Learning Energy-Based Models by Diffusion Recovery Likelihood

Ruiqi Gao
UCLA
ruiqigao@ucla.edu

Yang Song
Stanford
yangsong@cs.stanford.edu

Ben Poole
Google Brain
pooleb@google.com

Ying Nian Wu
UCLA
ywu@stat.ucla.edu

Diederik P. Kingma
Google Brain
durk@google.com

Abstract

While energy-based models (EBMs) exhibit a number of desirable properties, training and sampling on high-dimensional datasets remains challenging. Inspired by recent progress on diffusion probabilistic models, we present a diffusion recovery likelihood method to tractably learn and sample from a sequence of EBMs trained on increasingly noisy versions of a dataset. Each EBM is trained with recovery likelihood, which maximizes the conditional probability of the data at a certain noise level given their noisy versions at a higher noise level. Optimizing recovery likelihood is more tractable than marginal likelihood, as sampling from the conditional distributions is much easier than sampling from the marginal distributions. After training, synthesized images can be generated by the sampling process that initializes from Gaussian white noise distribution and progressively samples the conditional distributions at decreasingly lower noise levels. Our method generates high fidelity samples on various image datasets. On unconditional CIFAR-10 our method achieves FID 9.60 and inception score 8.58, superior to the majority of GANs. Moreover, we demonstrate that unlike previous work on EBMs, our long-run MCMC samples from the conditional distributions do not diverge and still represent realistic images, allowing us to accurately estimate the normalized density of data even for high-dimensional datasets.

1 Introduction

EBMs [27, 33, 21, 47, 10, 43, 6, 7, 25, 35, 5, 11, 4, 8, 2, 12] are an appealing class of probabilistic models, which can be viewed as generative versions of discriminators [18, 26, 28, 12], yet can be learned from unlabeled data. Despite a number of desirable properties, two challenges remain for training EBMs on high-dimensional datasets. First, learning EBMs by maximum likelihood requires Markov Chain Monte Carlo (MCMC) to generate samples from the model, which can be extremely expensive. Second, as pointed out in [34], the energy potentials learned with non-convergent MCMC do not have a valid steady-state, in the sense that samples from long-run Markov chains can differ greatly from observed samples, making it difficult to evaluate the learned energy potentials.

Another line of work, originating from [37], is to learn from a diffused version of the data, which are obtained from the original data via a diffusion process that sequentially adds Gaussian white noise. From such diffusion data, one can learn the conditional model of the data at a certain noise level given their noisy versions at the higher noise level of the diffusion process. After learning the sequence of conditional models that invert the diffusion process, one can then generate synthesized images from

Preprint. Under review.

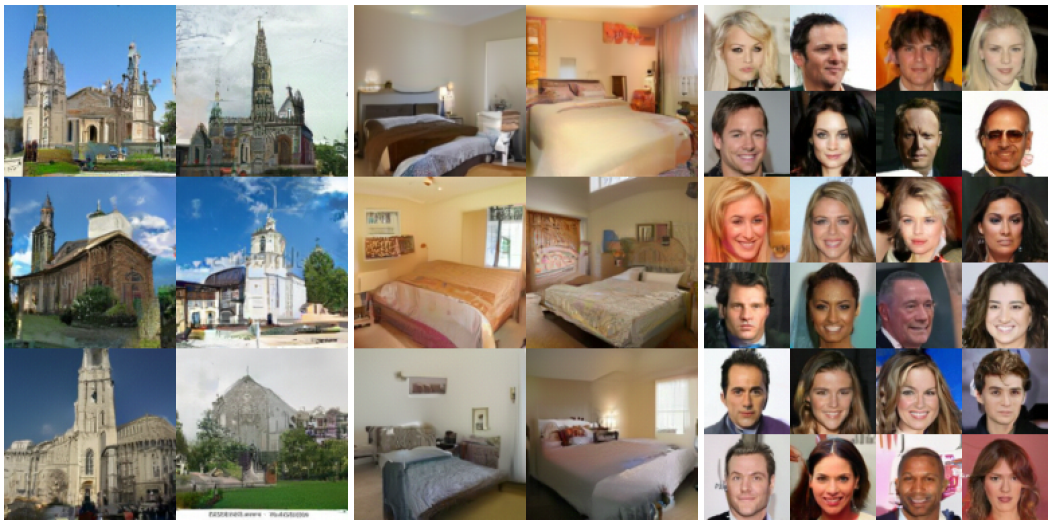

Figure 1: Generated samples on LSUN 128^2 church_outdoor *(left)*, LSUN 128^2 bedroom *(center)* and CelebA 64^2 *(right)*.

Gaussian white noise images by ancestral sampling. Building on [37], [17] further developed the method, obtaining strong image synthesis results.

Inspired by [37] and [17], we propose a *diffusion recovery likelihood* method to tackle the challenge of training EBMs directly on a dataset by instead learning a sequence of EBMs for the *marginal* distributions of the diffusion process. The sequence of marginal EBMs are learned with recovery likelihoods that are defined as the conditional distributions that invert the diffusion process. Compared to standard MLE learning of EBMs, learning marginal EBMs by diffusion recovery likelihood only requires sampling from the conditional distributions, which is much easier than sampling from the marginal distributions. After learning the marginal EBMs, we can generate synthesized images by a sequence of conditional samples initialized from the Gaussian white noise distribution. Unlike [17] that approximates the reverse process by normal distributions, in our case the conditional distributions are derived from the marginal EBMs, which are more flexible. The framework of recovery likelihood was originally proposed in [1]. In our work, we adapt it to learning the sequence of marginal EBMs from the diffusion data.

Our work is also related to the denoising score matching method of [41], which was further developed by [38, 39] for learning from diffusion data. These methods learn the score functions (the gradients of the energy functions) directly, instead of using the gradients of learned energy functions as in EBMs. The training objective used for diffusion probabilisitic models is a weighted version of the denoising score matching objective, as revealed by [17].

We demonstrate the efficacy of diffusion recovery likelihood on CIFAR-10, CelebA and LSUN datasets. The generated samples are of high fidelity and comparable to GAN-based methods. On CIFAR-10, we achieve FID 9.60 and inception score 8.58, exceeding existing methods of learning explicit EBMs to a large extent. We also demonstrate that diffusion recovery likelihood outperforms denoising score matching from diffusion data if we naively take the gradients of explicit energy functions as the score functions. More interestingly, by using a thousand diffusion time steps, we demonstrate that even very long MCMC chains from the sequence of conditional distributions produce samples that represent realistic images. With the faithful long-run MCMC samples from the conditional distributions, we can accurately estimate the marginal partition function at zero noise level by importance sampling, and thus evaluate the normalized density of data under the EBM.

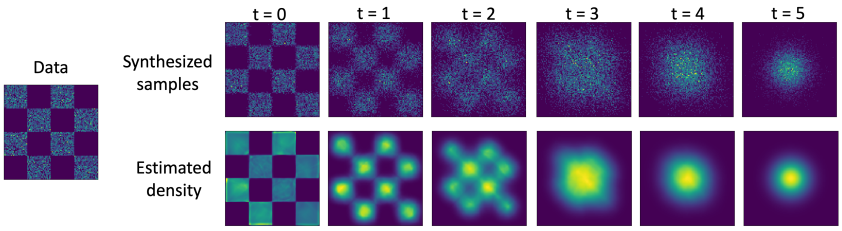

Figure 3: Illustration of diffusion recovery likelihood on 2D checkerboard example. *Top*: progressively generated samples. *Bottom*: estimated marginal densities.

