Bounds all around: training energy-based models with bidirectional bounds

Anonymous Author(s)
Affiliation
Address
email

Abstract

Energy-based models (EBMs) provide an elegant framework for density estimation, but they are notoriously difficult to train. Recent work has established links to generative adversarial networks, where the EBM is trained through a minimax game with a variational value function. We propose a bidirectional bound on the EBM log-likelihood, such that we maximize a lower bound and minimize an upper bound when solving the minimax game. We link one bound to a gradient penalty that stabilize training, thereby provide grounding for best engineering practice. To evaluate the bounds we develop a new and efficient estimator of the Jacobi-determinant of the EBM generator. We demonstrate that these developments significantly stabilize training and yield high-quality density estimation and sample generation.

1 Energy-based models

Energy-based models (EMBs) are probabilistic models that draw inspiration from physics and have a long history in machine learning (Hopfield, 1982; Hinton and Sejnowski, 1983; Smolensky, 1986). An EBM is specified in terms of an energy function $E_\theta : \mathcal{X} \to \mathbb{R}$ that is parameterized by θ and defines a probability distribution over $\mathcal{X}$ from the Gibbs density:

$$p_\theta(\mathbf{x}) = \frac{\exp(-E_\theta(\mathbf{x}))}{Z_\theta}, \qquad Z_\theta = \int \exp(-E_\theta(\mathbf{x}))\mathrm{d}\mathbf{x}, \tag{1}$$

where Z_θ is the normalization constant or partition function. In principle, any density can be described this way for a suitable choice of E_θ. EBMs are typically learned using maximum likelihood estimation (MLE), where we wish to find a value of θ that minimized the *negative* data log-likelihood:

$$L(\theta) := -\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}(\mathbf{x})}\left[\log p_\theta(\mathbf{x})\right] = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}[E_\theta(\mathbf{x})] + \log Z_\theta, \tag{2}$$

where p_{data} is the data generating distribution.

The fundamental challenge with EBMs for non-standard energy functions is the lack of a closed-form expression for the normalization constant, Z_θ, which hinders exact pointwise density evaluation, sampling and learning. Therefore, these tasks are traditionally performed using approximate methods such as Markov chain Monte Carlo (MCMC, Metropolis et al., 1953; Neal et al., 2011). Similar to the Boltzmann learning rule (Hinton and Sejnowski, 1983; Osogami, 2017), gradient-based learning for MLE involves evaluating the gradient $\nabla_\theta L(\theta) = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}[\nabla_\theta E_\theta(\mathbf{x})] - \mathbb{E}_{\mathbf{x}\sim p_\theta}[\nabla_\theta E_\theta(\mathbf{x})]$, where the expectation in the last term necessitates MCMC approximation. Recently, a series of papers (Kim and Bengio, 2016; Kumar et al., 2019; Abbasnejad et al., 2020; Che et al., 2020) have established links between EBMs and *Wasserstein GANs* (WGANs, Arjovsky et al., 2017), such that EBMs can be approximately learned through a minimax game. This allows for significant improvements in EBM learning over MCMC based methods (Grathwohl et al., 2021).

In this paper, we remark that variational bounds on the value function of the minimax game can be problematic as this is both maximized and minimized. We propose to both upper and lower bound

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

the negative log-likelihood, and alternate between their respective minimization and maximization to alleviate this concern. Evaluation of the bounds requires evaluating the entropy of the generator, and we provide a new efficient estimator. We link the upper bound to the use of gradient penalties which are known to stabilize training. Experimentally, we demonstrate that our approach matches or surpasses state-of-the-art on diverse tasks at negligible performance increase.

1.1 Background: Variational bounds and minimax games

Following Grathwohl et al. (2021), we can lower bound the log normalization constant using a proposal (variational) distribution $p_g(\mathbf{x})$ and Jensen's inequality:

$$\log Z_\theta = \log \mathbb{E}_{\mathbf{x}\sim p_g}\left[\frac{\exp(-E_\theta(\mathbf{x}))}{p_g(\mathbf{x})}\right] \geq \mathbb{E}_{\mathbf{x}\sim p_g}\left[\log \frac{\exp(-E_\theta(\mathbf{x}))}{p_g(\mathbf{x})}\right] = -\mathbb{E}_{\mathbf{x}\sim p_g}[E_\theta(\mathbf{x})] + H[p_g] \tag{3}$$

where $H[p_g] = -\mathbb{E}_{p_g}[\log p_g]$ is the (differential) entropy of the proposal distribution. This means that we also obtain a lower bound on the negative log likelihood function given by

$$\mathcal{L}(\theta) := \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}(\mathbf{x})}[E_\theta(\mathbf{x})] - \mathbb{E}_{\mathbf{x}\sim p_g(\mathbf{x})}[E_\theta(\mathbf{x})] + H[p_g] \leq L(\theta). \tag{4}$$

We note that this bound is tight when $p_g = p_\theta$, since the bound on the log normalization constant (3) can be equivalently expressed as $\log Z_\theta \geq \log Z_\theta - \mathrm{KL}\left(p_g \| p_\theta\right)$. To tighten the bound, we seek a proposal distribution p_g that maximises $\mathcal{L}(\theta)$, while for MLE we want to find a θ that minimizes $\mathcal{L}(\theta)$. Accordingly, MLE can be formulated as a minimax game:

$$\min_{E_\theta} \max_{p_g} \left\{\mathcal{L}(\theta)\right\} = \min_{E_\theta} \max_{p_g} \left\{\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}(\mathbf{x})}[E_\theta(\mathbf{x})] - \mathbb{E}_{\mathbf{x}\sim p_g(\mathbf{x})}[E_\theta(\mathbf{x})] + H[p_g]\right\}. \tag{5}$$

This gives a tractable and MCMC-free approach to EBM learning, assuming we can evaluate the entropy (Grathwohl et al., 2021).

