Maximum Entropy Generators for Energy-Based Models

Rithesh Kumar [1] **Anirudh Goyal** [1] **Aaron Courville** [1 2] **Yoshua Bengio** [1 2 3]

Abstract

Unsupervised learning is about capturing dependencies between variables and is driven by the contrast between the probable vs. improbable configurations of these variables, often either via a generative model that only samples probable ones or with an energy function (unnormalized log-density) that is low for probable ones and high for improbable ones. Here, we consider learning both an energy function and an efficient approximate sampling mechanism. Whereas the discriminator in generative adversarial networks (GANs) learns to separate data and generator samples, introducing an entropy maximization regularizer on the generator can turn the interpretation of the critic into an energy function, which separates the training distribution from everything else, and thus can be used for tasks like anomaly or novelty detection. Then, we show how Markov Chain Monte Carlo can be done in the generator latent space whose samples can be mapped to data space, producing better samples. These samples are used for the negative phase gradient required to estimate the log-likelihood gradient of the data space energy function. To maximize entropy at the output of the generator, we take advantage of recently introduced neural estimators of mutual information. We find that in addition to producing a useful scoring function for anomaly detection, the resulting approach produces sharp samples while covering the modes well, leading to high Inception and Fréchet scores.

1. Introduction

The early work on deep learning relied on unsupervised learning (Hinton et al., 2006; Bengio et al., 2007; Larochelle et al., 2009) to train energy-based models (LeCun et al., 2006), in particular Restricted Boltzmann Machines, or RBMs. However, it turned out that training energy-based models without an analytic form for the normalization constant is very difficult, because of the challenge of estimating the gradient of the partition function, also known as the negative phase part of the log-likelihood gradient (described in more details below, Sec. 2). Several algorithms were proposed for this purpose, such as Contrastive Divergence (Hinton, 2000) and Stochastic Maximum Likelihood (Younes, 1998; Tieleman, 2008), relying on Markov Chain Monte Carlo (MCMC) to approximately sample from the energy-based model. However, because they appear to suffer from either high bias or high variance (due to long mixing times), training of RBMs and other Boltzmann machines has not remained competitive after the introduction of variational auto-encoders (Kingma & Welling, 2014) and generative adversarial networks or GANs (Goodfellow et al., 2014).

In this paper, we revisit the question of training energy-based models, taking advantage of recent advances in GAN-related research, and we propose a novel approach – called MEG for Maximum Entropy Generators – for training energy functions and sampling from them. The main inspiration for the proposed solution is the earlier observation from (Bengio et al., 2013) who noticed that sampling in latent space of a stack of auto-encoders (and then applying a decoder to map back to data space) led to faster mixing and more efficient sampling. The authors observed that whereas the data manifold is generally very complex and curved, the corresponding distribution in latent space tends to be much simpler and flatter. This was visually verified by interpolating in latent space and projecting back to data space through the decoder, observing that the resulting samples look like examples from the data distribution. In other words, the latent space manifold is approximately convex, with most points interpolated between examples encoded in latent space also having high probability. MEG is a related approach that also provides two energy functions, one in data space and one in latent space. A key ingredient of the proposed approach is the need to regularize the generator so as to increase the entropy of its output random variable. This is needed to ensure the sampling of diverse negative examples that can kill off spurious minima of the energy function. This need was first identified by Kim & Bengio (2016), who showed that, in order for an approximate sampler to match the density associated with an energy

[1]Montréal Institute for Learning Algorithms (Mila) [2]Canadian Institute for Advanced Research (CIFAR) [3]Institute for Data Valorization (IVADO). Correspondence to: Rithesh Kumar <rithesh.kumar@umontreal.ca>.

function, a compromise must be reached between sampling low-energy configurations and obtaining a high-entropy distribution. However, estimating and maximizing the entropy of a complex high-dimensional distribution is not trivial, and for this purpose we take advantage of very recently proposed GAN-based approaches for maximizing mutual information (Belghazi et al., 2018; Oord et al., 2018; Hjelm et al., 2018), since the mutual information between the input and the output of the generator is equal to the entropy at the output of the generator.

