Improving Adversarial Energy-Based Model via Diffusion Process

Cong Geng [1] **Tian Han** [2] **Peng-Tao Jiang** [1] **Hao Zhang** [1] **Jinwei Chen** [1] **Søren Hauberg** [3] **Bo Li** [1]

Abstract

Generative models have shown strong generation ability while efficient likelihood estimation is less explored. Energy-based models (EBMs) define a flexible energy function to parameterize unnormalized densities efficiently but are notorious for being difficult to train. Adversarial EBMs introduce a generator to form a minimax training game to avoid expensive MCMC sampling used in traditional EBMs, but a noticeable gap between adversarial EBMs and other strong generative models still exists. Inspired by diffusion-based models, we embedded EBMs into each denoising step to split a long-generated process into several smaller steps. Besides, we employ a symmetric Jeffrey divergence and introduce a variational posterior distribution for the generator's training to address the main challenges that exist in adversarial EBMs. Our experiments show significant improvement in generation compared to existing adversarial EBMs, while also providing a useful energy function for efficient density estimation.

1. Introduction

Energy-based models (EBMs) are known as one type of generative models that draw inspiration from physics and have been widely studied in machine learning (Hopfield, 1982; Hinton & Sejnowski, 1983; Smolensky, 1986). EBMs define an unnormalized probability distribution over data space from a Gibbs density, which can be useful for several visual tasks, such as image classification (Grathwohl et al., 2020), out-of-distribution (OOD) detection (Liu et al., 2020) or semi-supervised learning (Gao et al., 2020). However, training an EBM using maximum likelihood estimation can be challenging due to the lack of a closed-form expression for the normalization constant. MCMC-based EBMs (Du & Mordatch, 2019; Nijkamp et al., 2019) evaluate the gradient of the objective through Markov chain Monte Carlo (MCMC) sampling on the defined energy function, which can be computationally expensive for both training and sampling. Adversarial EBMs (Grathwohl et al., 2021; Geng et al., 2021) introduce a *generator* to form a minimax game between alternative optimization of this generator and energy function, allowing for MCMC-free EBM training and fast sampling.

Although adversarial EBMs have great potential in distribution modeling, they still have some limitations that can be mainly attributed to three reasons. First, as is pointed out in Mescheder et al. (2018) and Geng et al. (2021), minimax training can be unstable if two alternative optimization steps are not well balanced. This instability poses a significant challenge in fitting the marginal energy distribution through adversarial training, particularly when dealing with large-scale, complex, and multi-modal data distributions. Secondly, most adversarial EBMs adopt KL divergence to optimize the generator. Since KL divergence is an asymmetric measure of divergence, relying solely on KL divergence may not be sufficient (Arjovsky et al., 2017). Thirdly, optimizing the generator requires computing an intractable entropy term, leading to a trade-off between generation and density estimation (Grathwohl et al., 2021).

To address these issues, we get inspiration from diffusion-based models (Ho et al., 2020) which specify a diffusion process to transform a data distribution by adding Gaussian noise with multiple steps until obtaining an approximate Gaussian distribution. A denoising diffusion process is learned by minimizing the KL divergence between true conditional denoising distribution $q(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ and a parameterized conditional distribution $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ at each noise step. Diffusion Recovery Likelihood (DRL) (Gao et al., 2021) and Denoising diffusion GAN (DDGAN) (Xiao et al., 2022) combined EBM or GAN with diffusion process respectively. They trained a sequence of EBMs or GANs by matching true and modeled denoising distributions at each time step. The efficacy of these methods demonstrates that matching two denoising distributions is more tractable, as sampling from these conditional distributions is much easier than sampling from their marginal distributions. However, DDGAN lacks a useful density estimate for its discriminator

[1]vivo Mobile Communication Co., Ltd, China [2]Department of Computer Science, Stevens Institute of Technology, USA [3]Technical University of Denmark, Copenhagen, Denmark. Correspondence to: Søren Hauberg <sohau@dtu.dk>, Bo Li <libra@vivo.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

and DRL has only one energy function, making sampling inefficient. These limitations are precisely where the strengths of EBM lie. Based on these arguments, we also incorporate adversarial EBMs into diffusion processes at each noise step. Thus for each EBM, the target distribution is a conditional distribution that is less multi-modal and easier to learn. Besides, with our defined generated denoising distribution through a generator, a symmetric Jeffrey divergence (Jeffreys, 1946) can be easily employed to remedy inadequate fitting, and a variational posterior distribution is introduced to compute the entropy term. Therefore, three main challenges in training adversarial EBMs can be overcome. To our knowledge, it is the first time that adversarial EBM has been well integrated into the framework of diffusion models. In summary, the contributions of our paper are as follows:

- We propose an MCMC-free training framework for EBMs to incorporate a sequence of adversarial EBMs into a denoising diffusion process. This framework avoids MCMC in both training and sampling.
- We learn EBMs by optimizing conditional denoising distributions instead of marginal distributions to alleviate the training burden. We train the generator by minimizing a symmetric Jeffrey divergence to help distribution matching and introduce a variational posterior distribution to compute the entropy term.
- We demonstrate that our model achieves significant improvements on sample quality compared to existing adversarial EBMs. We also verify the ability of our energy function in density estimation.

2. Related Work

There is a long history of EBMs in machine learning dating back to Hopfield networks (Hopfield, 1982) and Boltzmann machines (Hinton & Sejnowski, 1983). Recently, deep EBMs have grown in popularity, especially for image generation, and can broadly be classified into two categories. MCMC-based methods simulate Markov chains during training (Du & Mordatch, 2019; Pang et al., 2020; Arbel et al., 2021) or sampling (Song et al., 2020; Song & Ermon, 2019). This can be expensive, slow, and hard to control. Cooperative learning methods (Xie et al., 2020; 2021; 2022; Cui & Han, 2023) jointly train a generator and an energy function for MCMC teaching, using the generator as a fast initializer for Langevin sampling to alleviate the MCMC burden. They, however, remain inefficient.

The other category is adversarial EBMs (Zhai et al., 2016; Geng et al., 2021; Kumar et al., 2019; Zhao et al., 2017), which introduce a 'generator' to form a minimax game between the energy function and the introduced generator. Han et al. (2019; 2020) and Kan et al. (2022) explained adversarial EBM from the view of triangle divergence and bi-level optimization problem respectively. Adversarial EBMs inherit the advantages of GANs and avoid MCMC sampling, but they also bring the risk of unstable training.

The original diffusion-based model (Ho et al., 2020; Sohl-Dickstein et al., 2015) learns a finite-time reversed diffusion from a forward process and gets strong performance on image generation, but the sampling process is slow. Several works (Song et al., 2021a; Lu et al., 2022b; Bao et al., 2022) improve training-free samplers and significantly improve the sampling speed going from 1000 steps sampling to only a few. DDGAN (Xiao et al., 2022) proposed to model each denoising step using a multimodal conditional GAN, tackling the generative learning trilemma of fast sampling while maintaining strong mode coverage and sample quality. However, its discriminator fails to provide a density estimate, which is crucial for many tasks (Du et al., 2023; Zhang et al., 2022; Du et al., 2022).

Although some diffusion-based methods (Kingma et al., 2021; Song et al., 2021b; Lu et al., 2022a) use score functions or variational lower bounds to estimate the density of the data distribution, these methods tend to be complicated and inefficient, which is precisely where energy-based models excel. DRL (Gao et al., 2021) and CDRL (Zhu et al., 2024) tried to combine EBM with a diffusion-based model and got state-of-the-art generation performance on several image datasets. Still, they relied on MCMC sampling which remains inefficient.

3. Denoising Diffusion Adversarial EBM

Preliminaries Let $\mathbf{x} \sim q(\mathbf{x})$ be a training example from an underlying data distribution. An EBM is specified in terms of an energy function $E_\theta : \mathcal{X} \to \mathbb{R}$, that is parameterized by θ, defines a probability distribution over $\mathcal{X}$ from the Gibbs distribution:

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

where Z_θ is the normalization constant (or partition function). In principle, any density can be described in this way for a suitable choice of E_θ. We learn the EBM through maximum likelihood estimation (MLE), i.e. we seek θ^* that maximizes the data log-likelihood:

$$\begin{aligned} \mathcal{L}(\theta^*) &:= \max_\theta \mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\left[\log p_\theta(\mathbf{x})\right] \\ &= \max_\theta \{\mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\left[E_\theta(\mathbf{x})\right] - \log Z_\theta\}. \end{aligned} \tag{2}$$

The fundamental challenge for training EBMs is the lack of a closed-form expression for the normalization constant. A common way to solve this is to approximate the gradient of $\mathcal{L}(\theta)$ directly and apply gradient-based optimization:

$$\frac{\partial \mathcal{L}}{\partial \theta} = \mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\left[\frac{\partial}{\partial \theta} E_\theta(\mathbf{x})\right] - \mathbb{E}_{\mathbf{x}\sim p_\theta(\mathbf{x})}\left[\frac{\partial}{\partial \theta} E_\theta(\mathbf{x})\right]. \tag{3}$$

Estimating the above equation needs MCMC sampling from the energy distribution $p_\theta(\mathbf{x})$, which is both unstable and expensive. Adversarial EBMs (Grathwohl et al., 2021; Geng et al., 2021; Han et al., 2019; Kan et al., 2022) avoid MCMC by introducing a variational distribution p_ϕ to approximate p_θ. Therefore, a minimax game can be formed to alternately optimize two adversarial steps:

1. $\min_{p_\phi} D(p_\phi, p_\theta)$ with a certain proper divergence D.
2. $\max_{E_\theta} \left[\mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\left[E_\theta(\mathbf{x})\right] - \mathbb{E}_{\mathbf{x}\sim p_\phi(\mathbf{x})}\left[E_\theta(\mathbf{x})\right]\right]$.

This two-step training strategy can be interpreted as a bi-level optimization problem (Liu et al., 2022), where the minimization step is a lower-level (LL) subproblem and the maximization step an upper-level (UL) subproblem. Geng et al. (2021) demonstrated that if p_ϕ fails to be optimized, it runs the risk of maximizing an upper bound, causing a noticeable performance gap compared with mainstream generative models in complex and sparse datasets.