Our key issue is that both minimizing and maximizing a lower bound is potentially unstable. Good results have been reported from using a lower bound in the minimax game (Dai et al., 2017; Kumar et al., 2019; Grathwohl et al., 2021), but the conceptual issue remains.

The WGAN loss function is, as noted by Arjovsky et al. (2017), very similar to the lower bound $\mathcal{L}(\theta)$,

$$L_{\text{WGAN}} = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}\left[E_\theta(\mathbf{x})\right] - \mathbb{E}_{\mathbf{x}\sim p_g}\left[E_\theta(\mathbf{x})\right], \tag{6}$$

with the missing entropy term being the only difference. Lessons from the WGAN can, thus, be expected to apply to the EBM setting as well. One such lesson is that the WGAN objective tend to result in unstable optimization, which Arjovsky et al. (2017) worked around through gradient clipping. We hypothesize that this instability is rooted in the minimization of a lower bound.

2 Approximate minimax games through bidirectional bounds

Instead of solving a minimax game with a lower bounded value function (5), we propose to bound $L(\theta)$ from both above and below:

$$\lfloor L(\theta) \rfloor \leq L(\theta) \leq \lceil L(\theta) \rceil. \tag{7}$$

With this, we can now follow an optimization strategy of alternating

1. minimize $\lceil L(\theta) \rceil$ with respect to E_θ.
2. maximize $\lfloor L(\theta) \rfloor$ with respect to p_g.

This avoids the potential pitfalls of minimizing a lower bound.

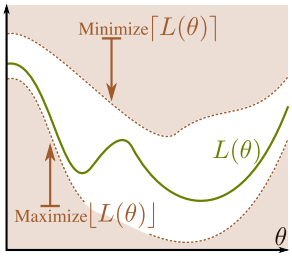


Figure 1: The bidirectional bounds 'sandwich' the negative data log-likelihood.

2.1 A lower bound for maximizing p_g

We already have a lower bound $\mathcal{L}(\theta)$ on $L(\theta)$ in Equation (4). The first two terms of the bound can readily be evaluated by sampling from p_{data} and p_g, respectively. The entropy of p_g is less straight-forward. Following standard practice in GANs, we define p_g through a base variable

$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, which is passed through a 'generator' network $G : \mathbb{R}^d \to \mathbb{R}^D$, i.e. $\mathbf{x} = G(\mathbf{z}) \sim p_g$. By standard change-of-variables, we have

$$\log p_g(G(\mathbf{z})) = \log p_0(\mathbf{z}) - \frac{1}{2} \log \det(\mathbf{J}_\mathbf{z}^\intercal \mathbf{J}_\mathbf{z}), \tag{8}$$

where $\mathbf{J}_\mathbf{z} \in \mathbb{R}^{D \times d}$ is the Jacobian of G at $\mathbf{z}$. The entropy of p_g is then

$$H[p_g] = -\mathbb{E}_{\mathbf{x} \sim p_g} \left[\log p_g(\mathbf{x})\right] = H[p_0] + \mathbb{E}_{\mathbf{z} \sim p_0} \left[\frac{1}{2} \log \det(\mathbf{J}_\mathbf{z}^\intercal \mathbf{J}_\mathbf{z})\right]. \tag{9}$$

The entropy of p_0 trivially evaluates to $H[p_0] = {}^d\!/\!_2(1 + \log(2\pi))$. To avoid an expensive evaluation of the log-determinant-term, we note that it is easily bounded from below:

$$\frac{1}{2} \log \det(\mathbf{J}_\mathbf{z}^\intercal \mathbf{J}_\mathbf{z}) = \frac{1}{2} \sum_{i=1}^{d} \log s_i^2 \geq d \log s_1, \tag{10}$$

where $s_d \geq \ldots \geq s_1$ are the singular values of the Jacobian $\mathbf{J}_\mathbf{z}$. Note that in general, this bound is not tight. With this, we have our final lower bound:

$$L(\theta) \geq \lfloor L(\theta) \rfloor = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})}[E_\theta(\mathbf{x})] - \mathbb{E}_{\mathbf{x} \sim p_g(\mathbf{x})}[E_\theta(\mathbf{x})] + H[p_0] + \mathbb{E}_{\mathbf{z} \sim p_0}\left[d \log s_1(\mathbf{z})\right]. \tag{11}$$

2.2 An upper bound for minimizing E_θ

There are many ways to arrive at an upper bound for the log-likelihood. Due to the strong ties between the lower bound (4) and the WGAN objective, we take inspiration from how WGANs are optimized. The original WGAN paper (Arjovsky et al., 2017) noted optimization instabilities unless *gradient clipping* was applied. Gulrajani et al. (2017) noted that this clipping could be avoided by adding a *gradient penalty*, such that the loss become

$$L_{\text{WGAN-GP}} = L_{\text{WGAN}} + \lambda \mathbb{E}_{\hat{\mathbf{x}} \sim g(\mathbf{x})}\left[(\|\nabla_{\hat{\mathbf{x}}} E_\theta(\hat{\mathbf{x}})\|_2 - 1)^2\right], \tag{12}$$

where $\hat{x}$ is a uniform interpolation between real data and generated samples. This is generally acknowledged as a good way to train a WGAN (Gulrajani et al., 2017). We observe that this loss is similar to our lower bound, but the gradient penalty adds a positive term, such that one could speculate that we might have an upper bound. With this intuition in hand, we prove the following statement in the supplementary material.