In this context, the main contributions of this paper are the following:

- proposing MEG, a general architecture, sampling and training framework for energy functions, taking advantage of an estimator of mutual information between latent variables and generator output and approximating the negative phase samples with MCMC in latent space.
- improving over (Kim & Bengio, 2016) by using GAN-based entropy maximization and the gradient norm regularizer on the energy, preventing the explosion of energy values during training (a major issue we faced while reproducing (Kim & Bengio, 2016)).
- showing that the resulting energy function can be successfully used for anomaly detection, thereby improving on recently published results with energy-based models;
- showing that MEG produces sharp images - with competitive Inception and Fréchet scores - and which also better cover modes than standard GANs and WGAN-GPs, while not suffering from the common blurriness issue of many maximum likelihood generative models.

2. Likelihood Gradient Estimator for Energy-Based Models and Difficulties with MCMC-Based Gradient Estimators

Let $\boldsymbol{x}$ denote a sample in the data space $\mathcal{X}$ and $E_\theta : \mathcal{X} \to \mathbb{R}$ an energy function corresponding to minus the logarithm of an unnormalized estimated density density function

$$p_\theta(\boldsymbol{x}) = \frac{e^{-E_\theta(\boldsymbol{x})}}{Z_\theta} \propto e^{-E_\theta(\boldsymbol{x})} \tag{1}$$

where $Z_\theta := \int e^{-E_{\boldsymbol{\theta}}(\boldsymbol{x})} d\boldsymbol{x}$ is the partition function or required normalizing constant. Let p_D be the training distribution, from which the training set is drawn. Towards optimizing the parameters θ of the energy function, the maximum likelihood parameter gradient is

$$\frac{\partial \mathbb{E}_{\boldsymbol{x}\sim p_D}[-\log p_\theta(\boldsymbol{x})]}{\partial \theta} = \mathbb{E}_{\boldsymbol{x}\sim p_D}\left[\frac{\partial E_\theta(\boldsymbol{x})}{\partial \theta}\right] - \mathbb{E}_{\boldsymbol{x}\sim p_\theta(\boldsymbol{x})}\left[\frac{\partial E_\theta(\boldsymbol{x})}{\partial \theta}\right] \tag{2}$$

where the second term is the gradient of $\log Z_\theta$, and the sum of the two expectations is zero when training has converged, with expected energy gradients in the positive phase (under the data p_D) matching those under the negative phase (under $p_\theta(\boldsymbol{x})$). Training thus consists in trying to separate two distributions: the positive phase distribution (associated with the data) and the negative phase distribution (where the model is free-running and generating configurations by itself). This observation has motivated the pre-GAN idea presented by Bengio (2009) that "model samples are negative examples" and a classifier could be used to learn an energy function if it separated the data distribution from the model's own samples. Shortly after introducing GANs, Goodfellow (2014) also made a similar connection, related to noise-contrastive estimation (Gutmann & Hyvarinen, 2010). One should also recognize the similarity between Eq. 2 and the objective function for Wasserstein GANs or WGAN (Arjovsky et al., 2017). In the next section, we examine a way to train what appears to be a particular form of WGAN that makes the discriminator compute an energy function.

The main challenge in Eq. 2 is to obtain samples from the distribution p_θ associated with the energy function E_θ. Although having an energy function is convenient to obtain a score allowing to compare the relative probability of different $\boldsymbol{x}$'s, it is difficult to convert an energy function into a generative process. The commonly studied approaches for this are based on Markov Chain Monte Carlo, in which one iteratively updates a candidate configuration, until these configurations converge in distribution to the desired distribution p_θ. For the RBM, the most commonly used algorithms have been Contrastive Divergence (Hinton, 2000) and Stochastic Maximum Likelihood (Younes, 1998; Tieleman, 2008), relying on the particular structure of the RBM to perform Gibbs sampling. Although these MCMC-based methods are appealing, RBMs (and their deeper form, the deep Boltzmann machine) have not been competitive in recent years compared to autoregressive models (van den Oord et al., 2016), variational auto-encoders (Kingma & Welling, 2014) and generative adversarial networks or GANs (Goodfellow et al., 2014).

