分下界 = min KL散度，
估计分布与真实分布越接近

$$-\log p_\theta(\vec{x}_0)$$

$$\leq -\log p_\theta(\vec{x}_0) + D_{KL}\left(q(\vec{x}_{1:T}|\vec{x}_0) \,\|\, p_\theta(\vec{x}_{1:T}|\vec{x}_0)\right)$$

先验，真实　后验，估计

$$= -\log p_\theta(\vec{x}_0) + E_{\vec{x}_{1:T}\sim q(\vec{x}_{1:T}|\vec{x}_0)}\left[\log \frac{q(\vec{x}_{1:T}|\vec{x}_0)}{p_\theta(\vec{x}_{1:T}|\vec{x}_0)}\right]$$

$$= -\log p_\theta(\vec{x}_0) + E_q\left[\log \frac{q(\vec{x}_{1:T}|\vec{x}_0)}{p_\theta(\vec{x}_{0:T})} + \log p_\theta(\vec{x}_0)\right]$$

（$\log ab = \log a + \log b$，条件概率）

$$= E_{\vec{x}_{1:T}\sim q(\vec{x}_{1:T}|\vec{x}_0)}\left[\log \frac{q(\vec{x}_{1:T}|\vec{x}_0)}{p_\theta(\vec{x}_{0:T})}\right]$$

两边对 $\vec{x}_0$ 再取期望，形成负对数似然上界

$$E_{x_0\sim q}\left[-\log p_\theta(\vec{x}_0)\right] \leq E_{\vec{x}_{0:T}\sim q}\left[-\log \frac{p_\theta(\vec{x}_{0:T})}{q(\vec{x}_{1:T}|\vec{x}_0)}\right]$$

论文中只写3 q，但 q 代表3两件东西

1. $\vec{x}_0 \sim q(\vec{x}_0)$：未知数据真实分布
2. $\vec{x}_{1:T} \sim q(\vec{x}_{1:T}|\vec{x}_0)$：加噪已知正态分布

S 不等式

□ ELBO证完

$$E_q\left[-\log \frac{p_\theta(\vec{x}_{0:T})}{q(\vec{x}_{1:T}|\vec{x}_0)}\right]$$ 5展开

[1]

$$= E_q\left[-\log \frac{p_\theta(\vec{x}_T)\prod_{i=1}^{T} p_\theta(\vec{x}_{i-1}|\vec{x}_i)}{\prod_{i=1}^{T} q(\vec{x}_i|\vec{x}_{i-1})}\right]$$

[3]

$$= E_q\left[-\log p_\theta(\vec{x}_T) - \sum_{t=1}^{T} \log \frac{p_\theta(\vec{x}_{t-1}|\vec{x}_t)}{q(\vec{x}_t|\vec{x}_{t-1})}\right]$$ 5右

A Extended derivations

Below is a derivation of Eq. (5), the reduced variance variational bound for diffusion models. This material is from Sohl-Dickstein et al. [53]; we include it here only for completeness.

$$L = \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] \tag{17}$$

$$= \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t\geq 1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] \tag{18}$$ 已知

$$= \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t>1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})} - \log \frac{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}{q(\mathbf{x}_1|\mathbf{x}_0)}\right] \tag{19}$$

$$= \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t>1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}\cdot\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} - \log \frac{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}{q(\mathbf{x}_1|\mathbf{x}_0)}\right] \tag{20}$$ 贝叶斯

$$= \mathbb{E}_q\left[-\log \frac{p(\mathbf{x}_T)}{q(\mathbf{x}_T|\mathbf{x}_0)} - \sum_{t>1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)} - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)\right] \tag{21}$$ $\log ab = \log a + \log b$

Loss函数的KL散度解读角度：

q分布与 p_θ 预测的分布差

[7]

$$= \mathbb{E}_q\left[D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \,\|\, p(\mathbf{x}_T)) + \sum_{t>1} D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) \,\|\, p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)) - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)\right] \tag{22}$$