2 Background

Let $\mathbf{x} \sim p_{\text{data}}(\mathbf{x})$ denote a training example, and $p_\theta(\mathbf{x})$ denote a model's probability density function that aims to approximates $p_{\text{data}}(\mathbf{x})$. An energy-based model (EBM) is defined as:

$$p_\theta(\mathbf{x}) = \frac{1}{Z_\theta} \exp(f_\theta(\mathbf{x})), \tag{1}$$

where $Z_\theta = \int \exp(f_\theta(\mathbf{x}))d\mathbf{x}$ is the partition function, which is analytically intractable for high-dimensional $\mathbf{x}$. For images, we parameterize $f_\theta(\mathrm{x})$ with a convolutional neural network with a scalar output.

The energy-based model in equation 1 can, in principle, be learned through MLE. Specifically, suppose we observe samples $\mathbf{x}_i \sim p_{\text{data}}(\mathbf{x})$ for $i = 1, 2, ..., n$. The log-likelihood function is

$$\mathcal{L}(\theta) = \frac{1}{n} \sum_{i=1}^{n} \log p_\theta(\mathbf{x}_i) \doteq \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}}[\log p_\theta(\mathbf{x})]. \tag{2}$$

In MLE, we seek to maximize the log-likelihood function, where the gradient approximately follows

$$-\frac{\partial}{\partial \theta} D_{\text{KL}}(p_{\text{data}} \| p_\theta) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}} \left[\frac{\partial}{\partial \theta} f_\theta(\mathbf{x}) \right] - \mathbb{E}_{\mathbf{x} \sim p_\theta} \left[\frac{\partial}{\partial \theta} f_\theta(\mathbf{x}) \right]. \tag{3}$$

The expectations can be approximated by averaging over the observed samples and the synthesized samples drawn from the model distribution $p_\theta(\mathbf{x})$ respectively. Generating synthesized samples from $p_\theta(\mathbf{x})$ can be done with Markov Chain Monte Carlo (MCMC) such as Langevin dynamics (or Hamiltonian Monte Carlo [9]), which iterates

$$\mathbf{x}^{\tau+1} = \mathbf{x}^\tau + \frac{\delta^2}{2} \nabla_{\mathbf{x}} f_\theta(\mathbf{x}^\tau) + \delta \boldsymbol{\epsilon}^\tau, \tag{4}$$

where τ indexes the time, δ is the step size, and $\boldsymbol{\epsilon}^\tau \sim \mathcal{N}(0, \boldsymbol{I})$. The difficulty lies in the fact that for high-dimensional and multi-modal distributions, MCMC sampling can take a long time to converge, and the sampling chains may have difficulty traversing modes. As demonstrated in Figure 2, training EBMs with synthesized samples from non-convergent MCMC results in malformed energy landscapes [35], even if the samples from the model look reasonable.

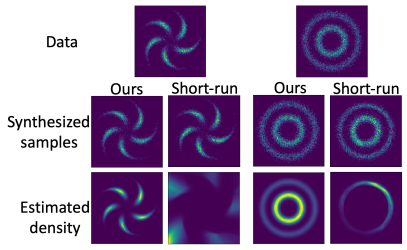

Figure 2: Comparison of learning EBMs by diffusion recovery likelihood (Ours) versus marginal likelihood (Short-run).

3 Recovery Likelihood

3.1 From Marginal to Conditional

Given the difficulty of sampling from the marginal density $p_\theta(\mathbf{x})$, following [1], we use the recovery likelihood defined by the density of the observed sample conditional on a noisy sample perturbed by isotropic Gaussian noise. Specifically, let $\tilde{\mathbf{x}} = \mathbf{x} + \sigma \boldsymbol{\epsilon}$ be the noisy observation of $\mathbf{x}$, where

$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$. Suppose $p_\theta(\mathbf{x})$ is defined by the EBM in equation 1, then the conditional EBM can be derived as

$$p_\theta(\mathbf{x}|\tilde{\mathbf{x}}) = \frac{1}{\tilde{Z}_\theta(\tilde{\mathbf{x}})} \exp\left(f_\theta(\mathbf{x}) - \frac{1}{2\sigma^2}\|\tilde{\mathbf{x}} - \mathbf{x}\|^2\right), \tag{5}$$

where $\tilde{Z}_\theta(\tilde{\mathbf{x}}) = \int \exp\left(f_\theta(\mathbf{x}) - \frac{1}{2\sigma^2}\|\tilde{\mathbf{x}} - \mathbf{x}\|^2\right) d\mathbf{x}$ is the partition function of this conditional EBM. See Appendix A.1 for the derivation. Compared to $p_\theta(\mathbf{x})$ (equation 1), the extra quadratic term $\frac{1}{2\sigma^2}\|\tilde{\mathbf{x}} - \mathbf{x}\|^2$ in $p_\theta(\mathbf{x}|\tilde{\mathbf{x}})$ constrains the energy landscape to be localized around $\tilde{\mathbf{x}}$, making the latter less multi-modal and easier to sample from. As we will show later, when σ is small, $p_\theta(\mathbf{x}|\tilde{\mathbf{x}})$ is approximately a single mode Gaussian distribution, which greatly reduces the burden of MCMC.

A more general formulation is $\tilde{\mathbf{x}} = a\mathbf{x} + \sigma\boldsymbol{\epsilon}$, where a is a positive constant. In that case, we can let $\mathbf{y} = a\mathbf{x}$, and treat $\mathbf{y}$ as the observed sample. Assume $p_\theta(\mathbf{y}) = \frac{1}{Z_\theta}\exp(f_\theta(\mathbf{y}))$, then by *change of variable*, the density function of $\mathbf{x}$ can be derived as $g_\theta(\mathbf{x}) = ap_\theta(a\mathbf{x})$.

3.2 Maximizing recovery likelihood

With the conditional EBM, assume we have observed samples $\mathbf{x}_i \sim p_{\text{data}}(\mathbf{x})$ and the corresponding perturbed samples $\tilde{\mathbf{x}}_i = \mathbf{x}_i + \sigma\boldsymbol{\epsilon}_i$ for $i = 1, ..., n$. We define the *recovery log-likelihood function* as

$$\mathcal{J}(\theta) = \frac{1}{n}\sum_{i=1}^{n} \log p_\theta(\mathbf{x}_i|\tilde{\mathbf{x}}_i). \tag{6}$$

The term *recovery* indicates that we attempt to recover the clean sample $\mathbf{x}_i$ from the noisy sample $\tilde{\mathbf{x}}_i$. Thus, instead of maximizing $\mathcal{L}(\theta)$ in equation 2, we can maximize $\mathcal{J}(\theta)$, whose distributions are easier to sample from. Specifically, we generate synthesized samples by K steps of Langevin dynamics that iterates

$$\mathbf{x}^{\tau+1} = \mathbf{x}^\tau + \frac{\delta^2}{2}(\nabla_{\mathbf{x}} f_\theta(\mathbf{x}^\tau) + \frac{1}{\sigma^2}(\tilde{\mathbf{x}} - \mathbf{x}^\tau)) + \delta\boldsymbol{\epsilon}^\tau. \tag{7}$$

The model is then updated following the same learning gradients as MLE (equation 3), because the quadratic term $-\frac{1}{2\sigma^2}\|\tilde{\mathbf{x}} - \mathbf{x}\|^2$ is not related to θ. Following the classical analysis of MLE, we can show that the point estimate given by maximizing recovery likelihood is an unbiased estimator of the true parameters, which means that given enough data, a rich enough model and exact synthesis, maximizing the recovery likelihood learns θ such that $p_{\text{data}}(\mathbf{x}) = p_\theta(\mathbf{x})$. See Appendix A.2 for a theoretical explanation.