3.1. Adversarial EBM with Denoising Diffusion Process

To make adversarial EBMs scalable to complex distributions, we borrow the diffusion-based framework to split the generation process into multiple steps. For each step, we only need to learn a conditional distribution rather than a complex marginal distribution. Similar to DDPM (Ho et al., 2020), we also use a Markov chain for the diffusion process. Specifically, for the forward process, we gradually add noise to the real data $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ in T steps with pre-defined variance schedule β_t:

$$q\left(\mathbf{x}_{1:T} \mid \mathbf{x}_0\right) = \prod_{t\geq 1} q\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right), \tag{4}$$

$$q\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}\right).$$

Then we can get the denoising distribution $q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right) = \frac{q(\mathbf{x}_{t-1})q(\mathbf{x}_t|\mathbf{x}_{t-1})}{q(\mathbf{x}_t)}$ as our target distribution to learn. In diffusion-based models, the regular training objective is to minimize the KL divergence between the true denoising distribution $q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)$ and a modeled one $p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)$ for each t, i.e.

$$\mathcal{L} = -\sum_{t\geq 1} \mathbb{E}_{q(\mathbf{x}_t)}[D_{\mathrm{KL}}(q\left(\mathbf{x}_{t-1}|\mathbf{x}_t\right) \| p_\theta\left(\mathbf{x}_{t-1}|\mathbf{x}_t\right))] + C. \tag{5}$$

Inspired by DRL (Gao et al., 2021), we design a sequence of conditional EBM using an energy function conditioned on t to define our modeled denoising distribution,

$$p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right) = \frac{\exp(E_\theta(\mathbf{x}_{t-1}, t-1))q(\mathbf{x}_t|\mathbf{x}_{t-1})}{\tilde{Z}_{\theta,t}(\mathbf{x}_t)} \tag{6}$$

$$\tilde{Z}_{\theta,t}(\mathbf{x}_t) = \int \exp(E_\theta(\mathbf{x}_{t-1}, t-1))q(\mathbf{x}_t|\mathbf{x}_{t-1})\mathrm{d}\mathbf{x}_{t-1}$$

where $E_\theta(\boldsymbol{x}_t, t) : \mathbb{R}^D \times \mathbb{R} \to \mathbb{R}$ is the energy function parameterized by θ. Thus $\mathcal{L}$ can be simplified and expressed by energy function:

$$\begin{aligned}\mathcal{L} = \sum_{t\geq 1} \mathbb{E}_{q(\mathbf{x}_{t-1},\mathbf{x}_t)}\big[&E_\theta(\mathbf{x}_{t-1}, t-1) \\ &+ \log q(\mathbf{x}_t|\mathbf{x}_{t-1}) - \log \tilde{Z}_{\theta,t}(\mathbf{x}_t)\big].\end{aligned} \tag{7}$$

It is easy to see that when Eq. (7) is optimized, $p_\theta(\mathbf{x}_t) = \frac{\exp\left(E_\theta(\mathbf{x}_t,t)\right)}{Z_{\theta,t}} = q(\mathbf{x}_t)$ (see Appendix).

Since $\tilde{Z}_{\theta,t}$ is intractable, if we follow the common choice to compute the gradient of the objective:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \theta} = &\sum_{t\geq 1} \mathbb{E}_{q(\mathbf{x}_{t-1},\mathbf{x}_t)}\left[\frac{\partial}{\partial\theta} E_\theta(\mathbf{x}_{t-1}, t-1)\right] \\ &- \mathbb{E}_{q(\mathbf{x}_t)p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}\left[\frac{\partial}{\partial\theta} E_\theta(\mathbf{x}_{t-1}, t-1)\right],\end{aligned} \tag{8}$$

the second term has to be estimated by MCMC-sampling from the distribution $q(\mathbf{x}_t)p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ (Gao et al., 2021), which can be computational demanding and unstable during training.

Therefore, we adopt an adversarial EBM by introducing a variational conditional distribution $p_\phi\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)$ parameterized by ϕ to form a two-step minimax game:

DDAEBM minimax game

1. $\min_{p_\phi} \sum_{t\geq 1} \mathbb{E}_{q(\mathbf{x}_t)}\left[D\left(p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t), p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)\right)\right]$
2. $\max_{E_\theta} \sum_{t\geq 1} \big[\mathbb{E}_{q(\mathbf{x}_{t-1},\mathbf{x}_t)} E_\theta(\mathbf{x}_{t-1}, t-1) - \mathbb{E}_{q(\mathbf{x}_t)p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)} E_\theta(\mathbf{x}_{t-1}, t-1)\big]$

This adversarial training strategy alternately optimizes p_ϕ and E_θ, maintaining one as fixed while optimizing the other. This gives a tractable and MCMC-free approach to EBM training. We simply refer to our proposed method as **D**enoising **D**iffusion **A**dversarial **E**nergy-**B**ased **M**odel (DDAEBM).

3.2. Minimization w.r.t. p_ϕ

First, we need to have a specific form of our introduced variational conditional distribution $p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)$. DDIM (Song et al., 2021a) and DDPM (Ho et al., 2020) define $p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)$ as a Gaussian distribution to match the Gaussian posterior distribution $q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0\right)$. However, as demonstrated in DRL, this normal distribution approximation is only accurate when β_t is small, it may not be reasonable when there are few denoising steps, as the denoising

distributions can be complex and multimodal. Therefore, we define $p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)$ with a reparameterization trick,

$$p_\phi(\mathbf{x}_{t-1} \mid \mathbf{x}_t) := \int p_\phi(\mathbf{x}_0 \mid \mathbf{x}_t)\, q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \boldsymbol{x}_0)\, \mathrm{d}\mathbf{x}_0 = \int p(\mathbf{z}) q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0 = G_\phi(\mathbf{x}_t, \mathbf{z}, t))\, \mathrm{d}\mathbf{z}, \quad (9)$$

where $p_\phi(\mathbf{x}_0 \mid \mathbf{x}_t)$ is an implicit distribution imposed by a neural network called *the generator* $G_\phi(\mathbf{x}_t, \mathbf{z}, t) : \mathbb{R}^D \times \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^D$. Here, $p(\mathbf{z})$ is a d-dimensional latent variable following a standard Gaussian distribution $\mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{I})$, and $q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)$ is the posterior distribution defined in forward diffusion process. It is worth noting that this definition is also explored by Xiao et al. (2022), which has been verified to be effective in fitting complex and multi-modal conditional distributions, although they don't leverage an EBM structure to characterize the data distribution.

The KL divergence is, by far, the most common choice of divergence D for adversarial EBM training (Geng et al., 2021; Grathwohl et al., 2021; Zhai et al., 2016). However, given that the KL divergence is an asymmetrical metric, it compels $p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)$ to chase the major modes of $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$. Relying solely on the KL divergence as our objective can, therefore, be insufficient for the generator to effectively capture the energy distribution (Geng et al., 2021). Following Kan et al. (2022), we choose a symmetric Jeffrey divergence, to integrate both KL divergence and reverse KL divergence into our objective:

Ideal objective

$$\min_{p_\phi} \sum_{t\geq 1} \mathbb{E}_{q(\mathbf{x}_t)} \big[D_{\mathrm{KL}}(p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t) \| p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)) + D_{\mathrm{KL}}(p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \| p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)) \big]. \quad (10)$$

We divide our ideal objective into two terms for separate handling:

$$L_1 = \sum_{t\geq 1} \mathbb{E}_{q(\mathbf{x}_t)} D_{\mathrm{KL}}(p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t) \| p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)), \quad (11)$$

$$L_2 = \sum_{t\geq 1} \mathbb{E}_{q(\mathbf{x}_t)} D_{\mathrm{KL}}(p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \| p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)). \quad (12)$$

Bounding the first term of the ideal objective. To minimize our ideal objective, we first handle Eq. (11). It can be simplified by omitting some unrelated terms and obtaining:

$$\sum_{t\geq 1} -H[p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)] - \mathbb{E}_{q(\mathbf{x}_t) p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)} \log p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \quad (13)$$

Computing the entropy of the generated distribution is always a challenging task in EBMs. Several approaches (Kumar et al., 2019; Grathwohl et al., 2021; Geng et al., 2021) have been proposed for efficient entropy approximation, but these are all either time or memory-demanding. We need to compute a conditional entropy:

$$H[p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)] = H[p_\phi(\mathbf{x}_{t-1}, \mathbf{z}|\mathbf{x}_t)] - H[p_\phi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)] \quad (14)$$

where

$$p_\phi(\mathbf{x}_{t-1}, \mathbf{z}|\mathbf{x}_t) = p(\mathbf{z}) q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0 = G_\phi(\mathbf{x}_t, \mathbf{z}, t)). \quad (15)$$

$H[p_\phi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)]$ is intractable while $H[p_\phi(\mathbf{x}_{t-1}, \mathbf{z}|\mathbf{x}_t)]$ is a constant that can be ignored (see Appendix). Minimizing Eq. (13) can be problematic since we do not have access to $H[p_\phi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)]$, and we instead minimize a variational upper bound by introducing an approximate Gaussian posterior $q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)$. Here the mean and variance are represented through an encoder parameterized by ψ with $\mathbf{x}_{t-1}$, $\mathbf{x}_t$ and t as the input. We can easily obtain:

$$H[p_\phi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)] \leq -\mathbb{E}_{q(\mathbf{x}_t) p_\phi(\mathbf{x}_{t-1}, \mathbf{z}|\mathbf{x}_t)} \log q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t). \quad (16)$$

Expressing $\log p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ with Eq. (6) and applying Eq. (16) to Eq. (13), the first term Eq. (11) can be minimized by its simplified variational upper bound:

Upper bound of L_1

$$L_1 \leq \sum_{t\geq 1} \mathbb{E}_{q(\mathbf{x}_t) p_\phi(\mathbf{x}_{t-1}, \mathbf{z}|\mathbf{x}_t)} \Big[-E_\theta(\mathbf{x}_{t-1}, t-1) - \log q(\mathbf{x}_t|\mathbf{x}_{t-1}) - \log q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t) \Big] \quad (17)$$

Bounding the second term of the ideal objective. We now proceed with the minimization of our ideal objective, treating the second term Eq. (12). This term can also be bounded from above by utilizing an Evidence Lower Bound (ELBO) associated with our introduced Gaussian posterior $q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)$, since

$$\log p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t) \geq -D_{KL}(q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t) \| p(\mathbf{z})) + \mathbb{E}_{q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)} \log q(\mathbf{x}_{t-1}|\mathbf{x}_t, G_\phi(\mathbf{x}_t, \mathbf{z}, t)) \quad (18)$$

We can get the upper bound of Eq. (12) by applying Eq. (18):

Upper bound of L_2

$$L_2 \leq \sum_{t\geq 1} \mathbb{E}_{q(\mathbf{x}_t) p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} D_{KL}(q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t) \| p(\mathbf{z})) - \mathbb{E}_{q(\mathbf{x}_t) p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)} \log q(\mathbf{x}_{t-1}|\mathbf{x}_t, G_\phi(\mathbf{x}_t, \mathbf{z}, t)) \quad (19)$$

A Monte Carlo approximation of Eq. (19) requires sampling from $q(\mathbf{x}_t) p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$. As $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is designed to fit $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$, we use samples of $q(\mathbf{x}_{t-1}, \mathbf{x}_t)$ with an importance ratio to replace those of $q(\mathbf{x}_t) p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, as done in

BiDVL (Kan et al., 2022):

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \lambda(t-1)q(\mathbf{x}_{t-1}|\mathbf{x}_t) \tag{20}$$

The importance ratio is designed inspired by ACT (Kong et al., 2023):

$$\lambda(t-1) = w\left(t'(t)\right)^{\log\frac{1}{2}\left(\frac{w_{\text{mid}}}{w}\right)}, \tag{21}$$

where w and w_{mid} are the hyperparameters selected based on the datasets. $t'(\cdot)$ is a function denoting the time of $\mathbf{x}_t$ in the Variance Preserving (VP) SDE (Song et al., 2021c), as our diffusion process can be viewed as the discretization of the continuous-time VP SDE (see Appendix). The ultimate overall objective for implementation of the minimization step is the sum of Eq. (17) and Eq. (19) w.r.t p_ϕ and q_ψ.