What has been hypothesized as a reason for the poorer results obtained with energy-based models trained with an MCMC estimator for the negative phase gradient is that running a Markov chain in data space is fundamentally difficult when the distribution is concentrated (e.g, near manifolds) and has many modes separated by vast areas of low probability. This mixing challenge is discussed by Bengio et al.

(2013) who argue that a Markov chain is very likely to produce only sequences of highly probable configurations. If two modes are far from each other and only local moves are possible (which is typically the case with MCMCs), it becomes exponentially unlikely to traverse the 'desert' of low probability that can separate two modes. This makes mixing between modes difficult in high-dimensional spaces with strong concentration of probability mass in some places (e.g. corresponding to different categories) and very low probability elsewhere. In the same paper, the authors propose a heuristic method for jumping between modes, based on performing the random walk not in data space but rather in the latent space of an auto-encoder. Data samples can then be obtained by mapping the latent samples to data space via the decoder. They argue that auto-encoders tend to flatten the data distribution and bring the different modes closer to each other. The MEG sampling method proposed here is similar but leads to learning both an energy function in data space and one in latent space, from which we find that better samples are obtained, after the latent space samples are transformed into data space samples by the generator network. The energy function can be used to perform the appropriate Metropolis-Hastings rejection. Having an efficient way to approximately sample from the energy function also opens to the door to estimating the log-likelihood gradient with respect to the energy function according to Eq. 2, as outlined below.

3. Turning GAN Discriminators into Energy Functions with Entropy Maximization

Turning a GAN discriminator into an energy function has been studied in the past (Kim & Bengio, 2016; Zhao et al., 2016; Dai et al., 2017) but in order to turn a GAN discriminator into an energy function, a crucial and difficult requirement is the maximization of entropy at the output of the generator. Let us see why. In Eq. 2, we can replace the difficult to sample p_θ by another generative process, say p_G, such as the generative distribution associated with a GAN generator:

$$\mathcal{L}_E = \mathbb{E}_{\boldsymbol{x}\sim p_D}\left[E_\theta(\boldsymbol{x})\right] - \mathbb{E}_{\boldsymbol{x}\sim p_G(\boldsymbol{x})}\left[E_\theta(\boldsymbol{x})\right] + \Omega \tag{3}$$

$$\frac{\partial \mathcal{L}_E}{\partial \theta} = \mathbb{E}_{\boldsymbol{x}\sim p_D}\left[\frac{\partial E_\theta(\boldsymbol{x})}{\partial \theta}\right] - \mathbb{E}_{\boldsymbol{x}\sim p_G(\boldsymbol{x})}\left[\frac{\partial E_\theta(\boldsymbol{x})}{\partial \theta}\right] + \frac{\partial \Omega}{\partial \theta} \tag{4}$$

where Ω is a regularizer that we found necessary to avoid numerical problems in the scale (temperature) of the energy. In this paper we use a gradient norm regularizer $\Omega = \left|\left|\nabla_{\boldsymbol{x}} E_\theta(\boldsymbol{x})\right|\right|^2$ (Gulrajani et al., 2017) for this purpose, where $\boldsymbol{x}$ is sampled from the training data distribution.

Justification for the gradient norm regularizer Apart from acting as a smoothness regularizer, it also makes data points $\boldsymbol{x}$ energy minima, because $\left|\left|\frac{\partial E(\boldsymbol{x})}{\partial \boldsymbol{x}}\right|\right|$ should be 0 at data points. It encourages observed data x to lie near minima of the energy function (as they should if the probability is concentrated). We found this helps to learn a better energy function, and it is similar in spirit to score matching, which also carves the energy function so that it has local minima at the training points. The regularizer also stabilizes the temperature (scale) of the energy function, making training stable, avoiding continued growth of the magnitude of energies as training continues. This regularized objective is also similar to the training objective of a WGAN-GP (Gulrajani et al., 2017).