Theorem 1 *Suppose $f : \mathcal{X} \to \mathbb{R}$ is L-Lipschitz continuous, $g(x)$ is a probability density function with finite support, then there exists constants $M, m \geq 0$ and $p \geq 1$ such that:*

$$\log \mathbb{E}_{\mathbf{x} \sim g(\mathbf{x})}\left[f(\mathbf{x})\right] - \mathbb{E}_{\mathbf{x} \sim g(\mathbf{x})}\left[\log f(\mathbf{x})\right] \leq M(\mathbb{E}_{\mathbf{x} \sim g(\mathbf{x})}\left[|\nabla_\mathbf{x} \log f(\mathbf{x})|^p\right] + m)^{1/p}. \tag{13}$$

Note that if $\mathbb{E}_{\mathbf{x} \sim g(\mathbf{x})}\left[|\nabla_\mathbf{x} \log f(\mathbf{x})|^p\right] + m \geq 1$, the bound in Equation (13) can be simplified by dropping the power $1/p$,

$$\log\left[\mathbb{E}_{\mathbf{x} \sim g(\mathbf{x})}\left[f(\mathbf{x})\right]\right] - \mathbb{E}_{\mathbf{x} \sim g(\mathbf{x})}\left[\log f(\mathbf{x})\right] \leq M \mathbb{E}_{\mathbf{x} \sim g(\mathbf{x})}\left[|\nabla_\mathbf{x} \log f(\mathbf{x})|^p\right] + Mm. \tag{14}$$

For our model, we set $f(\mathbf{x}) = \frac{\exp(-E_\theta(\mathbf{x}))}{p_g(\mathbf{x})}$ and $g(\mathbf{x}) = p_g(\mathbf{x})$. Empirically, we observe the bound simplification in Equation (14) holds through-out most of the training, and use the following upper bound:

$$\lceil L(\theta) \rceil = \mathcal{L}(\theta) + M \mathbb{E}_{\mathbf{x} \sim p_g(\mathbf{x})}\left[|\nabla_\mathbf{x} E_\theta(\mathbf{x}) + \nabla_\mathbf{x} \log p_g(\mathbf{x})|^p\right] + Mm. \tag{15}$$

For a given choice of $f(\mathbf{x})$, we can view m as a constant and the term Mm can be ignored during optimization. This upper bound can directly be interpreted as the lower bound (4) plus a gradient penalty, albeit one of a different form than the traditional WGAN penalty, which is derived purely from a regularization perspective. Our upper bound can, thus, be seen as a justification of the regularization from a maximum likelihood perspective.

Bound tightness. When $p_\theta = p_g$, we have that $|\nabla_\mathbf{x} E_\theta(\mathbf{x}) + \nabla_\mathbf{x} \log p_g(\mathbf{x})|^p = 0$. In Theorem 1, m is a constant related to the Lipschitz constant of $\log f(\mathbf{x})$ satisfying $|\nabla_\mathbf{x} \log f(\mathbf{x})|^p \leq |\nabla_\mathbf{y} \log f(\mathbf{y})|^p + m$ for all $\mathbf{x}, \mathbf{y}$ (see proof for details). When $p_\theta = p_g$ we also have $|\nabla_\mathbf{x} \log f(\mathbf{x})|^p = 0$, such that $m = 0$. Our upper bound is then $\lceil L(\theta) \rceil = \mathcal{L}(\theta) = L(\theta)$, and hence it is tight.

3 Numerical evaluation of the bounds

Evaluating the lower bound. To evaluate the lower bound in Equation (11), we need the smallest singular value of the Jacobian $\mathbf{J} = \partial_{\mathbf{z}} G(\mathbf{z})$. Recall that this singular value satisfy $s_1 = \|\mathbf{J}\mathbf{v}_{\text{min}}\|_2 = \min_{\mathbf{v}\neq\mathbf{0}} \frac{\|\mathbf{J}\mathbf{v}\|_2}{\|\mathbf{v}\|_2}$. We can then evaluate the singular value by finding $\mathbf{v}_{\text{min}}$ with an iterative optimization algorithm, where we opt to use the celebrated single-vector *LOBPCG algorithm* (Knyazev, 1998). This method performs an iterative minimization of the generalized Rayleigh quotient,

$$\rho(\mathbf{v}) := \frac{\mathbf{v}^\intercal \mathbf{J}^\intercal \mathbf{J}\mathbf{v}}{\mathbf{v}^\intercal \mathbf{v}}, \tag{16}$$

which converges to $\mathbf{v}_{\text{min}}$. The gradient of $\rho(\mathbf{v})$ is proportional to $r = \mathbf{J}^\intercal\mathbf{J}\mathbf{v} - \rho(\mathbf{v})\mathbf{v}$. To avoid computing the Jacobian $\mathbf{J}$, we use Jacobian-vector products, which can be efficiently evaluated using automatic differentiation. To compute $\mathbf{J}^\intercal\mathbf{J}\mathbf{v}$, we use the following trick (in `pytorch`-notation):

$$\mathbf{J}^\intercal\mathbf{J}\mathbf{v} = ((\mathbf{J}\mathbf{v})^\intercal\mathbf{J})^\intercal = \nabla_{\mathbf{z}}\left((\mathbf{J}\mathbf{v})^\intercal \texttt{detach()} \cdot G(\mathbf{z})\right)^\intercal. \tag{17}$$

The optimal learning rate for this iterative scheme can be found by maximizing the Rayleigh quotient (16). Finally, we follow the suggestions of Knyazev (2001) to improve numerical stability and accelerate convergence, which we omit here for brevity.