3.3. Maximization step w.r.t. E_θ

After minimizing step w.r.t p_ϕ and q_ψ, we approximately assume $p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t) = p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$. Next, we should optimize the energy function by maximizing:

$$\begin{aligned}\sum_{t\geq 1}\big[&\mathbb{E}_{q(\mathbf{x}_{t-1},\mathbf{x}_t)}E_\theta(\mathbf{x}_{t-1},t-1)\\ &-\mathbb{E}_{q(\mathbf{x}_t)p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)}E_\theta(\mathbf{x}_{t-1},t-1)\big]\end{aligned} \tag{22}$$

Similar to most adversarial EBMs, we also find it helpful to stabilize training by adding a ℓ_2-regularizer for the gradient of our energy:

$$\frac{\gamma}{2}\mathbb{E}_{q(\mathbf{x}_{t-1})}\left[\left\|\nabla_{\mathbf{x}_{t-1}}\left[E_\theta\left(\mathbf{x}_{t-1},t-1\right)+\log q(\mathbf{x}_t|\mathbf{x}_{t-1})\right]\right\|^2\right] \tag{23}$$

where γ is the regularization coefficient. This regularization is a gradient penalty that alleviates training instability due to insufficient training at the minimization step (Grathwohl et al., 2021; Kumar et al., 2019).

4. Experiments

We evaluate our DDAEBM in different scenarios across different data scales from 2-dimension toy datasets to large-scale image datasets. We test our energy function mainly on toy datasets and MNIST datasets which are easy to visualize and intuitive to measure. For large-scale datasets, we focus on image generation. We further perform some additional studies such as out-of-distribution (OOD) detection and ablation studies to verify our model's superiority. We briefly introduce the network architecture design, while additional implementation details are presented in the Appendix. For the generator trained on large image datasets, we adopt the same modified NCSN++ architecture as DDGAN (Xiao et al., 2022) or DDPM++ with a slightly different structure as described in Score SDE (Song et al., 2021c), where diffused samples $\boldsymbol{x}_t$, time t and latent variable $\mathbf{z}$ are the input of the network. For the energy function, we adopt traditional NCSN++ or DDPM++ architectures from Score SDE except that we remove the last scale-by-sigma operation and replace it with a negative ℓ_2 norm between input $\boldsymbol{x}_t$ and the output of the Unet (Du et al., 2023), i.e.

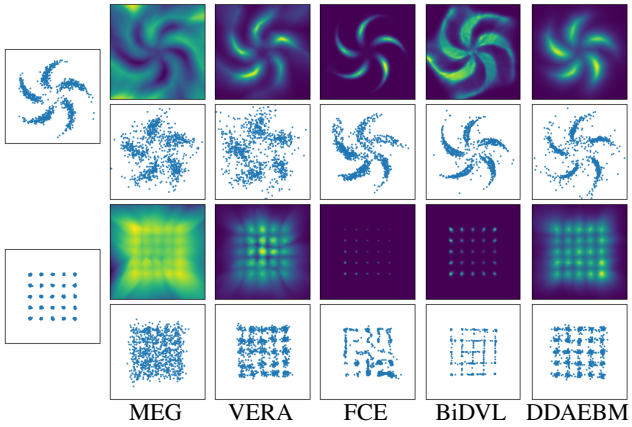


Figure 1. Density estimation and generation on 25-Gaussians and pinwheel datasets. For each dataset, the first row shows the estimated densities and the second row shows generated samples.

$$E_\theta\left(\mathbf{x}_t,t\right) = -\frac{1}{2}\|\mathbf{x}_t - f_\theta\left(\mathbf{x}_t,t\right)\|^2. \tag{24}$$

4.1. Performance on 2D synthetic data

Fig. 1 demonstrates the density estimation and generation results of our DDAEBM on the 25-Gaussians and pinwheel datasets, which are two challenging toy datasets. We compare our model with several mainstream adversarial EBMs. These methods are all MCMC-free through training a generator and an energy function alternatively in two steps. We can observe that only our DDAEBM can get satisfying performance both for generation and density estimation on these two datasets. For pinwheel, most methods fail on density estimation except FCE and our DDAEBM. FCE (Gao et al., 2020) trains an energy function using constrastive learning which has a strong ability on density estimation, but it uses a GLOW network as its generator which limits its ability on generation. For 25-Gaussians, although the generation of our DDAEBM is more dispersed than that of BiDVL around each mode, samples are mostly centered around each mode instead of being between two modes.

4.2. Fitting flow models with energy function

Since our energy function is an unnormalized likelihood estimator with a difficult-to-estimate normalization constant, we consider an example where we can evaluate the exact log-likelihood associated with our energy function. Following Grathwohl et al. (2021), we train the NICE model (Dinh et al., 2015) as an energy function on the MNIST dataset. NICE is a normalizing flow model that allows for exact

Table 1. Log-likelihoods for NICE models as the energy function

Method	MLE	MLE(t)	SSM (Song et al., 2020)	DSM (Vincent, 2011)	CoopNet (Xie et al., 2020)	WGAN-0GP (Mescheder et al., 2018)	MEG (Kumar et al., 2019)	VERA (Grathwohl et al., 2021)	DDAEBM (ours)
Test LL	-791	-879	-2039	-4363	-1465	-1214	-1023	-1021	**-902**

likelihood computation and sampling. We get -879 on the MLE test using our network, which is close to the -791 of the traditional NICE model provided by Song et al. (2020). Data is preprocessed following Grathwohl et al. (2021). We observe that using the NICE network leads to worse

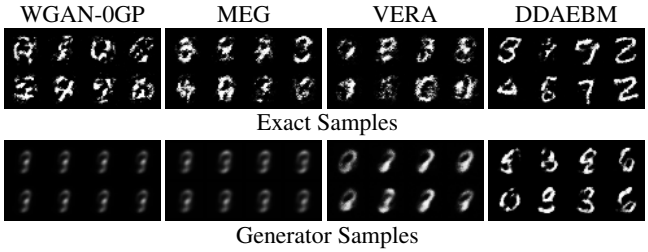


Figure 2. Exact samples from the NICE model and generated samples from the generator for WGAN-based methods and our DDAEBM.

performance of our DDAEBM than other WGAN-based methods such as WGAN-0GP, MEG, and VERA. The reason is that for WGAN-based methods, the gradients of the energy function's output are significantly smaller for fake samples compared to real samples, making the training of energy function similar to MLE. Therefore we add a weight of 0.1 for the second term in Eq. (22) to reduce the effect of fake samples. We use the same trick on WGAN-based methods and find it can also improve their likelihood fitting but help little on generation even using a larger generator. We use an MLP network for our generator. Other experiment settings are the same as VERA (Grathwohl et al., 2021). Full experimental details can be found in the Appendix.

From Table 1 we observe that our method obtains the maximum log-likelihood on test data except MLE which maximizes the log-likelihood of training data as its objective. Fig. 2 shows the exact samples from NICE models and generated samples from the generator. Although all WGAN-based methods yield a good fit of log-likelihood, we can observe that WGAN-0GP and MEG fail to generate diverse and good-quality samples, VERA performs better but still has some mode collapse, our DDAEBM generates high-quality samples that match exact samples and reasonably capture the data distribution.

4.3. Image generation

To confirm the ability of our model to scale to large-scale datasets effectively, we conduct experiments of generation task on 32×32 CIFAR-10 (Krizhevsky et al., 2009), 64×64 CelebA (Liu et al., 2015), and 128×128 LSUN church (Yu et al., 2015) datasets. Each dataset represents a significant increase in complexity. All datasets are scaled to $[-1, 1]$ for pre-processing. For quantitative results, we adopt commonly used Fréchet inception distance (FID) and Inception Score (IS) to evaluate sample fidelity and the number of function evaluations (NFE) to evaluate sampling time. Table 2 shows quantitative results on CIFAR-10. We observe our model achieves FID 4.82 and IS 8.86, outperforming existing adversarial EBMs by a significant margin and performing comparably to strong baselines from other generative models such as GANs and diffusion-based models, which struggle with efficient density estimation. Note that