However, approximating p_θ with p_G works if the approximation is good. To achieve this, as first proposed by Kim & Bengio (2016), consider optimizing G to minimize the KL divergence $KL(p_G||p_\theta)$, which can be rewritten in terms of minimizing the energy of the samples from the generator while maximizing the entropy at the output of the generator:

$$KL(p_G||p_\theta) = -H[p_G] - E_{p_G}[\log p_\theta(\boldsymbol{x})] \tag{5}$$

When taking the gradient of $KL(p_G||p_\theta)$ with respect to the parameters $\boldsymbol{w}$ of the generator, the partition function of p_G disappears and we can equivalently tune $\boldsymbol{w}$ to minimize

$$\mathcal{L}_G = -H[p_G] + \mathbb{E}_{\boldsymbol{z}\sim p_z} E_\theta(G(\boldsymbol{z})) \tag{6}$$

where p_z is the prior distribution of the latent variable of the generator.

In order to approximately maximize the entropy $H[p_G]$ at the output of the generator, we propose to exploit another GAN-derived framework in order to estimate and maximize mutual information between the input and output of the generator network. The entropy at the output of a deterministic function (the generator in our case) can be computed using an estimator of mutual information between the input and output of that function, since the conditional entropy term is 0 because the function is deterministic. With $x = G(z)$ the function of interest and x discrete-valued: [1]

$$\begin{aligned} I(X,Z) &= H(X) - H(X|Z) \\ &= H(G(Z)) - \cancelto{0}{H(G(Z)|Z)} \end{aligned}$$

This idea of maximizing mutual information between inputs and outputs of an encoder to maximize the entropy of its output dates back at least to Linsker (1988); Bell & Sejnowski (1995) and their work on Infomax. This suggests using

[1] Although X and Z are perceived as continuous values and conditional differential entropy cannot be dismissed as 0, the authors would like to point out that these variables are actually floating point numbers with finite 32-bit (discrete) precision. Viewed like this, it is still true that conditional entropy is 0 (since even if there are quantization issues we get some loss of information, but still only one value of X is mapped to each given Z value)

modern neural mutual information maximization methods such as MINE (Belghazi et al., 2018), noise contrastive estimation (Oord et al., 2018) or DeepINFOMAX (Hjelm et al., 2018) to estimate and maximize the entropy of the generator. All these estimators are based on training a discriminator that separates the joint distribution $p(X, Z)$ from the product of the corresponding marginals $p(X)p(Z)$. As proposed by Brakel & Bengio (2017) in the context of using a discriminator to minimize statistical dependencies between the outputs of an encoder, the samples from the marginals can be obtained by creating negative examples pairing an X and a Z from different samples of the joint, e.g., by independently shuffling each column of a matrix holding a minibatch with one row per example. The training objective for the discriminator can be chosen in different ways. In this paper, we used the Deep INFOMAX (DIM) estimator (Hjelm et al., 2018), which is based on maximizing the Jensen-Shannon divergence between the joint and the marginal (see (Nowozin et al., 2016) for the original F-GAN formulation).

$$\begin{aligned}\mathcal{I}^{JSD}(X, Z) =& \mathbb{E}_{p(X,Z)}[-s_+(-T(X, Z))] \\ &- \mathbb{E}_{p(X)p(Z)}[s_+(T(X, Z))]\end{aligned} \tag{7}$$

where $s_+(a) = \log(1 + e^a)$ is the softplus function. The discriminator T used to increase entropy at the output of the generator is trained by maximizing $\mathcal{I}^{JSD}(X, Z)$ with respect to the parameters of T. With $X = G(Z)$ the output of the generator, $\mathcal{I}^{JSD}(G(Z), Z)$ is one of the terms to be minimized in the objective function for training G, with the effect of maximizing the generator's output entropy $H(G(Z))$.