3.3 Normal Approximation to Conditional

When the variance of perturbed noise σ^2 is small, $p_\theta(\mathbf{x}|\tilde{\mathbf{x}})$ can be approximated by a normal distribution via a first order Taylor expansion at $\tilde{\mathbf{x}}$. Specifically, the negative conditional energy is

$$-\mathcal{E}_\theta(\mathbf{x}|\tilde{\mathbf{x}}) = f_\theta(\mathbf{x}) - \frac{1}{2\sigma^2}\|\tilde{\mathbf{x}} - \mathbf{x}\|^2 \tag{8}$$

$$\doteq f_\theta(\tilde{\mathbf{x}}) + \langle \nabla_{\mathbf{x}} f_\theta(\tilde{\mathbf{x}}), \mathbf{x} - \tilde{\mathbf{x}}\rangle - \frac{1}{2\sigma^2}\|\tilde{\mathbf{x}} - \mathbf{x}\|^2 \tag{9}$$

$$= -\frac{1}{2\sigma^2}\left[\|\mathbf{x} - (\tilde{\mathbf{x}} + \sigma^2\nabla_{\mathbf{x}} f_\theta(\tilde{\mathbf{x}}))\|^2\right] + c, \tag{10}$$

where c include terms irrelevant of $\mathbf{x}$. In the above approximation, we do not perform second order Taylor expansion because σ^2 is small, and $\|\tilde{\mathbf{x}} - \mathbf{x}\|^2/2\sigma^2$ will dominate all the second order terms from Taylor expansion. Thus we can approximate $p_\theta(\mathbf{x}|\tilde{\mathbf{x}})$ by a Gaussian approximation $\widetilde{p}_\theta(\mathbf{x}|\tilde{\mathbf{x}})$:

$$\widetilde{p}_\theta(\mathbf{x}|\tilde{\mathbf{x}}) = \mathcal{N}\left(\mathbf{x}; \tilde{\mathbf{x}} + \sigma^2\nabla_{\mathbf{x}} f_\theta(\tilde{\mathbf{x}}), \sigma^2\right). \tag{11}$$

We can sample from this distribution using:

$$\mathbf{x}_{\text{gen}} = \tilde{\mathbf{x}} + \sigma^2\nabla_{\mathbf{x}} f_\theta(\tilde{\mathbf{x}}) + \sigma\boldsymbol{\epsilon}, \tag{12}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$. This resembles a single step of Langevin dynamics, except that $\sigma\boldsymbol{\epsilon}$ is replaced by $\sqrt{2}\sigma\boldsymbol{\epsilon}$ in Langevin dynamics. This normal approximation has two traits: (1) it verifies the fact that the conditional density $p_\theta(\mathbf{x}|\tilde{\mathbf{x}})$ can be generally easier to sample from when σ is small; (2) it provides hints of choosing the step size of Langevin dynamics, as discussed in section 3.5.

3.4 Connection to variational inference and score matching

The normal approximation to the conditional distribution leads to a natural connection to diffusion probabilistic models [37, 17] and denoising score matching [41, 38, 39]. Specifically, in diffusion probabilistic models, instead of modeling $p_\theta(x)$ as an energy-based model, it recruits variational inference and directly models the conditional density as

$$p_\theta(\mathbf{x}|\tilde{\mathbf{x}}) \sim \mathrm{N}\left(\tilde{\mathbf{x}} + \sigma^2 s_\theta(\tilde{\mathbf{x}}), \sigma^2\right), \tag{13}$$

which is in agreement with the normal approximation (equation 11), with $s_\theta(\mathbf{x}) = \nabla_{\mathbf{x}} f_\theta(\mathbf{x})$. On the other hand, the training objective of denoising score matching is to minimize

$$\frac{1}{2\sigma^2}\mathbb{E}_{p(\mathbf{x},\tilde{\mathbf{x}})}[\|\mathbf{x} - (\tilde{\mathbf{x}} + \sigma^2 s_\theta(\tilde{\mathbf{x}}))\|^2], \tag{14}$$

where $s_\theta(\mathbf{x})$ is the score of the density of $\tilde{\mathbf{x}}$. This objective is in agreement with the objective of maximizing log-likelihood of the normal approximation (equation 10), except that for normal approximation, $\nabla_{\mathbf{x}} f_\theta(\cdot)$ is the score of density of $\mathbf{x}$, instead of $\tilde{\mathbf{x}}$. However, the difference between the scores of density of $\mathbf{x}$ and $\tilde{\mathbf{x}}$ is of $O(\sigma^2)$, which is negligible when σ is sufficiently small (see Appendix A.3 for details).

As the normal approximation is accurate only when σ is small, it requires many time steps in the diffusion process for this approximation to work well, which is also reported in [17] and [39]. In contrast, the diffusion recovery likelihood framework can be more flexible in choosing the number of time steps and the magnitude of σ.

3.5 Diffusion recovery likelihood

As we discuss, sampling from $p_\theta(\mathbf{x}|\tilde{\mathbf{x}})$ becomes simple only when σ is small. In the extreme case when $\sigma \to \infty$, $p_\theta(\mathbf{x}|\tilde{\mathbf{x}})$ converges to the marginal distribution $p_\theta(\mathbf{x})$, which is again highly multi-modal and difficult to sample from. To keep σ small and meanwhile equip the model with the ability to generate new samples initialized from white noise, inspired by [37] and [17], we propose to learn a sequence of recovery likelihoods, on gradually perturbed observed data based on a diffusion process. Specifically, assume a sequence of perturbed observations $\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_T$ such that

$$\mathbf{x}_0 \sim p_{\mathrm{data}}(\mathbf{x}); \;\; \mathbf{x}_{t+1} = \sqrt{1-\sigma_{t+1}^2}\mathbf{x}_t + \sigma_{t+1}\boldsymbol{\epsilon}_{t+1}, \;\; t = 0, 1, ...T-1. \tag{15}$$

The scaling factor $\sqrt{1-\sigma_{t+1}^2}$ ensures that the sequence is a spherical interpolation between the observed sample and Gaussian white noise. Let $\mathbf{y}_t = \sqrt{1-\sigma_{t+1}^2}\mathbf{x}_t$, and we assume a sequence of conditional EBMs

$$p_\theta(\mathbf{y}_t|\mathbf{x}_{t+1}) = \frac{1}{\tilde{Z}_{\theta,t}(\mathbf{x}_{t+1})}\exp\left(f_\theta(\mathbf{y}_t, t) - \frac{1}{2\sigma_{t+1}^2}\|\mathbf{x}_{t+1} - \mathbf{y}_t\|^2\right), \;\; t = 0, 1, ..., T-1. \tag{16}$$

where $f_\theta(\mathbf{y}_t, t)$ is defined by a neural network conditioned on t.

We follow the learning algorithm in section 3.2. A question is how to determine the step size schedule δ_t of Langevin dynamics. Inspired by the sampling procedure of the normal approximation (equation 12), we set the step size $\delta_t = b\sigma_t$, where $b < 1$ is a tuned hyperparameter. This schedule turns out to work well in practice. Thus the K steps of Langevin dynamics iterates

$$\mathbf{y}_t^{\tau+1} = \mathbf{y}_t^\tau + \frac{b^2\sigma_t^2}{2}(\nabla_{\mathbf{y}} f_\theta(\mathbf{y}_t^\tau, t) + \frac{1}{\sigma_t^2}(\mathbf{x}_{t+1} - \mathbf{y}_t^\tau)) + b\sigma_t\boldsymbol{\epsilon}^\tau. \tag{17}$$

Algorithm 1 summarizes the training procedure. After training, we initialize the MCMC sampling from Gaussian white noise, and the synthesized sample at each time step serves to initialize the MCMC that samples from the model of the previous time step. See algorithm 2. To show the efficacy of our method, Figures 3 and 2 display several 2D toy examples learned by diffusion recovery likelihood.