Table 2. Results for generation on CIFAR-10 dataset

	Model	FID↓	IS↑	NFE↓
	Energy-based models			
MCMC	IGEBM (Du & Mordatch, 2019)	38.2	6.78	60
	JEM (Grathwohl et al., 2020)	38.4	**8.76**	20
	EBM-SR (Nijkamp et al., 2019)	44.50	6.21	100
	EBM-CD (Du et al., 2021)	25.1	7.85	500
	Hat EBM (Hill et al., 2022)	19.30	-	50
	DAMC (Yu et al., 2023)	57.72	-	100
	CoopNets (Xie et al., 2020)	33.61	6.55	50
	VAEBM (Xiao et al., 2021)	12.2	8.43	16
	CLEL-large (Lee et al., 2023)	**8.61**	-	1200
	DRL (Gao et al., 2021)	9.58	8.30	180
No MCMC	EBM Triangle (Han et al., 2019)	28.96	7.30	1
	MEG (Kumar et al., 2019)	35.02	6.49	1
	VERA (Grathwohl et al., 2021)	27.5	-	1
	EBM-BB (Geng et al., 2021)	28.63	7.45	1
	FCE (Gao et al., 2020)	37.30	-	1
	BiDVL (Kan et al., 2022)	20.75	-	1
	DDAEBM(ours)	**4.82**	**8.86**	4
	Other Generative Models			
	SNGAN (Miyato et al., 2018)	21.7	8.22	1
	BigGAN (Brock et al., 2019)	14.73	9.22	1
	StyleGAN2 w/ ADA (Karras et al., 2020a)	2.92	9.83	1
	NCSN-v2 (Song & Ermon, 2020)	10.87	8.40	1000
	DDIM (Song et al., 2021a)	4.67	8.78	50
	Score SDE (Song et al., 2021c)	**2.20**	**9.89**	2000
	DDGAN (Xiao et al., 2022)	3.75	9.63	4
	GLOW (Kingma & Dhariwal, 2018)	48.9	3.92	1

our DDAEBM is still less performant than DDGAN even though they use the same generator structure and parameterization of the generated denoising model. We further add an additional denoising step (Song & Ermon, 2020) using our energy function to refine the generated samples, we find the FID score can be improved to 3.73, which is on par with DDGAN. This denoising step can be approximately viewed as a one-step MCMC using our energy function. With this minor modification, we successfully bridge the gap between DDGAN and DDAEBM in terms of generation performance. Besides, the energy function in DDAEBM is

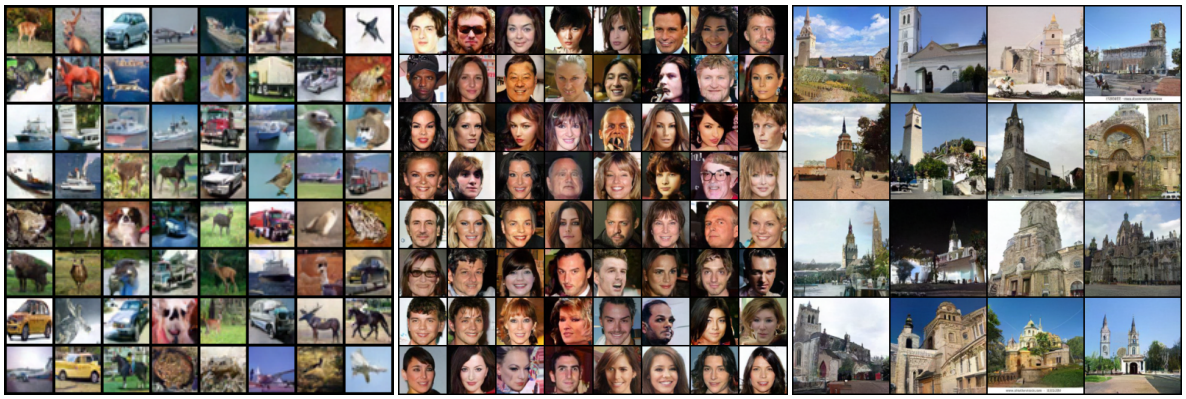


Figure 3. Randomly generated images with DDAEBM on 32×32 CIFAR-10, 64×64 CelebA and 128×128 LSUN church datasets.

tasked with providing a density estimate, not merely functioning as a discriminator. Therefore, the energy function in DDAEBM carries a heavier workload than the discriminator in DDGAN. In contrast, the discriminator in DDGAN merely distinguishes between real and fake sample pairs, primarily concentrating on the quality of the samples rather than on optimizing data likelihood. This explains why the network of our energy function is stronger than the discriminator of DDGAN.

For CelebA, we only report FID scores in Table 3 since the IS score is not widely reported. Our model gets the best performance among adversarial EBMs and outperforms score matching-based model NCSNv2 and VAE-based model NVAE. For LSUN church, Fig. 3 depicts high-fidelity synthesis sampled from the generator. We calculate FID on 50,000 samples using a PyTorch implementation from DDGAN and our model achieves 13.80 for this metric.

Table 3. FID scores on CelebA $64\times$ 64 dataset

Model	FID↓
SNGAN (Miyato et al., 2018)	50.4
COCO-GAN (Lin et al., 2019)	**4.0**
NVAE (Vahdat & Kautz, 2020)	14.74
VAEBM (Xiao et al., 2021)	5.31
NCSNv2 (Song & Ermon, 2020)	26.86
DDPM (Ho et al., 2020)	**3.50**
DRL (Gao et al., 2021)	5.98
joint EBM Triangle (Han et al., 2020)	24.7
BiDVL (Kan et al., 2022)	17.24
DDAEBM (ours)	**10.29**

4.4. Out of distribution detection

The energy function of EBMs can be viewed as an unnormalized density function that assigns low values on out-of-distribution (OOD) regions and high values on data regions, which is suitable for detecting OOD samples. To test this, we first train our DDAEBM on CIFAR-10 and calculate the energy $E_\theta(\mathbf{x}_0, 0)$ on in-distribution images from the CIFAR-10 test set and out-of-distribution images from datasets including SVHN (Netzer et al., 2011), Texture (Cimpoi et al., 2014), CIFAR-100 (Krizhevsky et al., 2009) and CelebA. Following Xiao et al. (2021), we use the area under the ROC curve (AUROC) as a quantitative metric on the energy scores, where high AUROC indicates that the model correctly assigns low density to OOD samples. Results are shown in Table 4, from where we can see our model achieves comparable performance with most of the baselines chosen from recent EBMs, VAEs, and Glow. Our model performs the best on SVHN and Texture datasets while on CIFAR-100 and CelebA it performs at a moderate level. Note that VAEBM performs slightly better than ours on most datasets, but it requires MCMC sampling to refine its generation, which can be inefficient. As is pointed out by Zhang et al. (2021), no method can guarantee better than random chance performance without assumptions on which out-distributions are relevant. Hence we see no clear winner on this task across all the datasets and we should take the results with a grain of salt.

Fig 4 further shows histograms of the energy output at $t = 0$ between the CelebA test dataset and several OOD distributions. For fake data, we diffuse real data $\mathbf{x}_0$ for one step and use $p_\phi(\mathbf{x}_0|\mathbf{x}_1)$ to generate fake data. For noisy data, we add noise to the real data with standard deviations of 0.01, 0.1, and 0.5. From Fig. 4 we observe our energy function assigns high values to real data and low values to fake and noisy data. Additionally, the energy values gradually decrease as the added noise increases. Although fake data and noisy data with 0.01 standard deviation have imperceptible differences for visualization, their energy values still exhibit clear distinctions when compared to real data. This demonstrates our energy function's capability for OOD detection.

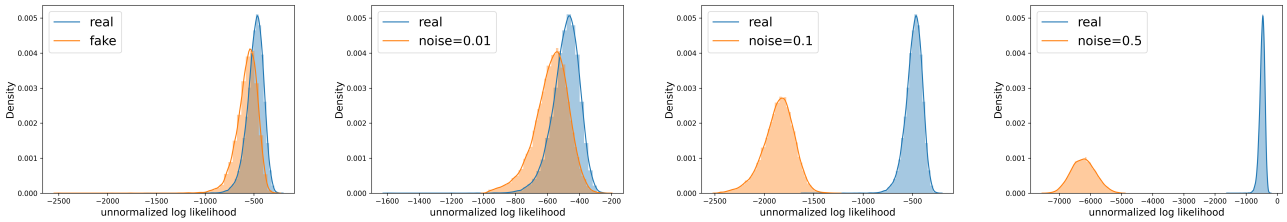


Figure 4. Histogram of unnormalized log-likelihood for comparison of real data and fake data or noisy data with the standard deviations being 0.01, 0.1, and 0.5. We provide the histogram comparison on CelebA test images.

Table 4. AUROC for out-of-distribution detection on SVHN, Texture, CIFAR-100 and CelebA test datasets, with CIFAR-10 as the in-distribution dataset.

Model	SVHN	Texture	CIFAR-100	CelebA
IGEBM (Du & Mordatch, 2019)	0.63	0.48	0.5	0.7
Glow (Kingma & Dhariwal, 2018)	0.05	-	0.55	0.57
NVAE (Vahdat & Kautz, 2020)	0.42	-	0.56	0.68
SVAE (Chen et al., 2018)	0.42	0.5	-	0.52
joint EBM Triangle (Han et al., 2019)	0.68	0.56	-	0.56
VERA (Grathwohl et al., 2021)	**0.83**	-	**0.73**	0.33
BiDVL (Kan et al., 2022)	0.76	-	-	**0.77**
VAEBM (Xiao et al., 2021)	**0.83**	-	0.62	**0.77**
DDAEBM(ours)	**0.83**	**0.62**	0.60	0.70

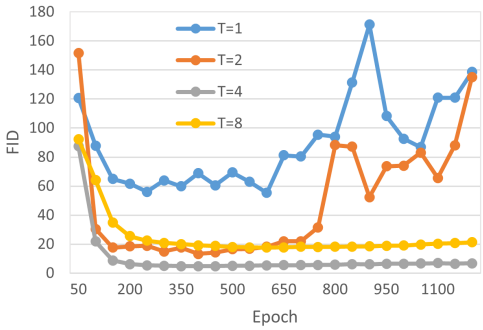


Figure 5. The FID score vs. the number of training epochs for a different number of time steps.

4.5. Ablation studies

Importance of proposed modifications. First, we investigate the effects of some proposed modifications in our model including the latent variable in our defined denoising distribution, introduced posterior $q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)$ and Jeffrey divergence for the generator's training. Table 5 reports FID and IS scores on CIFAR-10 dataset and AUROC for the OOD task with CIFAR-10/SVHN as the in-distribution/out-of-distribution datasets. We replace sampling from $p(\mathbf{z})$ in Eq. (9) with all zero-vectors to remove the effect of $\mathbf{z}$ and observe the FID and IS scores are significantly worse. We also train our model with the most commonly used KL divergence and get similar performance on generation, but OOD performance significantly drops. We further remove $\log q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)$-related term in our objective, which is equivalent to training a sequence of WGAN-0GPs embedded in a diffusion process. The generation and OOD performance are similar to the model trained with KL divergence. This experiment implies that asymmetric KL divergence and lack of $\log q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)$ lead to inaccurate learning of the energy function which, in turn, affects e.g. OOD performance. Symmetric Jeffrey divergence and entropy term $q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)$ ensure a better fit between the generated distribution and the energy function. This improves the energy training, resulting in improved OOD detection. More results can be found in the Appendix.