Using the JSD approximation, the actual training objective we used for G is

$$\mathcal{L}_G = -\mathcal{I}^{JSD}(G(Z), Z) + \mathbb{E}_{\boldsymbol{z}\sim p_z} E_\theta(G(\boldsymbol{z})) \tag{8}$$

where $Z \sim p_z$, the latent prior (typically a $N(0, I)$ Gaussian).

4. Proposed Latent Space MCMC For Obtaining Better Samples

One option to generate samples is simply to use the usual GAN approach of sampling a $z \sim p_z$ from the latent prior and then output $x = G(z)$, i.e., obtain a sample $\boldsymbol{x} \sim p_G$. Since we have an energy function, another option is to run MCMC in data space, and we have tried this with both Metropolis-Hastings (with a Gaussian proposal) and adjusted Langevin (detailed below, which does a gradient step down the energy and adds noise, then rejects high-energy samples). However, we have interestingly obtained the best samples by considering $E_\theta \circ G$ as an energy function in latent space and running an adjusted Langevin in that space (compare Fig. 4 with Fig. 1). Then, in order to produce a data space sample, we apply G. For performing the MCMC sampling, we use the Metropolis-adjusted Langevin algorithm (MALA), with Langevin dynamics producing a proposal distribution in the latent space as follows:

$$\tilde{\mathbf{z}}_{t+1} = \boldsymbol{z}_t - \alpha \frac{\partial E_\theta(G_\omega(\boldsymbol{z}))}{\partial \boldsymbol{z}} + \epsilon\sqrt{2 * \alpha}, \text{ where } \epsilon \sim \mathcal{N}(0, I_d)$$

Next, the proposed $\tilde{z}_{t+1}$ is accepted or rejected using the Metropolis Hastings algorithm, by computing the acceptance ratio:

$$r = \frac{p(\tilde{\mathbf{z}}_{t+1})q(\mathbf{z}_t|\tilde{\mathbf{z}}_{t+1})}{p(\mathbf{z}_t)q(\tilde{\mathbf{z}}_{t+1}|\mathbf{z}_t)} \tag{9}$$

$$\frac{p(\tilde{\mathbf{z}}_{t+1})}{p(\mathbf{z}_t)} = \exp\big\{ - E_\theta(G_\omega(\tilde{\mathbf{z}}_{t+1})) + E_\theta(G_\omega(\mathbf{z}_t))\big\} \tag{10}$$

$$q(\tilde{\mathbf{z}}_{t+1}|\mathbf{z}_t) \propto \exp\left(\frac{-1}{4\alpha}\big|\big|\tilde{\mathbf{z}}_{t+1} - \mathbf{z}_t + \alpha\frac{\partial E_\theta(G_\omega(\mathbf{z}_t))}{\partial \mathbf{z}_t}\big|\big|_2^2\right) \tag{11}$$

and accepting (setting $\boldsymbol{z}_{t+1} = \tilde{\boldsymbol{z}}_{t+1}$) with probability r.

Algorithm 1 **MEG Training Procedure** Default values: Adam parameters $\alpha = 0.0001, \beta_1 = 0.5, \beta_2 = 0.9; \lambda = 0.1; n_\varphi = 5$