Evaluating the upper bound. There are two challenges when evaluating Equation (15). The first is to compute $\nabla_{\mathbf{x}} \log p_g(\mathbf{x})$, where we empirically found that existing methods (Shi et al., 2018; Li and Turner, 2018) were too inefficient for our needs. To evaluate the term $\mathbb{E}_{\mathbf{x}\sim p_g(\mathbf{x})}\left[|\nabla_{\mathbf{x}} E_\theta(\mathbf{x}) + \nabla_{\mathbf{x}} \log p_g(\mathbf{x})|^p\right]$, we further loosen the bound

$$\begin{aligned}|\nabla_{G(\mathbf{z})} E_\theta(G(\mathbf{z})) + \nabla_{G(\mathbf{z})} \log p_g(G(\mathbf{z}))|^p &\leq \frac{|\nabla_{G(\mathbf{z})} E_\theta(G(\mathbf{z}))\mathbf{J}_{\mathbf{z}} + \nabla_{G(\mathbf{z})} \log p_g(G(\mathbf{z}))\mathbf{J}_{\mathbf{z}}|^p}{s_1^p} \\ &\leq \frac{|\nabla_{G(\mathbf{z})} E_\theta(G(\mathbf{z}))\mathbf{J}_{\mathbf{z}} + \nabla_{\mathbf{z}} \log p_g(G(\mathbf{z}))|^p}{s_1^p}.\end{aligned} \tag{18}$$

where s_1 is the smallest singular value of $\mathbf{J}_{\mathbf{z}}$. Detailed derivations are in the supplementary material. If we choose $p = 2$, then we can use Hutchinson's estimator (1989):

$$|\nabla_{\mathbf{x}} E_\theta(\mathbf{x})\mathbf{J}_{\mathbf{z}} + \nabla_{\mathbf{z}} \log p_g(G(\mathbf{z}))|^2 = \mathbb{E}_{\mathbf{v}}\left[\left(\nabla_{\mathbf{x}} E_\theta(\mathbf{x})\mathbf{J}_{\mathbf{z}}\mathbf{v} + \nabla_{\mathbf{z}} \log p_g(G(\mathbf{z}))\mathbf{v}\right)^2\right], \tag{19}$$

where $\mathbf{v} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$. This is easily evaluated using automatic differentiation.

The second challenge is to evaluate $\log p_g(\mathbf{x})$ which needs the Jacobian of the generator $G(\mathbf{z})$ as dictated by Equation (8). Here, we opt to use our entropy estimator as described above. We could alternatively use Hutchinson's estimator as proposed by Kumar et al. (2020). Experimentally we do not observe much difference between these two estimators.

4 Related work

In machine learning, there has been a long-standing interest in EBMs dating back to Hopfield networks (Hopfield, 1982), Boltzmann machines (Hinton and Sejnowski, 1983; Ackley et al., 1985) and restricted Boltzmann machines (Smolensky, 1986; Hinton, 2002), see e.g. reviews in the works by LeCun et al. (2006) and Scellier (2020). Learning and evaluation of these models are difficult since the normalization constant cannot be efficiently evaluated. MLE-based learning, such as the Boltzmann learning rule, relies on expensive MCMC sampling to estimate the gradient, and more advanced MCMC methods are used to reliably estimate the normalization constant (see e.g. Grosse et al., 2013; Frellsen et al., 2016). Consequently, learning algorithms that avoid the MCMC approximation have been heavily investigated. For instance, Hinton (2002) proposed k-step Contrastive Divergence (CD-k) to approximate the negative phase log-likelihood gradient, and Hyvärinen (2005) proposed an alternative method to train non-normalized graphical models using score matching. Deep versions of EBMs have subsequently been proposed, such as deep belief networks (Hinton et al., 2006) and deep Boltzmann machines (Salakhutdinov and Hinton, 2009).

In recent years, there have been renewed interest in deep generative EBMs, particularly for image generation. One particular focus area has been learning algorithms. Grathwohl et al. (2021) gave

an excellent overview of the current developments and distinct drawback of the different classes of learning methods, which we summarize here: MLE methods with MCMC sampling are slow and unstable, and while score matching based methods are comparable faster, they are also unstable and do not work with discontinuous nonlinearities. Noise-contrastive estimation (Gutmann and Hyvärinen, 2010) do not have these drawback, but it does not scale well with the data dimensionality.

Our proposed method belongs to a fourth class of algorithms that sidestep the costly MCMC sampling allowing fast sampling. Kim and Bengio (2016) and Zhai et al. (2016) proposed to use neural samplers like the generator of GANs. However, their approach for entropy maximization is different from ours. Kumar et al. (2019) estimated the entropy through its connection to mutual information, but they need an additional network to measure the entropy term. Dai et al. (2017) propose two approaches to entropy maximization: One, which maximizes the entropy by minimizing the conditional entropy using a variational upper bound, and another, which makes isotropic Gaussian assumptions for the data, which is not suitable for high-dimensional data. Grathwohl et al. (2021) used an auxiliary noisy neural sampler, which allows for gradient-based optimization without MCMC. However, the method is memory-consuming, and the hyperparameters are difficult to adjust. Abbasnejad et al. (2019) maximized the entropy by approximating the generator function's Jacobian log-determinant. However, their method is impractical in high dimensions as the Jacobian is computationally expensive.

5 Experiments

To demonstrate the efficiency of our energy-based model with bidirectional bounds (EBM-BB) we compare against a range of methods that represent state-of-the-art. As representatives of the GAN literature, we consider *deep convolutional GANs (DCGANs)* by Radford et al. (2016), *spectrally normalized GANS (SNGANs)* by Miyato et al. (2018) and the *WGAN with zero-centered gradient penalty (WGAN-0GP)* by Thanh-Tung et al. (2019). As representatives of EBMs, we consider the *maximum entropy generators (MEG)* by Kumar et al. (2019) and the *variational entropy regularized approximate maximum likelihood (VERA)* estimator (Grathwohl et al., 2021). To investigate the influence of our derived upper bound, we introduce another baseline in which we replace our upper bound with a zero-centered gradient penalty. We denote the resulting model *EBM-0GP*.