Number of time steps. We also examine the influence of varying the number of time steps T by plotting the FID score versus training epochs for different time steps in Fig 5. Note that $T = 1$ corresponds to training a standard adversarial EBM, which leads to inadequate performance and unstable training. $T = 2$ can get decent results very early but becomes unstable in later stages. $T = 4$ gives excellent generation results and stable training which is consistent with DDGAN. When T becomes larger, the training can be stable but there is an obvious degradation in performance. We hypothesize that although training for each step can be easier with a larger T, a higher capacity of the energy function is also required to accommodate more time steps. Thus, the choice of T is important. We consistently chose $T = 4$.

Table 5. Ablation studies for some proposed modifications

Model Variants	FID↓	IS↑	OOD (CIFAR-10/SVHN)↑
No latent variable	10.09	8.25	0.23
KL divergence	4.99	**9.11**	0.37
KL w/o $q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)$	4.90	8.90	0.38
DDAEBM	**4.82**	8.86	**0.83**

5. Conclusion

We propose to integrate adversarial EBMs into the denoising diffusion process to learn the complex multimodal data distribution in several steps. This allows us to split a long generated process into several small steps, making each

EBM easier to train. We define the generated denoising distribution by introducing a latent variable $\mathbf{z}$, which greatly accelerates sampling. With this definition, we employ a symmetric Jeffrey divergence to calibrate the training of our generator and introduce a variational posterior distribution $q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)$ to compute the entropy term, these operations address the long-standing training challenges existing in adversarial EBMs. Our model reduces the gap between adversarial EBMs and current mainstream generative models in terms of generation and provides a useful energy function with notable potential for a wide range of applications and downstream tasks.

Acknowledgements

This work was supported by a research grant (42062) from VILLUM FONDEN. This project received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement 757360). The work was partly funded by the Novo Nordisk Foundation through the Center for Basic Machine Learning Research in Life Science (NNF20OC0062606).

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, as all high-capacity generative models carry the risk of being used for misinformation, and our model is no exception, none of which we feel must be specifically highlighted here.

References

Arbel, M., Zhou, L., and Gretton, A. Generalized energy based models. *International Conference on Learning Representations*, 2021.

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pp. 214–223. PMLR, 2017.

Bao, F., Li, C., Zhu, J., and Zhang, B. Analytic-dpm: an analytic estimate of the optimal reverse variance in diffusion probabilistic models. *International Conference on Learning Representations*, 2022.

Brock, A., Donahue, J., and Simonyan, K. Large scale gan training for high fidelity natural image synthesis. *International Conference on Learning Representations*, 2019.

Chen, L., Dai, S., Pu, Y., Zhou, E., Li, C., Su, Q., Chen, C., and Carin, L. Symmetric variational autoencoder and connections to adversarial learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 661–669. PMLR, 2018.

Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., and Vedaldi, A. Describing textures in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3606–3613, 2014.

Cui, J. and Han, T. Learning energy-based model via dual-mcmc teaching. *Advances in Neural Information Processing Systems*, 2023.

Dieng, A. B., Ruiz, F. J., Blei, D. M., and Titsias, M. K. Prescribed generative adversarial networks. *arXiv preprint arXiv:1910.04302*, 2019.

Dinh, L., Krueger, D., and Bengio, Y. Nice: Non-linear independent components estimation. In *International Conference Learning Representations Workshops*, 2015.

Du, Y. and Mordatch, I. Implicit generation and generalization in energy-based models. *Advances in Neural Information Processing Systems*, 2019.

Du, Y., Li, S., Tenenbaum, J., and Mordatch, I. Improved contrastive divergence training of energy based models. *International Conference on Machine Learning*, 2021.

Du, Y., Li, S., Tenenbaum, J., and Mordatch, I. Learning iterative reasoning through energy minimization. In *International Conference on Machine Learning*, pp. 5570–5582. PMLR, 2022.

Du, Y., Durkan, C., Strudel, R., Tenenbaum, J. B., Dieleman, S., Fergus, R., Sohl-Dickstein, J., Doucet, A., and Grathwohl, W. S. Reduce, reuse, recycle: Compositional generation with energy-based diffusion models and mcmc. In *International Conference on Machine Learning*, pp. 8489–8510. PMLR, 2023.

Gao, R., Nijkamp, E., Kingma, D. P., Xu, Z., Dai, A. M., and Wu, Y. N. Flow contrastive estimation of energy-based models. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7518–7528, 2020.

Gao, R., Song, Y., Poole, B., Wu, Y. N., and Kingma, D. P. Learning energy-based models by diffusion recovery likelihood. *International Conference on Learning Representations*, 2021.

Geng, C., Wang, J., Gao, Z., Frellsen, J., and Hauberg, S. Bounds all around: training energy-based models with bidirectional bounds. *Advances in Neural Information Processing Systems*, 34:19808–19821, 2021.

Grathwohl, W., Wang, K.-C., Jacobsen, J.-H., Duvenaud, D., Norouzi, M., and Swersky, K. Your classifier is secretly an energy based model and you should treat it like one.

International Conference on Learning Representations, 2020.

Grathwohl, W. S., Kelly, J. J., Hashemi, M., Norouzi, M., Swersky, K., and Duvenaud, D. No MCMC for me: Amortized sampling for fast and stable training of energy-based models. In *International Conference on Learning Representations*, 2021.

Han, T., Nijkamp, E., Fang, X., Hill, M., Zhu, S.-C., and Wu, Y. N. Divergence triangle for joint training of generator model, energy-based model, and inferential model. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8670–8679, 2019.

Han, T., Nijkamp, E., Zhou, L., Pang, B., Zhu, S.-C., and Wu, Y. N. Joint training of variational auto-encoder and latent energy-based model. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7978–7987, 2020.

Hill, M., Nijkamp, E., Mitchell, J., Pang, B., and Zhu, S.-C. Learning probabilistic models from generator latent spaces with hat ebm. *Advances in Neural Information Processing Systems*, 35:928–940, 2022.

Hinton, G. E. and Sejnowski, T. J. Optimal perceptual inference. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 448, 1983.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020.

Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982. ISSN 0027-8424.

Jeffreys, H. An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 186(1007):453–461, 1946.

Jolicoeur-Martineau, A., Piché-Taillefer, R., Combes, R. T. d., and Mitliagkas, I. Adversarial score matching and improved sampling for image generation. *International Conference on Learning Representations*, 2021.

Kan, G., Lü, J., Wang, T., Zhang, B., Zhu, A., Huang, L., Guo, G., and Snoussi, H. Bi-level doubly variational learning for energy-based latent variable models. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 18460–18469, 2022.

Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J., and Aila, T. Training generative adversarial networks with limited data. *Advances in Neural Information Processing Systems*, 33:12104–12114, 2020a.

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T. Analyzing and improving the image quality of stylegan. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8110–8119, 2020b.

Kingma, D., Salimans, T., Poole, B., and Ho, J. Variational diffusion models. *Advances in Neural Information Processing Systems*, 34:21696–21707, 2021.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. *Advances in Neural Information Processing Systems*, 31, 2018.

Kong, F., Duan, J., Sun, L., Cheng, H., Xu, R., Shen, H., Zhu, X., Shi, X., and Xu, K. Act: Adversarial consistency models. *arXiv preprint arXiv:2311.14097*, 2023.

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

Kumar, R., Ozair, S., Goyal, A., Courville, A., and Bengio, Y. Maximum entropy generators for energy-based models. *arXiv preprint arXiv:1901.08508*, 2019.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *International Conference on Machine Learning*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Lee, H., Jeong, J., Park, S., and Shin, J. Guiding energy-based models via contrastive latent variables. *International Conference on Learning Representations*, 2023.

Lin, C. H., Chang, C.-C., Chen, Y.-S., Juan, D.-C., Wei, W., and Chen, H.-T. Coco-gan: Generation by parts via conditional coordinating. In *International Conference on Computer Vision*, pp. 4512–4521, 2019.

Lin, Z., Khetan, A., Fanti, G., and Oh, S. Pacgan: The power of two samples in generative adversarial networks. *IEEE Journal on Selected Areas in Information Theory*, pp. 324–335, May 2020.

Liu, R., Gao, J., Zhang, J., Meng, D., and Lin, Z. Investigating bi-level optimization for learning and vision from a unified perspective: A survey and beyond. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, pp. 10045–10067, 2022.

Liu, W., Wang, X., Owens, J., and Li, Y. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 33:21464–21475, 2020.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *International Conference on Computer Vision*, pp. 3730–3738, 2015.

Lu, C., Zheng, K., Bao, F., Chen, J., Li, C., and Zhu, J. Maximum likelihood training for score-based diffusion

odes by high order denoising score matching. In *International Conference on Machine Learning*, pp. 14429–14460. PMLR, 2022a.

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022b.

Mescheder, L., Geiger, A., and Nowozin, S. Which training methods for gans do actually converge? In *International Conference on Machine Learning*, pp. 3481–3490. PMLR, 2018.

Misra, N., Singh, H., and Demchuk, E. Estimation of the entropy of a multivariate normal distribution. *Journal of multivariate analysis*, 92(2):324–342, 2005.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. *International Conference on Learning Representations*, 2018.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. 2011.

Nijkamp, E., Hill, M., Zhu, S.-C., and Wu, Y. N. Learning non-convergent non-persistent short-run mcmc toward energy-based model. *Advances in Neural Information Processing Systems*, 32, 2019.

Orlitsky, A. Information theory. In Meyers, R. A. (ed.), *Encyclopedia of Physical Science and Technology (Third Edition)*, pp. 751–769. Academic Press, New York, third edition edition, 2003.

Pang, B., Han, T., Nijkamp, E., Zhu, S.-C., and Wu, Y. N. Learning latent space energy-based prior model. *Advances in Neural Information Processing Systems*, 33: 21994–22008, 2020.

Smolensky, P. *Information Processing in Dynamical Systems: Foundations of Harmony Theory*, pp. 194–281. MIT Press, Cambridge, MA, USA, 1986.

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

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. *International Conference on Learning Representations*, 2021a.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

Song, Y. and Ermon, S. Improved techniques for training score-based generative models. *Advances in Neural Information Processing Systems*, 33:12438–12448, 2020.

Song, Y., Garg, S., Shi, J., and Ermon, S. Sliced score matching: A scalable approach to density and score estimation. In *Uncertainty in Artificial Intelligence*, pp. 574–584. PMLR, 2020.