Algorithm 1 Training

repeat
  Sample $t \sim \text{Unif}(\{0, ..., T-1\})$.
  Sample pairs $(\mathbf{y}_t, \mathbf{x}_{t+1})$.
  Set synthesized sample $\mathbf{y}_t^- = \mathbf{x}_{t+1}$.
  for $\tau \leftarrow 1$ to K **do**
    Update $\mathbf{y}_t^-$ according to equation 17.
  end for
  Update θ following the gradients
  $\frac{\partial}{\partial \theta} f_\theta(\mathbf{y}_t, t) - \frac{\partial}{\partial \theta} f_\theta(\mathbf{y}_t^-, t)$.
until converged.

Algorithm 2 Progressive sampling

Sample $\mathbf{x}_T \sim \mathcal{N}(0, \boldsymbol{I})$.
for $t \leftarrow T-1$ to T **do**
  $\mathbf{y}_t = \mathbf{x}_{t+1}$.
  for $\tau \leftarrow 1$ to K **do**
    Update $\mathbf{y}_t$ according to equation 17.
  end for
  $\mathbf{x}_t = \mathbf{y}_t / \sqrt{1 - \sigma_{t+1}^2}$.
end for
return $\mathbf{x}_0$.

4 Experiments

To show that diffusion recovery likelihood is flexible for diffusion process of various magnitudes of noise, we test the method under two settings: (1) $T = 6$, with $K = 30$ steps of Langevin dynamic per time step, $b = 0.0002$; (2) $T = 1000$, with sampling from normal approximation. (2) resembles the noise schedule of [17] and the magnitude of noise added at each time step is much smaller compared to (1). For both settings, we set σ_t^2 to increase linearly. The network structure of $f_\theta(x, t)$ is based on Wide ResNet [46] and we remove weight normalization. t is encoded by Transformer sinusoidal position embedding as in [17]. Architecture and training details are in Appendix B. Henceforth we simply refer the two settings as *T6* and *T1k*.

4.1 Image generation

Figures 1 and 4 display uncurated samples generated from learned models on CIFAR-10, CelebA 64×64, LSUN 64×64 and 128×128 datasets under *T6* setting. The samples are of high fidelity and comparable to GAN-based methods. Appendix C.2 provides more generated samples. Table 1 summarizes the quantitative evaluation on CIFAR-10 in terms of Frechet Inception Distance (FID) [15] and inception scores [36]. Our model achieves FID 9.60 and inception score 8.58, which outperforms existing methods of learning explicit energy-based models to a large extent, and is superior to a majority of GAN-based methods. Note that the score-based methods [38, 39, 17] directly parametrize and learn the score of data distribution, whereas our goal is to learn explicit energy-based models.

Figure 4: Generated samples on unconditional CIFAR-10 *(left)* and LSUN 64^2 church_outdoor *(center)* and LSUN 64^2 bedroom *(right)*.

Ablation study. See Table 2. We investigate the effect of choosing different number of time steps T and number of sampling steps K. First, to show that it is beneficial to learn by diffusion recovery likelihood, we compare against a baseline approach ($T = 1, K = 180$) where we use only one time step, so that the recovery likelihood becomes marginal likelihood. The approach is adopted by [35] and [5]. For fair comparison, we equip the baseline method the same budget of MCMC sampling

as our *T6* setting (i.e., 180 sampling steps). Next, we report the sample quality of setting *T1k*. We test two training objectives for this setting: (1) maximizing recovery likelihoods (T = 1000, K = 0) and (2) maximizing the approximated normal distributions (T=1000, K=0 (DSM)). As mentioned in section 3.4, (2) is equivalent to the training objectives of denoising score matching [38, 39] and diffusion probabilistic model [17], except that the score functions are taken as the gradients of explicit energy functions. Compared to (1), (2) results in a better FID score yet a worse inception score. Both (1) and (2) performs worse than setting T6. A possible explanation is that the sampling error may accumulate with many time steps. Last, we examine the influence of varying the number of sampling steps while fixing the number of time steps. The training becomes unstable when the number of sampling steps are not enough ($T = 6, K = 10$), and more sampling steps lead to better sample quality. However, since $K = 50$ does not gain significant improvement versus $K = 30$, yet of much higher computational cost, we keep $K = 30$ for image generation on all datasets.

Table 1: FID and inception scores on CIFAR-10.

Model	FID↓	Inception↑
GAN-based		
WGAN-GP [14]	36.4	7.86 ± .07
SNGAN [30]	21.7	8.22 ± .05
SNGAN-DDLS [2]	15.42	9.09 ± .10
StyleGAN2-ADA [20]	3.26	**9.74** ± .05
Score-based		
NCSN [38]	25.32	8.87 ± .12
NCSN-v2 [39]	31.75	-
DDPM [17]	**3.17**	9.46 ± .11
Explicit EBM-conditional		
CoopNets [44]	-	7.30
EBM-IG [5]	37.9	8.30
JEM [11]	38.4	8.76
Explicit EBM		
CoopNets [42]	33.61	6.55
EBM-SR [35]	-	6.21
EBM-IG [5]	38.2	6.78
Ours (*T6*)	9.60	8.58 ± .12

Table 2: Ablation of training objectives, time steps T and sampling steps K on CIFAR-10. $K = 0$ indicates that we sample from the normal approximation.

Setting / Objective	FID↓	Inception↑
T = 1, K = 180	32.12	6.89 ± 0.08
T = 1000, K = 0	25.12	7.85 ± 0.08
T = 1000, K = 0 (DSM)	21.88	7.75 ± 0.06
T = 6, K = 10	-	-
T = 6, K = 30	9.60	8.58 ± 0.12
T = 6, K = 50	**9.36**	**8.68** ± 0.11

Table 3: Test bits per dimension on CIFAR-10. We use AIS to estimate the partition function. See section 4.2.

Model	BPD
DDPM [17]	3.70
Glow [23]	3.35
Flow++ [16]	3.08
GPixelCNN [40]	3.03
Sparse Transformer [3]	2.80
DistAug [19]	**2.56**
Ours† (*T1k*)	3.18

Interpolation. As shown in Figure 5, our model is capable of smooth interpolation between two generated samples. Specifically, for two samples $\mathbf{x}_0^{(0)}$ and $\mathbf{x}_0^{(1)}$, we do a sphere interpolation between the initial white noise images $\mathbf{x}_T^{(0)}$ and $\mathbf{x}_T^{(1)}$ and the noise terms of Langevin dynamics $\boldsymbol{\epsilon}_{t,\tau}^{(0)}$ and $\boldsymbol{\epsilon}_{t,\tau}^{(1)}$ for every sampling step at every time step. More interpolation results can be found in Appendix C.1.

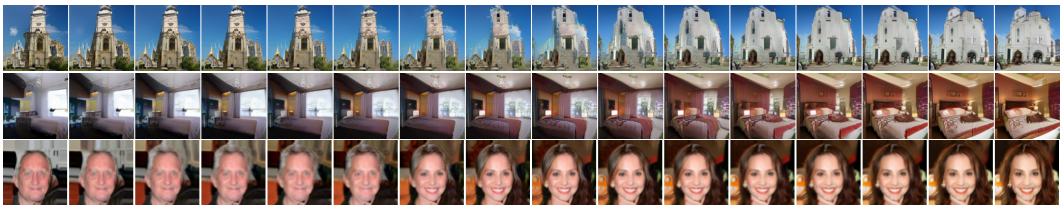

Figure 5: Interpolation results between the leftmost and rightmost generated samples. For *top* to *bottom*: LSUN church_outdoor 128^2, LSUN bedroom 128^2 and CelebA 64^2.