5.1 Training details

On toy data and MNIST, all models are realized with multi-layer perceptrons (MLPs), while for natural images we, use the convolutional architecture from `StudioGAN` (Kang and Park, 2020). Details of the network architecture are given in the supplementary material. All experiments are conducted on a single 12GB NVIDIA Titan GPU using a `pytorch` (Paszke et al., 2017) implementation. To improve training, we use a positive margin for our energy function to balance the bounds. Specifically, we let

$$\lceil L(\theta) \rceil = \lfloor L(\theta) \rfloor + \left[M \mathbb{E}_{\mathbf{x} \sim p_g(\mathbf{x})} \left[|\nabla_{\mathbf{x}} E_\theta(\mathbf{x}) + \nabla_{\mathbf{x}} \log p_g(\mathbf{x})|^p \right] - \zeta \right]_+ , \tag{20}$$

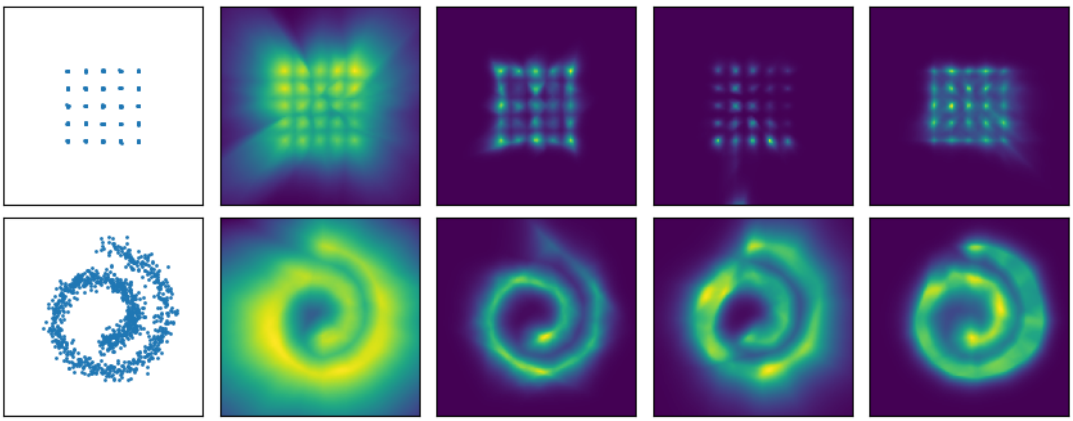
where $[\cdot]_+ = \max(0, \cdot)$ is the usual hinge. In all experiments, $\zeta = 1$ as this allows us to apply the simplified bound in Equation (15).

5.2 Toy data

Figure 2 shows estimated densities on the 25-Gaussians and swiss-roll datasets using both our methods and two baselines. We observe that MEG and EBM-0GP can have inaccurate density information in some edge and peripheral areas. This may be due to insufficient or excessive maximization of entropy since the zero-centered gradient penalty is not a principled objective for maximum likelihood. VERA and our EBM-BB can learn a sharp distribution, but our method is more stable in some inner regions of the 25-Gaussians and edge regions of the swiss-roll.

Table 1: Number of captured modes and KL divergence between the real and generated distributions.

Model	Modes↑	KL↓
DCGAN	392 ± 7.4	8.012 ± 0.056
SNGAN	441 ± 39.0	2.755 ± 0.033
WGAN-0GP	$\mathbf{1000 \pm 0.0}$	0.048 ± 0.003
MEG	970 ± 1.5	0.662 ± 0.051
VERA	989 ± 9.0	0.152 ± 0.037
EBM-0GP(ours)	$\mathbf{1000 \pm 0.0}$	$\mathbf{0.044 \pm 0.003}$
EBM-BB (ours)	$\mathbf{1000 \pm 0.0}$	0.045 ± 0.003



(a) Data (b) MEG (c) VERA (d) EBM-0GP (ours) (e) EBM-BB (ours)

Figure 2: Density estimation on the 25-Gaussians and swiss-roll datasets.

5.3 MNIST

Mode counting. Mode collapse is a frequently occurring phenomenon in which the generator function maps all latent input to a small number of points in the observation space. In particular, GANs are plagued by this problem. Since the generator is trained to maximize the entropy of the generated distribution, several EBM-based models have been shown to capture all the modes of the data distribution. To empirically verify that our model also captures a variety of modes in the data distribution, we follow the test procedure of Kumar et al. (2019). We train our generative model on the StackedMNIST dataset, a synthetic dataset created by stacking MNIST on different channels. The true total number of modes is $1{,}000$, and they are counted using a pretrained MNIST classifier. The KL divergence is calculated empirically between the generated mode distribution and the data distribution. The results appear in Table 1. As expected, GAN-based methods suffer from mode collapse except for WGAN-0GP. All the EBM-based methods capture all or nearly all the modes of the data distribution. Both our model variants, EBM-0GP and EBM-BB, capture all modes and report the smallest KL measures. It worth mentioned that WGAN-0GP obtains comparable results, which was also observed by Grathwohl et al. (2021).

Entropy Estimation. In Figure 3, we explore the quality of our entropy estimator measured on MNIST (LeCun, 1998). We use networks with fully connected layers as the Jacobian is then easily derived in closed-form, giving us a ground truth for the entropy. We compare our estimator (red) with the ground truth (black) and Hutchinson's estimator (blue), as was proposed by Kumar et al. (2020). Finally, as we only run our iterative estimator for a few steps, we also compare with an estimator that uses the smallest singular value computed with high precision (green). We observe that Hutchinson's estimator is reasonable close to the ground truth but provides an upper bound, making it inapplicable for our lower bound. We further observe a noticeable gap between our estimator and the high precision singular value estimator. Finally, we see that all estimators follow roughly the same trend, which suggests that our estimator provides a suitable target for optimization.