Song, Y., Durkan, C., Murray, I., and Ermon, S. Maximum likelihood training of score-based diffusion models. *Advances in Neural Information Processing Systems*, 34: 1415–1428, 2021b.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. *International Conference on Learning Representations*, 2021c.

Srivastava, A., Valkov, L., Russell, C., Gutmann, M., and Sutton, C. Veegan: Reducing mode collapse in gans using implicit variational learning. *Advances in Neural Information Processing Systems*, Dec 2017.

Vahdat, A. and Kautz, J. Nvae: A deep hierarchical variational autoencoder. *Advances in Neural Information Processing Systems*, 33:19667–19679, 2020.

Vincent, P. A connection between score matching and denoising autoencoders. *Neural Computation*, pp. 1661–1674, Jul 2011.

Xiao, Z., Kreis, K., Kautz, J., and Vahdat, A. Vaebm: A symbiosis between variational autoencoders and energy-based models. *International Conference on Learning Representations*, 2021.

Xiao, Z., Kreis, K., and Vahdat, A. Tackling the generative learning trilemma with denoising diffusion gans. *International Conference on Learning Representations*, 2022.

Xie, J., Lu, Y., Gao, R., Zhu, S.-C., and Wu, Y. N. Cooperative training of descriptor and generator networks. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, pp. 27–45, Jan 2020.

Xie, J., Zheng, Z., and Li, P. Learning energy-based model with variational auto-encoder as amortized sampler. In *Association for the Advancement of Artificial Intelligence*, volume 35, pp. 10441–10451, 2021.

Xie, J., Zhu, Y., Li, J., and Li, P. A tale of two flows: Cooperative learning of langevin flow and normalizing flow toward energy-based model. *International Conference on Learning Representations*, 2022.

Yu, F., Seff, A., Zhang, Y., Song, S., Funkhouser, T., and Xiao, J. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

Yu, P., Zhu, Y., Xie, S., Ma, X., Gao, R., Zhu, S.-C., and Wu, Y. N. Learning energy-based prior model with diffusion-amortized mcmc. *Advances in Neural Information Processing Systems*, 2023.

Zhai, S., Cheng, Y., Feris, R., and Zhang, Z. Generative adversarial networks as variational training of energy based models. *arXiv preprint arXiv:1611.01799*, 2016.

Zhang, J., Xie, J., Zheng, Z., and Barnes, N. Energy-based generative cooperative saliency prediction. In *Association for the Advancement of Artificial Intelligence*, volume 36, pp. 3280–3290, 2022.

Zhang, L., Goldstein, M., and Ranganath, R. Understanding failures in out-of-distribution detection with deep generative models. In *International Conference on Machine Learning*, pp. 12427–12436. PMLR, 2021.

Zhao, J., Mathieu, M., and LeCun, Y. Energy-based generative adversarial network. *International Conference on Learning Representations*, 2017.

Zhu, Y., Xie, J., Wu, Y., and Gao, R. Learning energy-based models by cooperative diffusion recovery likelihood. *International Conference on Learning Representations*, 2024.

A. Extended derivations

A.1. Derivations of Eq. (7)

$$\begin{aligned} D_{\mathrm{KL}}\left(q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right) \| p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)\right) &= \int q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right) \log \frac{q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)}{p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)} d\mathbf{x}_{t-1} \\ &= \mathbb{E}_{q(\mathbf{x}_{t-1}|\mathbf{x}_t)} \log q(\mathbf{x}_{t-1}|\mathbf{x}_t) - \mathbb{E}_{q(\mathbf{x}_{t-1}|\mathbf{x}_t)} \log p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \end{aligned} \tag{25}$$

Thus the objective Eq. (5) can be written as:

$$\mathcal{L} = -\sum_{t\geq 1} \left[\mathbb{E}_{q(\mathbf{x}_{t-1},\mathbf{x}_t)} \log q(\mathbf{x}_{t-1}|\mathbf{x}_t) - \mathbb{E}_{q(\mathbf{x}_{t-1},\mathbf{x}_t)} \log p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)\right] + C \tag{26}$$

Since the first and the third term are independent of θ, they can be disregarded during optimization. Thus the objective is only the second term. Plugging Eq. (6) into the second term, we can obtain:

$$\mathcal{L} = \sum_{t\geq 1} \mathbb{E}_{q(\mathbf{x}_{t-1},\mathbf{x}_t)} E_\theta(\mathbf{x}_{t-1}, t-1) + \log q(\mathbf{x}_t|\mathbf{x}_{t-1}) - \log \tilde{Z}_{\theta,t}(\mathbf{x}_t) \tag{27}$$

A.2. Derivations of the optimum in Eq. (7)

When Eq. (7) gets optimum, due to the equivalence of Eq. (5) and Eq. (7), we can easily obtain $D_{\mathrm{KL}}\left(q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right) \| p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)\right) = 0$ for each time step, i.e., $q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right) = p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)$, which means:

$$\begin{aligned} &E_\theta(\mathbf{x}_{t-1}, t-1) + \log q(\mathbf{x}_t|\mathbf{x}_{t-1}) - \log \tilde{Z}_{\theta,t}(\mathbf{x}_t) = \\ &\log q(\mathbf{x}_{t-1}) + \log q(\mathbf{x}_t|\mathbf{x}_{t-1}) - \log q(\mathbf{x}_t) \end{aligned} \tag{28}$$

Then we can obtain

$$E_\theta(\mathbf{x}_{t-1}, t-1) - \log q(\mathbf{x}_{t-1}) = \log \tilde{Z}_{\theta,t}(\mathbf{x}_t) - \log q(\mathbf{x}_t) \tag{29}$$

The left side of the Eq. (29) is just related to $\mathbf{x}_{t-1}$, while the right side is just related to $\mathbf{x}_t$, as $\mathbf{x}_{t-1}$ can respond to all $\mathbf{x}_t$ in the whole space of $\mathcal{X}$, then we can get $E_\theta(\mathbf{x}_{t-1}, t-1) - \log q(\mathbf{x}_{t-1}) = C$, therefore, for $\forall t$, we have

$$\frac{\exp\left(E_\theta(\mathbf{x}_t, t)\right)}{Z_{\theta,t}} = q(\mathbf{x}_t), Z_{\theta,t} = \int \exp\left(E_\theta(\mathbf{x}_t, t)\right) d\mathbf{x}_t \tag{30}$$

We already have $p_\theta(\mathbf{x}_T) = q(\mathbf{x}_T)$ at $t = T$, when Eq. (5) gets optimum,

$$\begin{aligned} p_\theta(\mathbf{x}_{0:T}) &= q\left(\mathbf{x}_T\right) \prod_{t=1}^{T} p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right) \\ &= q\left(\mathbf{x}_T\right) \prod_{t=1}^{T} q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right) = q(\mathbf{x}_{0:T}) \end{aligned} \tag{31}$$

Therefore, we have $p_\theta(\mathbf{x}_t) = q(\mathbf{x}_t) = \frac{\exp\left(E_\theta(\mathbf{x}_t, t)\right)}{\boldsymbol{Z}_{\theta,t}}$.

A.3. Derivations of Eq. (8)

$$\frac{\partial \mathcal{L}}{\partial \theta} = \sum_{t\geq 1} \mathbb{E}_{q(\mathbf{x}_{t-1},\mathbf{x}_t)} \left[\frac{\partial \left(E_\theta(\mathbf{x}_{t-1}, t-1) - \log \tilde{Z}_{\theta,t}(\mathbf{x}_t)\right)}{\partial \theta}\right] \tag{32}$$

where

$$\begin{aligned} &\frac{\partial \log \tilde{Z}_{\theta,t}(\mathbf{x}_t)}{\partial \theta} = \frac{1}{\tilde{Z}_{\theta,t}} \frac{\partial \tilde{Z}_{\theta,t}}{\partial \theta} = \frac{1}{\tilde{Z}_{\theta,t}} \int \frac{\exp(E_\theta(\mathbf{x}_{t-1}, t-1)) q(\mathbf{x}_t|\mathbf{x}_{t-1})}{\partial \theta} d\mathbf{x}_{t-1} \\ &= \int \frac{\exp(E_\theta(\mathbf{x}_{t-1}, t-1)) q(\mathbf{x}_t|\mathbf{x}_{t-1})}{\tilde{Z}_{\theta,t}} \frac{\partial E_\theta(\mathbf{x}_{t-1}, t-1)}{\partial \theta} d\mathbf{x}_{t-1} \\ &= \int p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right) \frac{\partial E_\theta(\mathbf{x}_{t-1}, t-1)}{\partial \theta} d\mathbf{x}_{t-1} \end{aligned} \tag{33}$$

Therefore, we can obtain:

$$\frac{\partial \mathcal{L}}{\partial \theta} = \sum_{t \geq 1} \mathbb{E}_{q(\mathbf{x}_{t-1}, \mathbf{x}_t)} \frac{\partial E_\theta(\mathbf{x}_{t-1}, t-1)}{\partial \theta} - \mathbb{E}_{q(\mathbf{x}_t) p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} \frac{\partial E_\theta(\mathbf{x}_{t-1}, t-1)}{\partial \theta} \tag{34}$$

A.4. Derivations of Eq. (13)

$$D_{\mathrm{KL}}(p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t) \| p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)) = \mathbb{E}_{p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)} \log p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t) - \mathbb{E}_{p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)} \log p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \tag{35}$$

Thus

$$\begin{aligned}
&\sum_{t \geq 1} \mathbb{E}_{q(\mathbf{x}_t)} D_{\mathrm{KL}}(p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t) \| p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)) \\
&= \sum_{t \geq 1} \mathbb{E}_{q(\mathbf{x}_t) p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)} \log p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t) - \mathbb{E}_{q(\mathbf{x}_t) p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)} \log p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \\
&= \sum_{t \geq 1} -H[p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)] - \mathbb{E}_{q(\mathbf{x}_t) p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)} \log p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)
\end{aligned} \tag{36}$$

A.5. Derivations of the entropy

Since

$$p_\phi(\mathbf{x}_{t-1}, \mathbf{z}|\mathbf{x}_t) = p(\mathbf{z}) q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0 = G_\phi\left(\mathbf{x}_t, \mathbf{z}, t\right)\right), \tag{37}$$

according to the property of entropy (Orlitsky, 2003)

$$H[\mathbf{x}, \mathbf{z}] = H[\mathbf{z}] + H[\mathbf{x}|\mathbf{z}], \tag{38}$$

we can get

$$H[p_\phi(\mathbf{x}_{t-1}, \mathbf{z}|\mathbf{x}_t)] = H[\mathbf{z}] + H[q(\mathbf{x}_{t-1}|\mathbf{x}_t, G_\phi\left(\mathbf{x}_t, \mathbf{z}, t\right))] \tag{39}$$