4.2 Long-run chain analysis

Besides achieving high quality generation, a perhaps equally important aspect of learning EBMs is to obtain a faithful energy potential. A principle way to check the validity of the learned potential is to perform long-run sampling chains and see if the samples still remain realistic. However, as pointed out in [34], almost all existing methods of learning EBMs fail in getting realistic long-run chain samples. In this subsection, we demonstrate that by composing a thousand diffusion time steps (*T1k* setting), we can use MCMC to form steady long-run chains for the conditional distributions.

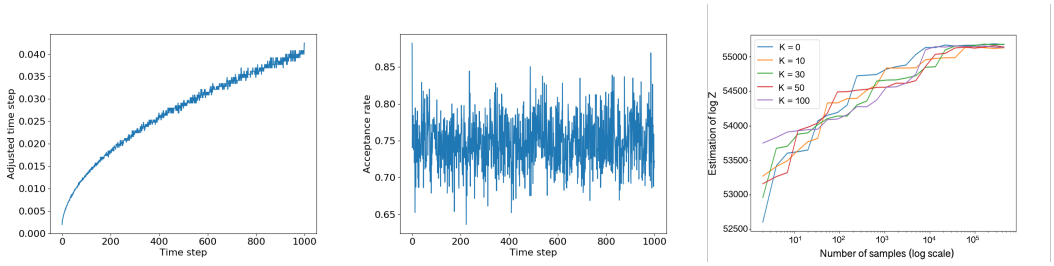

Figure 6: *Left*: Adjusted step size of HMC over time step. *Center*: Acceptance rate over time step. *Right*: Estimated log partition function over number of samples with different number of sampling steps per time step. The x axis is plotted in log scale.

First we prepare a faithful sampler for conducting long-run sampling. Specifically, after training the model under $T1k$ setting by maximizing diffusion recovery likelihood, for each time step, we first sample from the normal approximation and count it as one sampling step, and then use Hamiltonian Monte Carlo (HMC) [32] with 2 leapfrog steps to perform the consecutive sampling steps. To obtain a reasonable schedule of sampling step size, for each time step we adaptively adjust the step size of HMC to make the average acceptance rate range in $[0.6, 0.9]$, which is computed over 1000 chains for 100 steps. Figure 6 displays the adjusted step size (*left*) and acceptance rate (*center*) over time step. The adjusted step size increases logarithmically. With this step size schedule, we generate long-run chains from the learned sequence of conditional distributions. As shown in Figure 7, images remain realistic for even $100k$ sampling steps in total (i.e., 100 sampling steps per time step), resulting in FID 24.89. This score is close to the one computed on samples generated by $1k$ steps (i.e., sampled from normal approximation), which is 25.12.

Figure 7: Long-run chain samples from model-*T1k* with different total amount of HMC steps. From *left* to *right*: $1k$ steps, $10k$ steps and $100k$ steps.

More interestingly, given the faithful long-run MCMC samples from the *conditional* distributions, we can estimate the log ratio of the partition functions of the *marginal* distributions, and further estimate the partition function of $p_\theta(\mathbf{y}_0)$. The strategy is based on annealed importance sampling [31]. See Appendix A.4 for the implementation details. The right subfigure of Figure 6 depicts the estimated log partition function of $p_\theta(\mathbf{y}_0)$ over the number of MCMC samples used. To verify the estimation strategy and again check the long-run chain samples, we conduct multiple runs using samples generated with different numbers of HMC steps and display the estimation curves. All the curves saturate to values close to each other at the end, indicating the stability of long-run chain samples and the effectiveness of the estimation strategy. With the estimated partition function, by *change of variable*, we can estimate the normalized density of data as $g_\theta(\mathbf{x}_0) = \sqrt{1-\sigma_1^2} p_\theta(\sqrt{1-\sigma_1^2}\mathbf{x}_0)$. We report test bits per dimension on CIFAR-10 in Table 3. Note that the result should be taken with a grain of salt, because the partition function is estimated by samples and as shown in Appendix A.4, it is a stochastic lower bound of the true value, that will converge to the true value when the number of samples grows large.

5 Conclusion

We propose to learn EBMs by diffusion recovery likelihood, a variant of MLE applied to diffusion processes. We achieve high quality image synthesis, and with a thousand noise levels, we obtain faithful long-run MCMC samples that indicate the validity of the learned energy potentials. In future work, we look forward to investigating settings to combine the high quality synthesis of model-*T6* and faithful long-run MCMC samples of model-*T1k*, to achieve the best of two worlds. Since this method can learn EBMs efficiently with small budget of MCMC, we are also interested in scaling it up to higher resolution images and investigating this method in other data modalities.

References

[1] Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized denoising auto-encoders as generative models. In *Advances in neural information processing systems*, pages 899–907, 2013.

[2] Tong Che, Ruixiang Zhang, Jascha Sohl-Dickstein, Hugo Larochelle, Liam Paull, Yuan Cao, and Yoshua Bengio. Your gan is secretly an energy-based model and you should use discriminator driven latent sampling. *arXiv preprint arXiv:2003.06060*, 2020.

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

[4] Guillaume Desjardins, Yoshua Bengio, and Aaron C Courville. On tracking the partition function. In *Advances in neural information processing systems*, pages 2501–2509, 2011.

[5] Yilun Du and Igor Mordatch. Implicit generation and generalization in energy-based models. *arXiv preprint arXiv:1903.08689*, 2019.

[6] Chelsea Finn, Paul Christiano, Pieter Abbeel, and Sergey Levine. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. *arXiv preprint arXiv:1611.03852*, 2016.

[7] Ruiqi Gao, Yang Lu, Junpei Zhou, Song-Chun Zhu, and Ying Nian Wu. Learning generative convnets via multi-grid modeling and sampling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9155–9164, 2018.

[8] Ruiqi Gao, Erik Nijkamp, Diederik P Kingma, Zhen Xu, Andrew M Dai, and Ying Nian Wu. Flow contrastive estimation of energy-based models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7518–7528, 2020.

[9] Mark Girolami and Ben Calderhead. Riemann manifold langevin and hamiltonian monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(2):123–214, 2011.

[10] Anirudh Goyal Alias Parth Goyal, Nan Rosemary Ke, Surya Ganguli, and Yoshua Bengio. Variational walkback: Learning a transition operator as a stochastic recurrent net. In *Advances in Neural Information Processing Systems*, pages 4392–4402, 2017.

[11] Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. *arXiv preprint arXiv:1912.03263*, 2019.

[12] Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, and Richard Zemel. Learning the stein discrepancy for training and evaluating energy-based models without sampling.

[13] Roger B Grosse, Siddharth Ancha, and Daniel M Roy. Measuring the reliability of mcmc inference with bidirectional monte carlo. In *Advances in Neural Information Processing Systems*, pages 2451–2459, 2016.

[14] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777, 2017.

[15] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017.

[16] Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. *arXiv preprint arXiv:1902.00275*, 2019.

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

[18] Long Jin, Justin Lazarow, and Zhuowen Tu. Introspective classification with convolutional nets. In *Advances in Neural Information Processing Systems*, pages 823–833, 2017.

[19] Heewoo Jun, Rewon Child, Mark Chen, John Schulman, Aditya Ramesh, Alec Radford, and Ilya Sutskever. Distribution augmentation for generative modeling. In *Proceedings of Machine Learning and Systems 2020*, pages 10563–10576. 2020.

[20] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *arXiv preprint arXiv:2006.06676*, 2020.

[21] Taesup Kim and Yoshua Bengio. Deep directed generative models with energy-based probability estimation. *arXiv preprint arXiv:1606.03439*, 2016.

[22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[23] Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, pages 10215–10224, 2018.