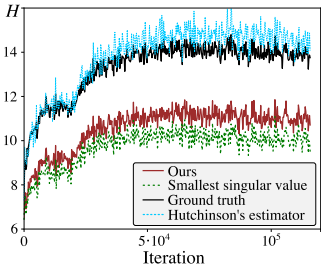


Figure 3: Entropy estimators.

5.4 Natural Images

Image generation. The data studied thus far are simple and perhaps do not challenge our upper bound and entropy estimator. We, therefore, train our model on the standard benchmark 32×32 CIFAR-10 (Krizhevsky et al., 2009) dataset and the 64×64 cropped ANIMEFACE[1] dataset, which both represent a significant increase in complexity. Similar to recent work on GANs (Miyato et al., 2018), we report Inception Score (IS), Fréchet Inception Distance (FID) scores and two F_β scores

[1] https://www.kaggle.com/splcher/animefacedataset

(a) WGAN-0GP (b) MEG (c) VERA (d) EBM-BB (ours)

Figure 4: Generated samples on CIFAR-10 with our method and various methods.

(Sajjadi et al., 2018). We compare with competitive GAN and EBM baselines (Table 2). All methods are reproduced using the same network architecture (see supplements for details) except DCGAN and IGEBM, where we respectively borrow the results from StudioGAN (Kang and Park, 2020) and the original implementation (Du and Mordatch, 2019). We use the DCGAN (Radford et al., 2016) network architecture for CIFAR-10 and a Resnet architecture (Kang and Park, 2020) for ANEMIFACE. Parameters are chosen as in the original papers. For VERA (Grathwohl et al., 2021), we were unable to reproduce the reported performance, so we choose hyper-parameters according to an extensive grid search. Like the original paper, we choose the entropy weight to be 0.0001.

From Table 2, we see that our EBM-BB model obtains the top performance on CIFAR-10 and is highly competitive on ANIMEFACE. Model samples are shown in Figures 4 and 5. For ANIMEFACE, our method generates more diverse samples in terms of face parts than the baselines. All models predominantly generate female faces. This suggests that while our approach captures more density modes than the baselines, all approaches still misses several modes. We further draw attention to several corrupt samples generated by VERA, despite extensive parameter search. For CIFAR-10, we observe no immediate differences in the generative capabilities between models.

Capacity usage. For the proposal distribution p_g to be adaptive, the associated generator network $G(\mathbf{z})$ should be able to use as much of its available capacity as possible. To compare different methods, we consider an implicit measure of capacity usage $\mathcal{C}_\mathbf{z}$ that locally measures the intrinsic dimensionality of $G(\mathbf{z})$. In particular, we use the anisotropy index (Wang and Ponce, 2021) that, for a given $\mathbf{z}$, measure the standard deviation of the norm of the directional derivative of $G(\mathbf{z})$ along the input dimensions, i.e.

Table 3: Anisotropy indices (↓).

Model	CIFAR-10	ANIMEFACE
WGAN-0GP	1.0823 ± 0.005	2.4366 ± 0.129
MEG	0.9467 ± 0.009	2.3600 ± 0.142
VERA	3.9694 ± 0.055	2.2929 ± 0.2074
EBM-0GP (ours)	1.0069 ± 0.011	2.4727 ± 0.0288
EBM-BB (ours)	$\mathbf{0.9431 \pm 0.0191}$	$\mathbf{1.8016 \pm 0.1345}$

$$\mathcal{C}_\mathbf{z} = \mathrm{std}\left(\{\|\mathbf{J}_\mathbf{z}\mathbf{e}_i\|\}_{i=1\ldots d}\right), \tag{21}$$

where $\mathrm{std}(\cdot)$ computes the standard deviation of the input, and $\mathbf{e}_i$ denotes the ith standard basis vector. A small $\mathcal{C}_\mathbf{z}$ indicates that the different input dimensions contribute equally to the output of G, which imply good capacity usage. We measure the mean value of $\mathcal{C}_\mathbf{z}$ for $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$ and report mean and standard deviation of the result running for several times in Table 3. We observe that most models, with the exception of VERA, perform well on CIFAR-10, while on ANIMEFACE there is more diversity. In both cases, EBM-BB has the best capacity usage. The large difference between EBM-0GP and EBM-BB on ANIMEFACE suggest that our proposed upper bound helps increase the entropy of the proposal distribution p_g.

(a) WGAN-0GP (b) MEG (c) VERA (d) EBM-BB (ours)

Figure 5: Generated samples on ANIMEFACE with our method and other generative models

Table 2: Comparison in terms of FID, Inception Score, F_8 score (weights recall higher than precision) and $F_{1/8}$ score (weights precision higher than recall).

Model	Inception↑	FID↓	F_8↑	$F_{1/8}$↑
		CIFAR-10		
DCGAN	6.64	49.03	0.795	0.83
WGAN-0GP	7.24 ± 0.035	29.31 ± 0.185	0.92 ± 0.010	0.95 ± 0.010
IGEBM	6.78	38.2	-	-
MEG	6.62 ± 0.243	34.55 ± 1.145	0.88 ± 0.001	0.92 ± 0.010
VERA	5.06 ± 0.555	66.38 ± 6.635	0.58 ± 0.080	0.79 ± 0.005
EBM-0GP (ours)	7.31 ± 0.029	28.82 ± 0.165	0.915 ± 0.005	0.95 ± 0.005
EBM-BB (ours)	$\mathbf{7.45 \pm 0.014}$	$\mathbf{28.63 \pm 0.290}$	$\mathbf{0.93 \pm 0.001}$	$\mathbf{0.95 \pm 0.008}$
		ANIMEFACE		
WGAN-0GP	2.22 ± 0.030	9.76 ± 0.674	$\mathbf{0.95 \pm 0.005}$	0.98 ± 0.005
MEG	2.20 ± 0.020	$\mathbf{9.31 \pm 0.007}$	$\mathbf{0.95 \pm 0.005}$	$\mathbf{0.98 \pm 0.001}$
VERA	2.15 ± 0.001	41.00 ± 1.072	0.515 ± 0.078	0.78 ± 0.013
EBM-0GP (ours)	2.23 ± 0.050	24.69 ± 0.095	0.836 ± 0.017	0.894 ± 0.006
EBM-BB (ours)	$\mathbf{2.26 \pm 0.005}$	12.75 ± 0.045	0.94 ± 0.001	0.96 ± 0.005