Require: Score penalty coefficient λ, # of θ updates per generator update n_φ , # of training iterations T, Adam hyperparameters α, β_1 and β_2.
Require: Energy function E_θ with parameters θ, entropy statistics function T_ϕ with parameters ϕ, generator function G_ω with parameters ω, minibatch size m,
for $t = 1, ..., T$ **do**
  for $1, ..., n_\varphi$ **do**
    Sample minibatch of real data $\{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(m)}\} \sim P_D$.
    Sample minibatch of latent $\{\mathbf{z}_0^{(1)}, ..., \mathbf{z}_0^{(m)}\} \sim P_z$.
    $\tilde{\mathbf{x}} \leftarrow G_\omega(\mathbf{z})$
    $\mathcal{L}_E \leftarrow \frac{1}{m}\Big[\sum_i^m E_\theta(\mathbf{x}^{(i)}) - \sum_i^m E_\theta(\tilde{\mathbf{x}}^{(i)}) + \lambda \sum_i^m \big|\big|\nabla_{\mathbf{x}^{(i)}} E_\theta(\mathbf{x}^{(i)})\big|\big|^2\Big]$
    $\theta \leftarrow \text{Adam}(\mathcal{L}_E, \theta, \alpha, \beta_1, \beta_2)$
  end for
  Sample minibatch of latent $\mathbf{z} = \{\mathbf{z}^{(1)}, ..., \mathbf{z}^{(m)}\} \sim P_z$.
  Per-dimension shuffle of $\mathbf{z}$, yielding $\{\tilde{\mathbf{z}}^{(1)}, ..., \tilde{\mathbf{z}}^{(m)}\}$.
  $\tilde{\mathbf{x}} \leftarrow G_\omega(\mathbf{z})$
  $\mathcal{L}_H \leftarrow \frac{1}{m}\sum_i^m \Big[\log\sigma(T_\phi(\tilde{\mathbf{x}}^{(i)}, \mathbf{z}^{(i)})) - \log\big(1 - \sigma(T_\phi(\tilde{\mathbf{x}}^{(i)}, \tilde{\mathbf{z}}^{(i)}))\big)\Big]$
  $\mathcal{L}_G \leftarrow \frac{1}{m}\Big[\sum_i^m E_\theta(\tilde{\mathbf{x}}^{(i)})\Big] + \mathcal{L}_H$
  $\omega \leftarrow \text{Adam}(\mathcal{L}_G, \omega, \alpha, \beta_1, \beta_2)$
  $\phi \leftarrow \text{Adam}(\mathcal{L}_H, \phi, \alpha, \beta_1, \beta_2)$
end for

Figure 1. Samples from the beginning, middle and end of the chain performing MCMC sampling in visible space. Initial sample is from the generator (p_G) but degrades as we follow MALA directly in data space. Compare with samples obtained by running the chain in latent space and doing the MH rejection according to the data space energy (Fig. 4). It can be seen that MCMC in data space has poor mixing and gets attracted to spurious modes.

The overall training procedure for MEG is detailed in Algorithm 1 and a visual overview is shown in Figure 2. We found that in order to obtain the best test-time samples, it was best to use the MALA procedure to sample in latent space and then project back in data space using an application of G, similarly to (Bengio et al., 2013). We call the resulting implicit distribution p_{EG}. Note that it is different from p_G (unless the chain has length 1) and it is different from $p_{E_\theta \circ G}$ since that density is in latent space. An additional argument is that latent space sampling of $\boldsymbol{z}$ according to MCMC to approximate $p_{E_\theta \circ G}$ automatically gets rid of spurious modes, which MCMC in x-space doesn't, simply because G does not represent those spurious modes, since the way G is trained highly penalizes spurious modes, while missing modes are not strongly penalized. Although p_G is trained to match p_θ, the match is not perfect, and running a latent-space MCMC on energy function $E_\theta \circ G$ gives better samples, once these z's are transformed to x-space by applying G.

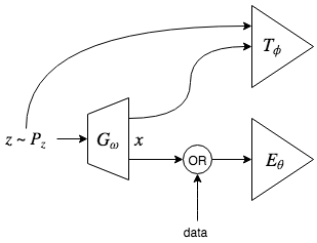

Figure 2. Model overview where G_ω is the Generator network, T_ϕ is the Statistics network used for MI estimation and E_θ is the energy network

5. Rationale

Improvements over (Kim & Bengio, 2016) Our research on learning good energy functions pointed us towards (Kim & Bengio, 2016), which produces a nice theoretical framework to learn a generator that approximates the negative phase MCMC samples of the energy function. However, we noticed that a major issue in (Kim & Bengio, 2016) is that their model used covariance to maximize entropy of the energy function. [2] Entropy maximization using a mutual information estimator is much more robust compared to covariance maximization. However, that alone was not enough and we obtained better performance by using the gradient norm regularizer (3) that helped stabilize the training as well as preventing the energy temperature explosion issue that we faced in replicating (Kim & Bengio, 2016). We also show a successful and efficient MCMC variant exploiting the generator latent space followed by the application of the generator to map to data space, avoiding poor mixing or spurious modes in the visible space.