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

[25] Rithesh Kumar, Anirudh Goyal, Aaron Courville, and Yoshua Bengio. Maximum entropy generators for energy-based models. *arXiv preprint arXiv:1901.08508*, 2019.

[26] Justin Lazarow, Long Jin, and Zhuowen Tu. Introspective neural networks for generative modeling. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2774–2783, 2017.

[27] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

[28] Kwonjoon Lee, Weijian Xu, Fan Fan, and Zhuowen Tu. Wasserstein introspective neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3702–3711, 2018.

[29] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Large-scale celebfaces attributes (celeba) dataset. *Retrieved August*, 15:2018, 2018.

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

[31] Radford M Neal. Annealed importance sampling. *Statistics and computing*, 11(2):125–139, 2001.

[32] Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

[33] Jiquan Ngiam, Zhenghao Chen, Pang W Koh, and Andrew Y Ng. Learning deep energy models. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 1105–1112, 2011.

[34] Erik Nijkamp, Mitch Hill, Tian Han, Song-Chun Zhu, and Ying Nian Wu. On the anatomy of mcmc-based maximum likelihood learning of energy-based models. *arXiv preprint arXiv:1903.12370*, 2019.

[35] Erik Nijkamp, Mitch Hill, Song-Chun Zhu, and Ying Nian Wu. On learning non-convergent short-run mcmc toward energy-based model. *arXiv preprint arXiv:1904.09770*, 2019.

[36] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016.

[37] Jascha Sohl-Dickstein, Eric A Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *arXiv preprint arXiv:1503.03585*, 2015.

[38] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, pages 11918–11930, 2019.

[39] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *arXiv preprint arXiv:2006.09011*, 2020.

[40] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. In *Advances in neural information processing systems*, pages 4790–4798, 2016.

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

[42] Jianwen Xie, Yang Lu, Ruiqi Gao, Song-Chun Zhu, and Ying Nian Wu. Cooperative training of descriptor and generator networks. *arXiv preprint arXiv:1609.09408*, 2016.

[43] Jianwen Xie, Yang Lu, Song-Chun Zhu, and Yingnian Wu. A theory of generative convnet. In *International Conference on Machine Learning*, pages 2635–2644, 2016.

[44] Jianwen Xie, Zilong Zheng, Xiaolin Fang, Song-Chun Zhu, and Ying Nian Wu. Cooperative training of fast thinking initializer and slow thinking solver for multi-modal conditional learning. *arXiv preprint arXiv:1902.02812*, 2019.

[45] 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.

[46] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[47] Junbo Zhao, Michael Mathieu, and Yann LeCun. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126*, 2016.

A Extended derivations

A.1 Derivation of equation 5

Let $\tilde{\mathbf{x}} = \mathbf{x} + \sigma\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$. Given the marginal distribution of

$$p_\theta(\mathbf{x}) = \frac{1}{Z_\theta}\exp(f_\theta(\mathbf{x})), \tag{18}$$

We can derive the conditional distribution of $\mathbf{x}$ given $\tilde{\mathbf{x}}$ as

$$\begin{aligned} p_\theta(\mathbf{x}|\tilde{\mathbf{x}}) &= p_\theta(\mathbf{x})p(\tilde{\mathbf{x}}|\mathbf{x})/p(\tilde{\mathbf{x}}) && (19)\\ &= \frac{1}{Z_\theta}\exp(f_\theta(\mathbf{x}))\frac{1}{(2\pi\sigma^2)^{\frac{n}{2}}}\exp(-\frac{1}{2\sigma^2}\|\tilde{\mathbf{x}}-\mathbf{x}\|^2)/p(\tilde{\mathbf{x}}) && (20)\\ &= \frac{1}{\tilde{Z}_\theta(\tilde{\mathbf{x}})}\exp\left(f_\theta(\mathbf{x}) - \frac{1}{2\sigma^2}\|\tilde{\mathbf{x}}-\mathbf{x}\|^2\right), && (21)\end{aligned}$$

where we absorb all the terms that are irrelevant of $\mathbf{x}$ as $\tilde{Z}_\theta(\tilde{\mathbf{x}})$.

A.2 Theoretical understanding

In this subsection, we analyze the asymptotic behavior of maximizing the recovery log-likelihood.

For model class $\{p_\theta(\mathbf{x}), \forall\theta\}$, suppose there exists θ^* such that $p_{\text{data}} = p_{\theta^*}$. According to the classical theory of MLE, let $\hat{\theta}_0$ be the point estimate by MLE. Then we have $\hat{\theta}$ is an unbiased estimator of θ^* with asymptotic normality:

$$\sqrt{n}(\hat{\theta}_0 - \theta^*) \to \mathcal{N}(0, \mathcal{I}_0(\theta^*)^{-1}), \tag{22}$$

where $\mathcal{I}_0(\theta) = \mathbb{E}_{\mathbf{x}\sim p_\theta}[-\nabla^2_\theta \log p_\theta(\mathbf{x})]$ is the Fisher information, and n is the number of observed samples.

Let $\hat{\theta}$ be the point estimate given by maximizing recovery log-likelihood, we can derive a result in parallel to that of MLE:

$$\sqrt{n}(\hat{\theta} - \theta^*) \to \mathcal{N}(0, \mathcal{I}(\theta^*)^{-1}), \tag{23}$$

where $\mathcal{I}(\theta) = \mathbb{E}_{p_\theta(\mathbf{x},\tilde{\mathbf{x}})}[-\nabla^2_\theta \log p_\theta(\mathbf{x}|\tilde{\mathbf{x}})]$. The relationship between $I_0(\theta)$ and $I(\theta)$ is that

$$\mathcal{I}_0(\theta) = \mathcal{I}(\theta) + \mathbb{E}_{p_\theta(\mathbf{x},\tilde{\mathbf{x}})}[-\nabla^2_\theta \log p_\theta(\tilde{\mathbf{x}})]. \tag{24}$$

Thus there is loss of information, but $\hat{\theta}$ is still an unbiased estimator of θ^* with asymptotic normality.

A.3 Difference between the scores of $p(\mathbf{x})$ and $p(\tilde{\mathbf{x}})$

For notation clarity, with $\tilde{\mathbf{x}} = \mathbf{x} + \boldsymbol{\epsilon}$, we let $\widetilde{p}$ be the distribution of $\tilde{\mathbf{x}}$, and p be the distribution of $\mathbf{x}$. Then for a smooth testing function with vanishing tails,

$$\begin{aligned} \mathbb{E}[h(\tilde{\mathbf{x}})] &= \mathbb{E}[h(\mathbf{x}+\boldsymbol{\epsilon})] && (25)\\ &\doteq \mathbb{E}[h(\mathbf{x}) + h'(\mathbf{x})\boldsymbol{\epsilon} + h''(\mathbf{x})\boldsymbol{\epsilon}^2/2] && (26)\\ &= \mathbb{E}[h(\mathbf{x})] + \mathbb{E}[h''(\mathbf{x})]\sigma^2/2. && (27)\end{aligned}$$

Integral by part,

$$\mathbb{E}[h''(\mathbf{x})] = \int h''(\mathbf{x})p(\mathbf{x})d\mathbf{x} = -\int h'(\mathbf{x})p'(\mathbf{x})d\mathbf{x} = \int p''(\mathbf{x})h(\mathbf{x})d\mathbf{x}. \tag{28}$$

Thus we have the heat equation

$$\widetilde{p}(\mathbf{x}) = p(\mathbf{x}) + p''(\mathbf{x})\sigma^2/2. \tag{29}$$

The score

$$\begin{aligned} \nabla_{\mathbf{x}}\log\tilde{p}(\mathbf{x}) &= \nabla_x \log p(\mathbf{x}) + \nabla_{\mathbf{x}}\log(1 + p''(\mathbf{x})/p(\mathbf{x})\sigma^2/2) && (30)\\ &\doteq \nabla_{\mathbf{x}}\log p(\mathbf{x}) + \nabla_{\mathbf{x}}[p''(\mathbf{x})/p(\mathbf{x})]\sigma^2/2. && (31)\end{aligned}$$

Thus the difference between the score of p and $\widetilde{p}$ is of the order σ^2, which is negligible when σ^2 is small.