Out-of-distribution detection. As EBMs are density estimators, they should, in principle, assign low likelihood to out-of-distribution (OOD) observations. OOD detection performance can then be seen as a proxy for the quality of the estimated density. To test how well our model fares in this regard, we follow previous work (Hendrycks and Gimpel, 2017; Hendrycks et al., 2019; Alemi et al., 2018; Choi et al., 2018; Ren et al., 2019; Havtorn et al., 2021) and report the threshold independent evaluation metrics of Area Under the Receiver Operator Characteristic (AUROC↑), Area Under the Precision Recall Curve (AUPRC↑) and False Positive Rate at 80% (FPR80↓), where the arrow indicates the direction of improvement of the metrics. The results are reported in Table 4. Each column of the table takes the form 'In-distribution / Out-of-distribution' in reference to the training and test set, respectively. We observe that on CIFAR-10 / SVHN, EBM-BB is the top performer. On CIFAR-10 / CIFAR-100 the overall performance degraded significantly, EBM-0GP fared noticeable better than other models. The overall degradation may be caused by the strong similarity between CIFAR-10 and CIFAR-100. For ANIMEFACE / Bedroom a similar situation occurs, even if these datasets are highly dissimilar.

Table 4: AUROC↑, AUPRC↑ and FPR80↓ for OOD detection for 'train / test' datasets.

Model	CIFAR-10 / SVHN			CIFAR-10 / CIFAR-100			ANIMEFACE / Bedroom		
	AUROC↑	AUPRC↑	FPR80↓	AUROC↑	AUPRC↑	FPR80↓	AUROC↑	AUPRC↑	FPR80↓
WGAN-0GP	0.8	0.71	0.39	0.54	0.55	0.77	0.51	0.85	0.72
MEG	0.786	0.67	0.42	0.52	0.53	0.8	0.56	0.82	0.77
VERA	0.62	0.43	0.64	0.51	0.51	0.79	**0.60**	**0.87**	**0.6**
EBM-0GP (ours)	0.63	0.79	0.58	**0.62**	**0.61**	**0.637**	**0.60**	0.74	0.76
EBM-BB (ours)	**0.88**	**0.95**	**0.1997**	0.53	0.54	0.7765	0.53	0.83	0.83

No clear winner can be found from this study. We note that our method consistently performs well, but surprisingly, so does WGAN-0GP even if it is not a density estimator. As OOD detection is a task that comes with many subtle pitfalls (Havtorn et al., 2021), we suggest that these results should be taken with a grain of salt even if our model performs well.

Running time. Table 5 gives an overview of the time required to train the proposed models and the baselines. All models were trained on a single 12GB Titan GPU. The computational cost of our method is comparable per-epoch to similar models such as WGAN-0GP, but we observe that our model should be trained with a lower learning rate, and therefore may increase the total number of epochs.

Table 5: Total training time.

Model	Iterations	Runtime
CIFAR-10		
WGAN-0GP	100000	11h
MEG	100000	10h
VERA	200000	15h
EBM-0GP	200000	20h
EBM-BB	200000	20h
ANIMEFACE		
WGAN-0GP	100000	16h
MEG	100000	27h
VERA	200000	44h
EBM-0GP	100000	38h
EBM-BB	100000	40h

6 Discussion

The first observation in our work is that current methods for training energy-based models (EBMs) interchangeably minimize and maximize a lower bound. As this may be a potential source of training instability, we propose to bound the negative log-likelihood from above and below and switch between bounds when minimizing and maximizing. The lower bound, $\lfloor L(\theta) \rfloor$, is similar to existing ones, but we provide a new algorithm for its realization. Unlike past work, this algorithm does not need additional networks and does not rely on hard-to-tune parameters; our only parameter controls the number of iterations for the Jacobian approximation and represents a trade-off between the accuracy of the bound and computational budget. The upper bound, $\lceil L(\theta) \rceil$, is new to the literature but similar to the common regularization practice of introducing gradient penalties. To the best of our knowledge, this is the first time that gradient penalties have been derived from the perspective of bounding the log-likelihood. It is rewarding that current best engineering practice can be justified from a probabilistic perspective. Empirically, we find our proposed model generally performs as well or better than the current state-of-the-art on a variety of tasks. The evidence suggests that our bidirectional bounds allow the generator to increase its entropy and capacity usage. We see this both directly and through improved sample quality on diverse datasets.

Limitations. The current drawbacks of the method are mainly three concerns. First, we find that a smaller-than-usual learning rate helps our model. While we expect this to be a matter of implementation rather than a more profound concern, the current implication is that training is approximately twice that of a WGAN-0GP. Second, our bounds rely on computing the smallest singular value of the Jacobian of the generator. We provide an efficient implementation of a method for this task, but it needs to run for several iterations to guarantee convergence. In practice, we stop after a fixed, low number of iterations, which we found to work well, but technically this violates the bound. This seems to be an unavoidable aspect of our approach, but we find that the benefits significantly compensate for this issue. The availability of a fast (approximate) entropy estimator that does not require additional networks and parameters is highly valuable when training EBMs. Finally, for the upper bound, we need an estimate of the volume of support M of the proposal distribution p_g. We do not have a viable method for this, and, in practice, we treat M as a hyper-parameter. We have not found it difficult to tune this parameter, but it nonetheless constitutes a limitation.