The following is an alternate version of L. It is not tractable to estimate, but it is useful for our discussion in Section 4.3

$$L = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t\geq 1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] \tag{23}$$

$$= \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t\geq 1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_{t-1}|\mathbf{x}_t)}\cdot\frac{q(\mathbf{x}_{t-1})}{q(\mathbf{x}_t)}\right] \tag{24}$$

$$= \mathbb{E}_q\left[-\log \frac{p(\mathbf{x}_T)}{q(\mathbf{x}_T)} - \sum_{t\geq 1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_{t-1}|\mathbf{x}_t)} - \log q(\mathbf{x}_0)\right] \tag{25}$$

$$= D_{\mathrm{KL}}(q(\mathbf{x}_T) \,\|\, p(\mathbf{x}_T)) + \mathbb{E}_q\left[\sum_{t\geq 1} D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t) \,\|\, p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))\right] + H(\mathbf{x}_0) \tag{26}$$

q 后验 : $q(\vec{x}_{t-1} | \vec{x}_t, \vec{x}_0)$

$$= \frac{q(\vec{x}_t | \vec{x}_{t-1}, \vec{x}_0) \cdot q(\vec{x}_{t-1} | \vec{x}_0)}{q(\vec{x}_t | \vec{x}_0)}$$ 贝叶斯

$$= (\text{计算略, 重参化})$$

$$= \mathcal{N}(\tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t I)$$

where $$\tilde{\mu}_t = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\vec{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\vec{x}_t$$

$$\tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$$ 8.

重点: q 后验分布的标准差仅依赖超参, 无需估计
p_θ 仅需预测均值 (期望)

推论 3. if $p \sim N(\mu_1, \sigma_1^2)$

$q \sim N(\mu_2, \sigma_2^2)$

$$D_{KL}(p \| q) = \log\frac{\sigma_2}{\sigma_1} + \frac{\sigma_1^2 + (\mu_1-\mu_2)^2}{2\sigma_2^2} - \frac{1}{2}$$

在扩散上下文中, $\sigma_1 = \sigma_2$, 所以 loss 就从

KL 散度简化为两个分布均值的均方差!

(MSE loss: mean squared error)

Loss =

$$\mathbb{E}_q\Big[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0)\parallel p(\mathbf{x}_T))}_{L_T}+\sum_{t>1}\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)\parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}}\underbrace{-\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\Big] \quad (5)$$

7

常数 (3.1)　　用均值/期望的MSE Loss! (3.2)

$$L_{t-1}=E_q\left[\frac{1}{2\sigma_t^2}\left[\tilde{\mu}_t(\vec{x}_t,\vec{x}_0)-\mu_\theta(\vec{x}_t,t)\right]^2\right]+C$$

$$=E_{\substack{\vec{x}_0\sim q\\ \vec{x}_t\sim q(\vec{x}_t|\vec{x}_0)}}\left[\frac{1}{2\sigma_t^2}\left[\tilde{\mu}_t(\vec{x}_t,\vec{x}_0)-\mu_\theta(\vec{x}_t,t)\right]^2\right]+C$$

[2]代入: ① $q(\vec{x}_t|\vec{x}_0)=N(\bar{\alpha}_t\vec{x}_0,(1-\bar{\alpha}_t)I)$ [4]

$\bar{\alpha}_t=\prod\alpha_i$

$\alpha_i=1-\beta_i$

② $\vec{x}_t=\sqrt{\bar{\alpha}_t}\,\vec{x}_0+\sqrt{1-\bar{\alpha}_t}\,\varepsilon$, where $\varepsilon\sim N(0,I)$

③ $\tilde{\mu}_t=\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\vec{x}_0+\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\vec{x}_t$ 重参化

②代入 $=\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\cdot\frac{\vec{x}_t-\sqrt{1-\bar{\alpha}_t}\,\varepsilon}{\sqrt{\bar{\alpha}_t}}+\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\vec{x}_t$ [8]

$$=\frac{1}{\sqrt{\alpha_t}}\cdot\left[\frac{\beta_t}{(1-\bar{\alpha}_t)}\vec{x}_t+\frac{\alpha_t-\bar{\alpha}_t}{1-\bar{\alpha}_t}\vec{x}_t-\frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\varepsilon\right]$$

$$=\frac{1}{\sqrt{\alpha_t}}\left[\vec{x}_t(\vec{x}_0,\varepsilon)-\frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\varepsilon\right]$$

$$L_{t-1} - C$$

$$= E_{x_0,\epsilon}\left[\frac{1}{2\sigma_t^2}\left\|\frac{1}{\sqrt{\alpha_t}}\left(\vec{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right) - \mu_\theta(\vec{x}_t, t)\right\|^2\right]$$

最后一个 trick：let $\mu_\theta(\vec{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(\vec{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta\right)$ a

核心：不预测分布均值，直接预测先验噪声，再求得均值，简化训练流程

$$\rightarrow = E_{x_0,\epsilon}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}\left\|\epsilon - \epsilon_\theta(\vec{x}_t, t)\right\|^2\right] \quad b$$

where $\vec{x}_t = \sqrt{\bar{\alpha}_t}\,\vec{x}_0 + \sqrt{1-\bar{\alpha}_t}\,\epsilon$

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on b
$$\nabla_\theta \left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2$$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t\mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

μ_θ ! a