Since $p(\mathbf{z})$ and $q(\mathbf{x}_{t-1}|\mathbf{x}_t, G_\phi\left(\mathbf{x}_t, \mathbf{z}, t\right))$ are both Gaussian distributions, according to (Misra et al., 2005), a m-dimensional Gaussian distribution $p(\mathbf{x})$ with mean μ and a $d \times d$ positive definite covariance matrix Σ, its entropy is

$$H\left[p(\mathbf{x})\right] = \frac{m}{2}[1 + \ln(2\pi)] + \frac{\ln|\Sigma|}{2}. \tag{40}$$

Therefore, the entropy of $p(\mathbf{z})$ and $q(\mathbf{x}_{t-1}|\mathbf{x}_t, G_\phi\left(\mathbf{x}_t, \mathbf{z}, t\right))$ can be computed directly:

$$H\left[p(\mathbf{z})\right] = \frac{d}{2}(1 + \log(2\pi)) \tag{41}$$

$$H[q(\mathbf{x}_{t-1}|\mathbf{x}_t, G_\phi\left(\mathbf{x}_t, \mathbf{z}, t\right))] = \frac{D}{2}(1 + \log(2\pi)) + \frac{D}{2} \log \tilde{\beta}_t \tag{42}$$

A.6. Derivations of Eq. (16)

$$\begin{aligned}
&H[p_\phi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)] = -\mathbb{E}_{q(\mathbf{x}_t) p_\phi(\mathbf{x}_{t-1}, \mathbf{z}|\mathbf{x}_t)} \log p_\phi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t) \\
&= -\mathbb{E}_{q(\mathbf{x}_t) p_\phi(\mathbf{x}_{t-1}, \mathbf{z}|\mathbf{x}_t)} \log \frac{p_\phi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)}{q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)} - \mathbb{E}_{q(\mathbf{x}_t) p_\phi(\mathbf{x}_{t-1}, \mathbf{z}|\mathbf{x}_t)} \log q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t) \\
&= -\mathbb{E}_{q(\mathbf{x}_t) p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)} D_{\mathrm{KL}}(p_\phi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t) \| q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)) - \mathbb{E}_{q(\mathbf{x}_t) p_\phi(\mathbf{x}_{t-1}, \mathbf{z}|\mathbf{x}_t)} \log q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t) \\
&\leq -\mathbb{E}_{q(\mathbf{x}_t) p_\phi(\mathbf{x}_{t-1}, \mathbf{z}|\mathbf{x}_t)} \log q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)
\end{aligned} \tag{43}$$

A.7. Derivations of Eq. (18)

$$
\begin{aligned}
\log p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t) &= \log \int p(\mathbf{z}) q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0 = G_\phi\left(\mathbf{x}_t, \mathbf{z}, t\right)\right) \mathrm{d}\mathbf{z} \\
&= \log \int q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t) \frac{p(\mathbf{z})}{q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)} q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0 = G_\phi\left(\mathbf{x}_t, \mathbf{z}, t\right)\right) \mathrm{d}\mathbf{z} \\
&\geq \int q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t) \left[\log \frac{p(\mathbf{z})}{q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)} + \log q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0 = G_\phi\left(\mathbf{x}_t, \mathbf{z}, t\right)\right)\right] \mathrm{d}\mathbf{z} \\
&= -D_{KL}\left(q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t) \| p(\mathbf{z})\right) + \mathbb{E}_{q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)} \log q(\mathbf{x}_{t-1}|\mathbf{x}_t, G_\phi\left(\mathbf{x}_t, \mathbf{z}, t\right))
\end{aligned} \tag{44}
$$

A.8. Derivations of Eq. (19)

$$
\begin{aligned}
&\sum_{t\geq 1} \mathbb{E}_{q(\mathbf{x}_t)} D_{\mathrm{KL}}(p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \| p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)) \\
=&\sum_{t\geq 1} \mathbb{E}_{q(\mathbf{x}_t) p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} \log p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) - \mathbb{E}_{q(\mathbf{x}_t) p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} \log p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t)
\end{aligned} \tag{45}
$$

The first term can be disregarded since it's independent of ϕ, thus the variational upper bound of Eq. (12) can be written as follows by applying Eq. (18) to the above equation:

$$
\begin{aligned}
-\sum_{t\geq 1} \mathbb{E}_{q(\mathbf{x}_t) p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} \log p_\phi(\mathbf{x}_{t-1}|\mathbf{x}_t) \leq &\sum_{t\geq 1} \mathbb{E}_{q(\mathbf{x}_t) p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} D_{KL}\left(q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t) \| p(\mathbf{z})\right) \\
&- \mathbb{E}_{q(\mathbf{x}_t) p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)} \log q(\mathbf{x}_{t-1}|\mathbf{x}_t, G_\phi\left(\mathbf{x}_t, \mathbf{z}, t\right))
\end{aligned} \tag{46}
$$

B. Experimental details

B.1. Diffusion process

We use the discretization of the continuous-time VP SDE (Song et al., 2021c) as our diffusion process, which is the same as DDGAN. For all datasets, we set the number of time steps T to be 4. The variance function of VP SDE is given by:

$$
\sigma^2\left(t'(t)\right) = 1 - e^{-\beta_{\min} t'(t) - 0.5(\beta_{\max} - \beta_{\min}) t'^2(t)}, \tag{47}
$$

where $t'(t) \in [0, 1]$, $t'(t)$ is a function of time step t denoting the time of $\mathbf{x}_t$ in VP SDE. $t'(t)$ can be any flexible time schedule of a forward SDE. The constants $\beta_{\max}$ and $\beta_{\min}$ are chosen differently for different datasets. β_t is designed as follows:

$$
\beta_t = 1 - \frac{1 - \sigma^2\left(t'(t)\right)}{1 - \sigma^2\left(t'(t-1)\right)} = 1 - e^{-\beta_{\min}\left(t'(t) - t'(t-1)\right) - 0.5(\beta_{\max} - \beta_{\min})\left(t'^2(t) - t'^2(t-1)\right)}, \tag{48}
$$

where $t \in \{0, 1, 2, \ldots T\}$. Except for LSUN church dataset, we use equidistant steps in time for $t'(t)$ on other datasets, i.e. $t'(t) = \frac{t}{T}$. For LSUN church, we borrow the time schedule from DRL, which focuses on the first stage of the diffusion process: $t'(t) = \begin{cases} \frac{t\lfloor \frac{500}{T-1} \rfloor}{999} & t \leq T \\ 1 & t = T \end{cases}$, we assume $T \leq 999$ in most cases. We choose this time schedule because signal-to-noise ratio (SNR) (Kingma et al., 2021) is strictly monotonically decreasing in time, in high-dimensional space, data becomes quite sparse, thus training more on the first stage may be more important for generation.

B.2. Network structure

B.2.1. Toy data

For toy datasets, our encoder, generator, and energy function all have two embedding networks to encode the toy data, latent variable, and time t into two features. Then a decoder is used to decode the concatenation of these features into our desired output. Network structures are illustrated in Table 6. We use the Adam optimizer with a learning rate of 10^{-4} for all the networks. The batch size is 200, and we train the model for 180k iterations. For FCE, we also choose Glow as the

Table 6. Network structures for toy datasets. BN denotes batch normalization. SE denotes sinusoidal embedding. $\mathbf{x}_{out}, t_{out}$ denote the outputs of two embedding networks with $\mathbf{x}_t$-related input and sinusoidal embedding of t as the inputs.

Energy	
$\mathbf{x}_t$	$SE(t)$
FC 16 PReLU FC 32	FC 16 PReLU FC 32
concat $[\mathbf{x}_{out}, t_{out}]$	
FC 300 PReLU	
FC 300 PReLU	
FC 1	

Generator	
concat $[\mathbf{x}_t, \mathbf{z}]$	$SE(t)$
FC 16 PReLU BN FC 32	FC 16 PReLU FC 32
concat $[\mathbf{x}_{out}, t_{out}]$	
FC 300 PReLU BN	
FC 300 PReLU BN	
FC 2	

Encoder	
concat $[\mathbf{x}_{t-1}, \mathbf{x}_t]$	$SE(t)$
FC 16 PReLU BN FC 32	FC 16 PReLU FC 32
concat $[\mathbf{x}_{out}, t_{out}]$	
FC 300 PReLU BN	
FC 300 PReLU BN	
FC 16	
mean: FC 2	logvar: FC 2

Table 7. Network structures for MNIST dataset. $permutate(\cdot)$ denotes the permutation operation in Glow.

Energy	
$permutate(\mathbf{x}_t)$	$SE(t)$
4 NICE layers	
NICE scale layer	

NICE layer	
FC 1000 PReLU + FC 1000 PReLU ($SE(t)$)	$\times 4$
FC 392	

Generator	
concat $[\mathbf{x}_t, \mathbf{z}]$	$SE(t)$
FC 1600 PReLU FC 1600	FC 50 PReLU FC 100
concat $[\mathbf{x}_{out}, t_{out}]$	
FC 3200 PReLU BN	
FC 1600 PReLU BN	
FC 784	

Encoder	
concat $[\mathbf{x}_{t-1}, \mathbf{x}_t]$	$SE(t)$
FC 3200 PReLU FC 1600	FC 50 PReLU FC 100
concat $[\mathbf{x}_{out}, t_{out}]$	
FC 1600 PReLU BN	
FC 1600 PReLU BN	
mean: FC 50	logvar: FC 50

generator, where we use 5 affine coupling layers amounting to 15 fully connected layers with 300 hidden units. For other adversarial EBMs, both the generator and energy function have 3 fully connected layers each with 300 hidden units and PReLU activations. Batch normalization is used in generators. The latent dimension is set to 2.

B.2.2. MNIST

For our NICE_t network, similar to NICE model in VERA (Grathwohl et al., 2021), we also have 4 coupling layers and each coupling layer has 5 hidden layers with 1000 units and PReLU nonlinearity. We integrate parameter t into NICE by adding a linear layer with the time embeddings as the inputs and PReLU nonlinearity to the output of each hidden layer. For other WGAN-based methods, we use the same NICE model as in VERA. We deepen their generators, which allows for a fairer comparison of our method. Their generators all have a latent dimension of 100 and 5 hidden layers with 1600 or 3200 units each. We adopt PReLU nonlinearity and batch normalization in their generators as is common with generator networks. Our generator and encoder employ the same structure as those for toy datasets except for more hidden units. The latent dimension is set to 50. Network structures are shown in Table 7. All models were trained for 400 epochs with the Adam optimizer. We use the learning rate 10^{-4} for all our networks. For other WGAN-based methods, we use learning rate 3×10^{-6} for energy function and 3×10^{-4} for generator. We use a batch size of 128 for all models.