Negative societal impact. All high-capacity generative models carry the risk of being used for misinformation, and our model is no exception. The value of EBMs over GANs is that they come with a likelihood function, which is more valuable in data analysis, than e.g. for creating deepfakes.

References

M. E. Abbasnejad, Q. Shi, A. v. d. Hengel, and L. Liu. A generative adversarial density estimator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10782–10791, 2019.

M. E. Abbasnejad, J. Shi, A. van den Hengel, and L. Liu. Gade: A generative adversarial approach to density estimation and its applications. *International Journal of Computer Vision*, 128(10): 2731–2743, 2020.

D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive Science*, 9(1):147–169, 1985.

A. A. Alemi, I. Fischer, and J. V. Dillon. Uncertainty in the variational information bottleneck. *arXiv preprint arXiv:1807.00906*, 2018.

M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

T. Che, R. Zhang, J. Sohl-Dickstein, H. Larochelle, L. Paull, Y. Cao, and Y. Bengio. Your GAN is secretly an energy-based model and you should use discriminator driven latent sampling. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 12275–12287. Curran Associates, Inc., 2020.

H. Choi, E. Jang, and A. A. Alemi. WAIC, but why? Generative ensembles for robust anomaly detection. *arXiv preprint arXiv:1810.01392*, 2018.

Z. Dai, A. Almahairi, P. Bachman, E. Hovy, and A. Courville. Calibrating energy-based generative adversarial networks. In *International Conference on Learning Representations*, 2017.

Y. Du and I. Mordatch. Implicit generation and modeling with energy based models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

J. Frellsen, O. Winther, Z. Ghahramani, and J. Ferkinghoff-Borg. Bayesian generalised ensemble markov chain monte carlo. In A. Gretton and C. C. Robert, editors, *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pages 408–416, Cadiz, Spain, 09–11 May 2016. PMLR.

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

R. B. Grosse, C. J. Maddison, and R. R. Salakhutdinov. Annealing between distributions by averaging moments. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein GANs. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

M. Gutmann and A. Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, 2010.

J. D. Havtorn, J. Frellsen, S. Hauberg, and L. Maaløe. Hierarchical VAEs know what they don't know. *arXiv preprint arXiv:2102.08248*, 2021.

D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*, 2017.

D. Hendrycks, M. Mazeika, and T. Dietterich. Deep anomaly detection with outlier exposure. In *International Conference on Learning Representations*, 2019.

G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

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

G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

J. J. Hopfield. 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.

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

A. Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

M. Kang and J. Park. ContraGAN: Contrastive learning for conditional image generation. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21357–21369. Curran Associates, Inc., 2020.

T. Kim and Y. Bengio. Deep directed generative models with energy-based probability estimation. *arXiv preprint arXiv:1606.03439*, 2016.

A. V. Knyazev. Preconditioned eigensolvers—an oxymoron. *Electron. Trans. Numer. Anal*, 7: 104–123, 1998.

A. V. Knyazev. Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method. *SIAM journal on scientific computing*, 23(2):517–541, 2001.

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

A. Kumar, B. Poole, and K. Murphy. Regularized autoencoders via relaxed injective probability flow. In *International Conference on Artificial Intelligence and Statistics*, pages 4292–4301. PMLR, 2020.

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

Y. LeCun. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

Y. Li and R. E. Turner. Gradient estimators for implicit models. In *International Conference on Learning Representations*, 2018.

N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

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

R. M. Neal et al. MCMC using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2 (11):2, 2011.

T. Osogami. Boltzmann machines and energy-based models. *arXiv preprint arXiv:1708.06008*, 2017.

A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. *NIPS 2017 Workshop Autodiff*, 2017.

A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations*, 2016.

J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. Depristo, J. Dillon, and B. Lakshminarayanan. Likelihood ratios for out-of-distribution detection. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

M. S. M. Sajjadi, O. Bachem, M. Lučić, O. Bousquet, and S. Gelly. Assessing Generative Models via Precision and Recall. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

R. Salakhutdinov and G. Hinton. Deep boltzmann machines. In D. van Dyk and M. Welling, editors, *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pages 448–455, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA, 16–18 Apr 2009. PMLR.

B. Scellier. *A deep learning theory for neural networks grounded in physics*. PhD thesis, Université de Montréal, Quebec, Canada, 2020.

J. Shi, S. Sun, and J. Zhu. A spectral approach to gradient estimation for implicit distributions. In *International Conference on Machine Learning*, pages 4644–4653. PMLR, 2018.

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

H. Thanh-Tung, T. Tran, and S. Venkatesh. Improving generalization and stability of generative adversarial networks. In *International Conference on Learning Representations*, 2019.

B. Wang and C. R. Ponce. The geometry of deep generative image models and its applications. *arXiv preprint arXiv:2101.06006*, 2021.

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

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] The last section explicitly raise this point of discussion.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] The last section explicitly raise this point of discussion.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] All theorems are fully stated.
   (b) Did you include complete proofs of all theoretical results? [Yes] They are available in the supplementary material.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] The code will be released under an open source license upon paper acceptance. We are unable to share the code as part of the supplementary material as it has to be anonymous, thereby preventing us from assigning a license to the code.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] The supplementary material provides the relevant details.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A] We reference all applied assets and let the referenced material provide the authoritative license.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] We only consider established benchmark data.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]