To understand the benefits of our model, we first visualize the energy densities learnt by our generative model on toy data. Next, we evaluate the efficacy of our entropy maximizer by running discrete mode collapse experiments to verify that we learn all modes and the corresponding mode count (frequency) distribution. Furthermore, we evaluate the performance of our model on sharp image generation, since this is a common failure mode of energy-based models trained with maximum likelihood - which produces blurry samples. We also compare MCMC samples in visible space and our proposed sampling from the latent space of the composed energy function. Finally, we run anomaly detection experiments to test the application of the learnt energy function.

[2] When we tried reproducing the results in that paper even with the help of the authors, we often got unstable results due to the explosion of scale (temperature) of the energies.

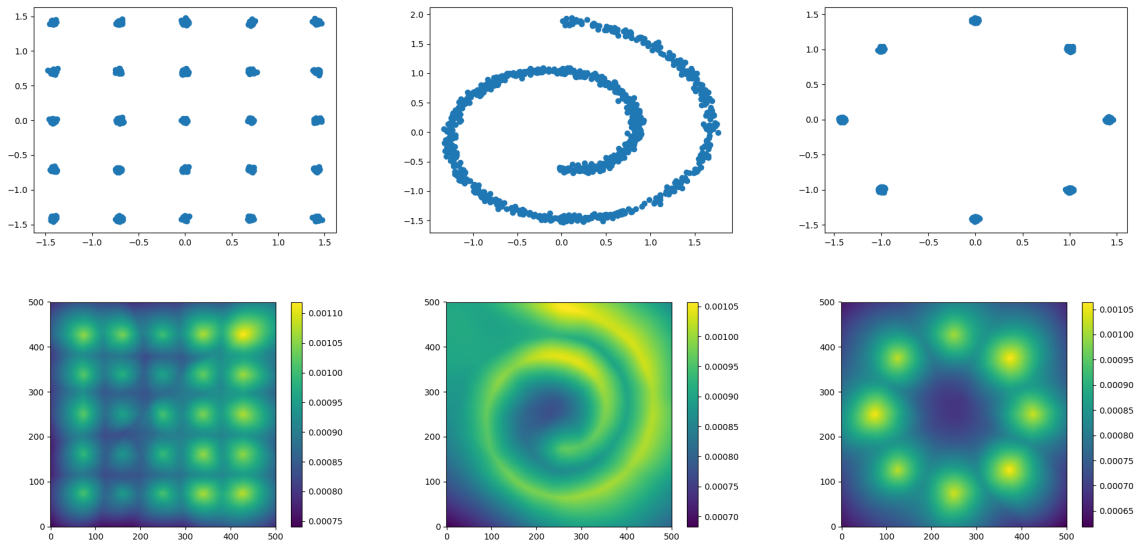

Figure 3. **Top:** Training samples for the 3 toy datasets - 25gaussians, swissroll and 8gaussians. **Bottom:** Corresponding probability density visualizations. Density was estimated using a sample based approximation of the partition function.

6. Experimental Setup

The code for the experiments can be found at https://github.com/ritheshkumar95/energy_based_generative_models/

6.1. Synthetic toy datasets

Generative models trained with maximum likelihood often suffer from the problem of spurious modes and excessive entropy of the trained distribution, where the model incorrectly assigns high probability mass to regions not present in the data manifold. Typical energy-based models such as RBMs suffer from this problem partly because of the poor approximation of the negative phase gradient, as discussed above.