A.4 Estimating the partition function

We can utilize the sequence of learned distributions of $\mathbf{y}_t$ ($= \sqrt{1-\sigma_{t+1}^2}\mathbf{x}_t$) to estimate the partition function. Specifically, the marginal distribution of $\mathbf{y}_t$ is

$$p_\theta(\mathbf{y}_t) = \frac{1}{Z_{\theta,t}} \exp\left(f_\theta(\mathbf{y}_t, t)\right) \tag{32}$$

We can estimate the ratio of the partition functions at two consecutive time steps using importance sampling

$$\frac{Z_{\theta,t}}{Z_{\theta,t+1}} = \mathbb{E}_{p_\theta(\mathbf{y}_{t+1})}\left[\exp(f_\theta(\mathbf{y}, t) - f_\theta(\mathbf{y}, t+1))\right] \tag{33}$$

$$\doteq \frac{1}{M}\sum_{i=1}^{M}\left[\exp(f_\theta(\mathbf{y}_{t+1,i}, t) - f_\theta(\mathbf{y}_{t+1,i}, t+1))\right], \tag{34}$$

where $\mathbf{y}_{t+1,i}$ are samples generated by progressive sampling. Starting from $t = T$, where $p_T(x)$ follows Gaussian distribution, we can compute $\log Z_t$ along the reverse path of the diffusion process, until we reach $t = 0$:

$$Z_{\theta,0} = Z_{\theta,T}\prod_{t=0}^{T-1}\frac{Z_{\theta,t}}{Z_{\theta,t+1}}. \tag{35}$$

In practice, since the ratio given by MCMC samples can vary across many orders of magnitude, it is more meaningful to estimate

$$\log Z_{\theta,0} = \log Z_{\theta,T} + \sum_{t=0}^{T-1}\log\frac{Z_{\theta,t}}{Z_{\theta,t+1}}. \tag{36}$$

Unfortunately, although equation 34 is an unbiased estimator of $Z_{\theta,t}/Z_{\theta,t+1}$, the logarithm of this estimator is generally a stochastic lower bound of $\log(Z_{\theta,t}/Z_{\theta,t+1})$ [13]. However, as we show below, this bound will gradually converge to an unbiased estimator of $\log(Z_{\theta,t}/Z_{\theta,t+1})$, as the number of samples becomes large. Specifically, let A be the estimator in equation 34, μ be the true value of $Z_{\theta,t}/Z_{\theta,t+1}$. We have $\mathbb{E}[A] = \mu$, then by second order Taylor expansion,

$$\mathbb{E}[\log A] \doteq \mathbb{E}\left[\log\mu + \frac{1}{\mu}(A-\mu) - \frac{1}{2\mu^2}(A-\mu)^2\right] \tag{37}$$

$$= \log\mu - \frac{1}{2\mu^2}\mathrm{Var}(A). \tag{38}$$

By *law of large number*, $\mathrm{Var}(A) \to 0$ as $M \to \infty$, and thus $\mathbb{E}[\log A] \to \log\mu$. This is also consistent with the estimation curves in the right subfigure of Figure 6: since $\mathrm{Var}(A) \geq 0$, the estimation curve increases from below as the number of samples becomes larger. When the curve becomes stable, it indicates the convergence.

B Experimental details

Model architecture. Our network structure is based on Wide ResNet [46]. Table 4 lists the detailed network structures of various resolutions. The number of ResBlocks at every level N is a hyperparameter that we sweep over. The values of N for various datasets are listed in Table 5. Each ResBlock consists of two Conv2D layers. For the second Conv2D layer, we use zero initialization for the weights, and add a trainable channel-wise scaling parameter to the output. We remove the weight normalization, and use leaky ReLU (slope $= 0.2$) as the activation function in ResBlocks. Spectral normalization [30] is used to regularize parameters in Conv2D layer, ResBlocks and Dense layer. For encoding time step t, we follow the scheme in [17]. Specifically, the time step t is first transformed into sinusoidal embedding, and then two Dense layers is added. The time embedding is added after the first Conv2D layer of each ResBlock.

Training. We use Adam [22] optimizer for all the experiments. We find that for high resolution images, using a smaller β_1 in Adam help stabilize training. We use learning rate 0.0001 for all the experiments. For the values of β_1, batch sizes and the number of training iterations for various datasets, see Table 5.

Datasets. We use the following datasets in our experiments: CIFAR-10 [24], CelebA [29] and LSUN [45]. CIFAR-10 is of resolution 32×32, and contains $50,000$ training images and $10,000$ test images. CelebA contains 202,599 face images, of which 162,770 are training images and 19,962 are test images. For processing, we first clip each image to 178×178 and then resize it to 64×64. For LSUN, we use church_outdoor and bedroom categories, which contains 126,227 and 3,033,042 training images respectively. Both categories contain 300 test images. For processing, we first crop each image to a square image of the smaller size among the height and weight, and then we resize it to 64×64 or 128×128. For resizing, we set antialias to True. We apply horizontal random flip as data augmentation for all datasets during training.

Evaluation metrics. We use FID and inception scores as quantitative evaluation metrics of sample quality. On CIFAR-10, we calculate FID and inception scores on 50,000 samples using the original code from [36] and [15].

Table 4: Model architectures of various solutions. N is a hyperparameter that we sweep over.

(a) Resolution 32×32

3×3 Conv2D, 128
N ResBlocks, 128 Downsample 2×2
N ResBlocks, 256 Downsample 2×2
N ResBlocks, 256 Downsample 2×2
N ResBlocks, 256
ReLU, global sum Dense 1

(b) Resolution 64×64

3×3 Conv2D, 128
N ResBlocks, 128 Downsample 2×2
N ResBlocks, 256 Downsample 2×2
N ResBlocks, 256 Downsample 2×2
N ResBlocks, 256 Downsample 2×2
N ResBlocks, 512
ReLU, global sum Dense 1

(c) Resolution 128×128

3×3 Conv2D, 128
N ResBlocks, 128 Downsample 2×2
N ResBlocks, 256 Downsample 2×2
N ResBlocks, 256 Downsample 2×2
N ResBlocks, 256 Downsample 2×2
N ResBlocks, 512 Downsample 2×2
N ResBlocks, 512
ReLU, global sum Dense 1

(d) Time embedding (temb)

sinusoidal embedding
Dense, leakyReLU
Dense

(e) ResBlock

leakyReLU, 3×3 Conv2D
+ Dense(leakyReLU(temb))
leakyReLU, 3×3 Conv2D
+ input

Table 5: Hyperparameters of various datasets.

Dataset	N	β_1 in Adam	Batch size	Training iterations
CIFAR-10	5	0.9	256	50k
CelebA	2	0.9	128	100k
LSUN church_outdoor 64^2	2	0.9	128	100k
LSUN bedroom 64^2	2	0.9	128	100k
LSUN church_outdoor 128^2	2	0.5	64	100k
LSUN bedroom 128^2	5	0.5	64	56k

C Additional experimental results

C.1 Additional interpolation results

Figures 8, 9 and 10 display more examples of interpolation between two generated samples on Celeba 64^2, LSUN church_outdoor 128^2 and LSUN bedroom 128^2.