B.2.3. LARGE-SCALE DATASETS

For large-scale datasets, we use sinusoidal positional embeddings for conditioning on integer time steps. For CIFAR-10 and CelebA datasets, our generator follows the modified Unet structure in DDGAN (Xiao et al., 2022), which provides z-conditioning to the NCSN++ architecture (Song et al., 2021c). Our energy function mostly follows NCSN++ in Score SDE, which also takes $\mathbf{x}_t$ and t as its inputs. The only difference is we remove the last scale-by-sigma operation in NCSN++ and replace it with a negative ℓ_2 norm between input $\boldsymbol{x}_t$ and the output of the NCSN++ network as in Eq. (24). Our encoder is designed as a CNN network that incorporates time embeddings and concatenates $\mathbf{x}_{t-1}$ and $\mathbf{x}_t$. The Encoder network is shown in Table 8. For LSUN dataset, we simply removed FIR upsampling/downsampling and progressive growing architecture from the z-conditioned NCSN++ and named it DDPM++ as our generator, following the naming convention of Score SDE. For energy function, we adopt DDPM++ in Score SDE except for the same modifications used in NCSN++. We find that DDPM++ helps generation on LSUN dataset. Our encoder borrows the network structure of

the discriminator in DDGAN, but replaces the final dense layer with two dense layers that separately output the mean and variance of $q_\psi(\mathbf{z}|\mathbf{x}_{t-1}, \mathbf{x}_t)$. See DDGAN for more details.

Table 8. Encoder structures for CIFAR-10 and CelebA datasets.

Encoder of CIFAR-10	
concat $[\mathbf{x}_{t-1}, \mathbf{x}_t]$	$SE(t)$
	FC 256 LeakyReLU (0.2)
	FC 256 LeakyReLU (0.2)
3×3 Conv2D 64, LeakyReLU (0.2)	
+ Dense 64 (t_{out})	
4×4 Conv2D 128, LeakyReLU (0.2)	
+Dense 128(t_{out})	
4×4 Conv2D 256, LeakyReLU (0.2)	
+Dense 256 (t_{out})	
4*4 Conv2D 512, LeakyReLU (0.2)	
+Dense 512 (t_{out})	
mean: 4×4 Conv2D 100	logvar: 4×4 Conv2D 100

Encoder of CelebA	
concat $[\mathbf{x}_{t-1}, \mathbf{x}_t]$	$SE(t)$
	FC 256 LeakyReLU (0.2)
	FC 256 LeakyReLU (0.2)
3×3 Conv2D 64, LeakyReLU (0.2)	
+ Dense 64 (t_{out})	
4×4 Conv2D 128, LeakyReLU (0.2)	
+Dense 128(t_{out})	
4×4 Conv2D 256, LeakyReLU (0.2)	
+Dense 256 (t_{out})	
4*4 Conv2D 512, LeakyReLU (0.2)	
+Dense 512 (t_{out})	
4×4 Conv2D 1024, LeakyReLU (0.2)	
+Dense 1024 (t_{out})	
mean: 4×4 Conv2D 100	logvar: 4×4 Conv2D 100

B.3. Hyperparameter settings

We specify the hyperparameters used for our generators and training optimization on each dataset in Table 9 and Table 10.

Table 9. Hyper-parameters for our generator network

	CIFAR-10	CelebA	LSUN church
# of ResNet blocks per scale	2	2	2
Initial # of channels	128	64	64
Channel multiplier for each scale	(1,2,2,2)	(1,1,2,2,4)	(1,2,2,4,4)
Scale of attention block	16	16	16
Latent Dimension	100	100	100
# of latent mapping layers	4	4	4
Latent embedding dimension	256	256	256

B.4. Evaluation

When evaluating FID and IS scores, we use 50k generated samples for CIFAR-10, CelebA, and LSUN church datasets. We use Pytorch 1.10.0 and CUDA 11.3 for training. Our training converges approximately 3 times faster than DDGAN, resulting in a comparable overall training time despite our model's slower per-epoch training speed compared to DDGAN.

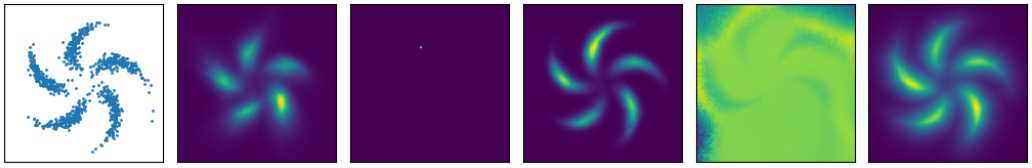
C. Additional results

C.1. More comparisons on Toy dataset

Fig 6 further demonstrates our model's strong capability on density estimation by comparing with score flow (Song et al., 2021b), DRL and DDGAN. Score flow and DRL specialize in density estimation, especially for large-scale datasets, but they can't perform well on toy datasets. We choose sub-VP SDE for the diffusion process and replace the score network with the gradient of an energy function, the network of which is the same as ours except for changing positional embeddings to random Fourier feature embeddings for conditioning on continuous time steps. DRL requires multiplying its energy function by a scaling factor of 0.01, which introduces a temperature parameter between the trained energy function and the target one. DDGAN combines the diffusion process with GAN instead of EBM, as we expected, DDGAN is not suitable as a density estimator because GAN-based models are not supposed to provide a density estimate.

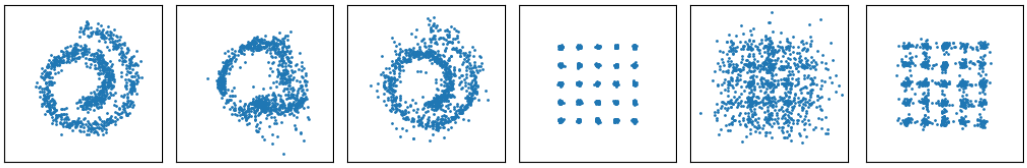
Table 10. Hyper-parameters for our training optimization

	MNIST	CIFAR-10	CelebA	LSUN church
Initial learning rate	1e-4	1e-4	5e-5	5e-5
β_{min}, β_{max} in Eq. (47)	(0.1, 10)	(0.1, 20)	(0.1, 20)	(0.1, 20)
w, w_{mid} in Eq. (21)	(1, 1)	(1, 1)	(1, 1)	(0.6, 0.2)
Adam optimizer β_1, β_2	(0.0, 0.9)	(0.0, 0.9)	(0.0, 0.9)	(0.0, 0.9)
EMA	None	0.9999	0.999	0.999
Batch size	128	64	32	12
# of training epochs	400	1200	400	400
# of GPUs	1	4	4	4



real data | Score flow | DRL w/o scale | DRL w/ scale 0.01 | DDGAN | DDAEBM

Figure 6. Density estimation on pinwheel dataset for different methods.



Swiss roll | DDGAN | DDAEBM | Swiss roll | DDAEBM (KL) | DDAEBM (Jeffrey)

Figure 7. More generation results on Swiss roll and 25-Gaussians datasets

We also show more generation results in Fig. 7 to compare our model with DDGAN and DDAEBM trained with KL divergence. Although DDGAN can achieve impressive sample quality on the 25-Gaussians dataset, as demonstrated in its initial paper, it gets limited quality on the Swiss roll dataset, which means DDGAN is not stable and robust enough across various datasets. Our DDAEBM can get much better generation performance on Swiss roll dataset, verifying our model's advantage in terms of robustness. We can also observe that without the reversed KL divergence, DDAEBM fails to converge to each mode, leading to poor sample quality. Our DDAEBM trained with symmetric Jeffrey divergence can improve generation a lot, resulting in satisfactory sample quality on each mode. This experiment demonstrates besides the training of energy function as shown in Table 5, training with Jeffrey divergence also has superiority to the generator's training by adding an extra reversed KL divergence.

Table 11. Mode coverage on StackedMNIST

Model	Modes	KL
VEEGAN (Srivastava et al., 2017)	762	2.173
PresGAN (Dieng et al., 2019)	**1000**	0.115
StyleGAN2 (Karras et al., 2020b)	940	0.424
Adv.DSM (Jolicoeur-Martineau et al., 2021)	**1000**	1.49
VAEBM (Xiao et al., 2021)	**1000**	0.087
DDGAN (Xiao et al., 2022)	**1000**	0.071
MEG (Kumar et al., 2019)	**1000**	0.042
EBM-BB (Geng et al., 2021)	**1000**	0.045
DDAEBM(ours)	**1000**	**0.033**

C.2. Mode counting

Generative Adversarial Networks (GANs) are notorious for mode collapse which is a phenomenon in which the generator function maps all samples to a small subset of the observation space. It's well known that EBM can alleviate mode collapse because of its entropy term. Therefore, we also evaluate the mode coverage of our model on the StackedMNIST dataset. StackedMNIST is a synthetic dataset that contains images generated by randomly choosing three MNIST images and stacking them along the RGB channels. Hence the true total number of modes is 1,000, and they are counted using a pretrained MNIST classifier. Similar to DDGAN, we also follow the setting of (Lin et al., 2020) and report the number of covered modes and the KL divergence from the categorical distribution over 1000 categories of generated samples to true data in Table 11. It shows that our model covers all modes and achieves the lowest KL compared to GAN-based models and EBMs. This demonstrates our model is effective in mitigating mode collapse.

C.3. Additional results on LSUN church

We provide more results on 128×128 LSUN church dataset. Since there are few baselines on this resolution, we just compare our model with DRL (Gao et al., 2021), which is superior to the majority of EBMs on various image datasets. Our DDAEBM can get better visual quality compared to DRL as illustrated in Fig. 8 and Fig. 9. We calculate FID on 50,000 samples using a PyTorch implementation from DDGAN. Our DDAEBM has 13.80 FID. Original paper of DRL reports 9.76 FID with TensorFlow for calculation. But we used officially provided checkpoints and evaluated FID using our PyTorch implementation, we only got 26.69 FID which is much worse than ours.

Figure 8. Generated samples using DDAEBM on LSUN church dataset. FID=13.80

Figure 9. Generated samples using DRL on LSUN church dataset. FID=26.69