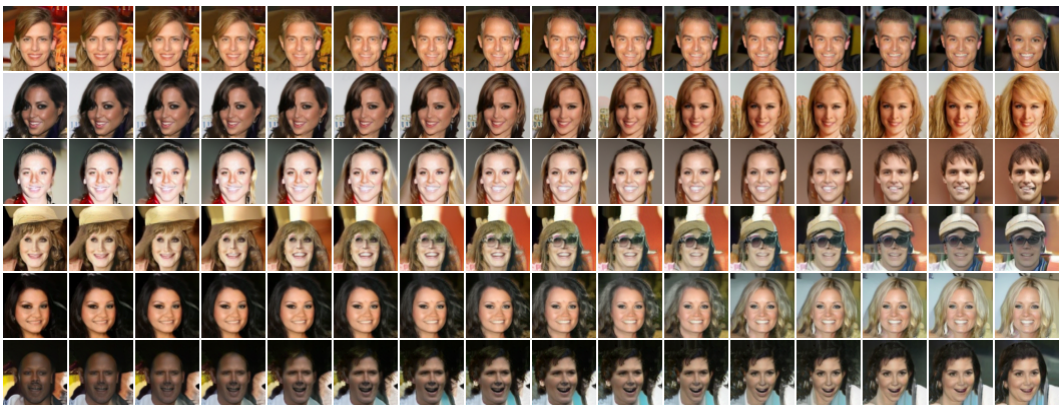

Figure 8: Interpolation results between the leftmost and rightmost generated samples on CelebA 64×64.

Figure 9: Interpolation results between the leftmost and rightmost generated samples on LSUN church_outdoor 128×128.

C.2 Additional uncurated samples

Figures 11, 12, 13, 14, 15 and 16 show uncurated samples from the learned models under *T6* setting on CIFAR-10, CelebA 64^2, LSUN church_outdoor 128^2, LSUN bedroom 128^2, LSUN church_outdoor 64^2 and LSUN bedroom 64^2 datasets.

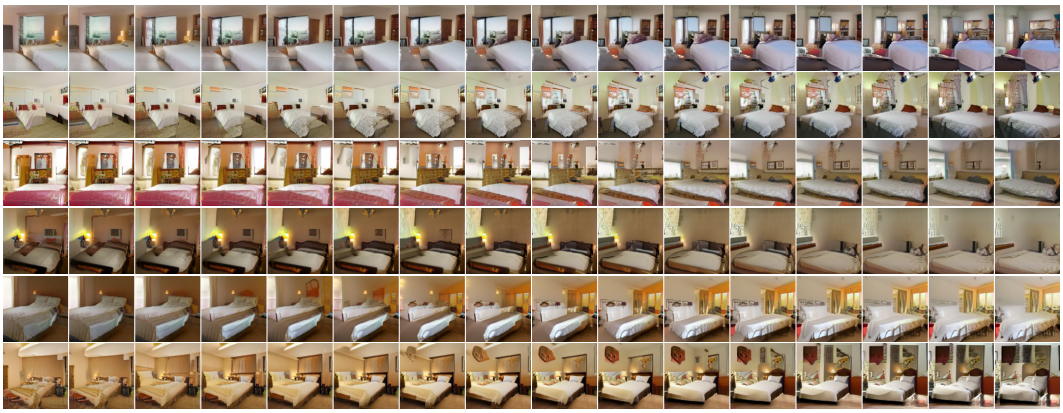

Figure 10: Interpolation results between the leftmost and rightmost generated samples on LSUN bedroom 128×128.

Figure 11: Generated samples on CIFAR-10.

Figure 12: Generated samples on CelebA 64×64.

Figure 13: Generated samples on LSUN church_outdoor 128×128.

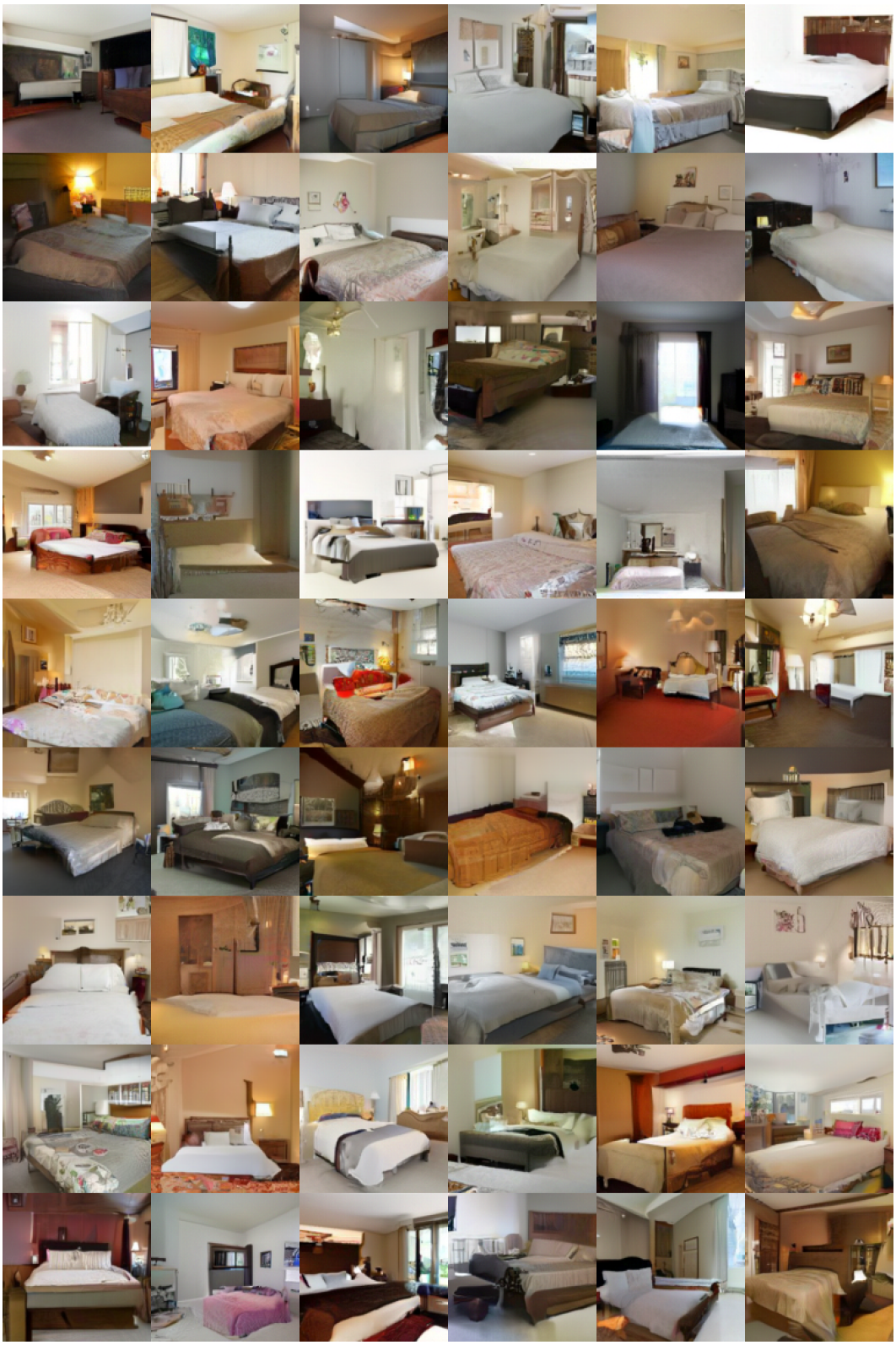

Figure 14: Generated samples on LSUN bedroom 128×128.

Figure 15: Generated samples on LSUN church_outdoor 64×64.

Figure 16: Generated samples on LSUN bedroom 64×64.