To check if MEG suffers from spurious modes, we train the energy-based model on synthetic 2D datasets (swissroll, 25gaussians and 8gaussians) similar to (Gulrajani et al., 2017) and visualize the energy function. From the probability density plots on Figure 1, we can see that the energy model doesn't suffer from spurious modes and learns a sharp distribution.

6.2. Discrete Mode Collapse Experiment

GANs have been notoriously known to have issues with mode collapse, by which certain modes of the data distribution are not at all represented by the generative model. Similar to the mode dropping issue that occurs in GANs, our generator is prone to mode dropping as well, since it is matched with the energy model's distribution using a reverse KL penalty $D_{\mathrm{KL}}[P_G \mid\mid P_E]$. Although the entropy maximization term attempts to fix this issue by maximizing the entropy of the generator's distribution, it is important to verify this effect experimentally. For this purpose, we follow the same experimental setup as (Metz et al., 2016) and (Srivastava et al., 2017). We train our generative model on the StackedMNIST dataset, which is a synthetic dataset created by stacking MNIST on different channels. The number of modes can be counted using a pretrained MNIST classifier, and the KL divergence can be calculated empirically between the mode count distribution produced by the generative model and true data (assumed to be uniform). From Table 1, we can see that our model naturally covers all the modes in that data, without dropping a single mode. Apart from just representing all the modes of the data distribution, our model also better matches the data distribution as evidenced by the very low KL divergence scores as compared to the baseline WGAN-GP.

We noticed empirically that modeling 10^3 modes was quite trivial for benchmark methods such as WGAN-GP (Gulrajani et al., 2017). Hence, we also try evaluating our model on a new dataset with 10^4 modes (4 stacks). The 4-StackedMNIST was created to have similar statistics to the original 3-StackedMNIST dataset. We randomly sample and fix 128×10^4 images to train the generative model and take 26×10^4 samples for evaluations.

6.3. Perceptual quality of samples

Generative models trained with maximum likelihood have often been found to produce more blurry samples. Our en-

Table 1. Number of captured modes and Kullback-Leibler divergence between the training and samples distributions for ALI (Dumoulin et al., 2016), Unrolled GAN (Metz et al., 2016), VeeGAN (Srivastava et al., 2017), PacGAN (Lin et al., 2017), WGAN-GP (Gulrajani et al., 2017). Numbers except our model and WGAN-GP are borrowed from (Belghazi et al., 2018)

(Max 10^3)	Modes	KL
Unrolled GAN	48.7	4.32
VEEGAN	150.0	2.95
WGAN-GP	959.0	0.7276
PacGAN	1000.0	0.06
Our model	1000.0	0.0313

(Max 10^4)	Modes	KL
WGAN-GP	9538.0	0.9144
Our model	10000.0	0.0480

ergy model is trained with maximum likelihood to match the data distribution and the generator is trained to match the energy model's distribution with a reverse KL penalty. To evaluate if our generator exhibits blurriness issues, we train our model on the standard benchmark 32x32 CIFAR10 dataset for image modeling. We additionally train our models on the 64x64 cropped CelebA - celebrity faces dataset to report qualitative samples from our model. Similar to recent GAN works (Miyato et al., 2018), we report both Inception Score (IS) and Fréchet Inception Distance (FID) scores on the CIFAR10 dataset and compare it with a competitive WGAN-GP baseline.

Table 2. Inception scores and FIDs with unsupervised image generation on CIFAR-10. Reliable 50000 sample estimates were used to compute Inception Score and FID, instead of the previous norm of 5000 samples.

Method	**Inception score**	**FID**
Real data	11.24±.12	7.8
WGAN-GP	6.81 ± .08	30.95
Our model (Generator)	6.49 ± .05	35.02
Our model (MCMC)	**6.94 ± .03**	33.85

From Table 2, we can see that in addition to learning an energy function, MEG trained generative model producing samples comparable to recent adversarial methods such as WGAN-GP (Gulrajani et al., 2017) widely known for producing samples of high perceptual quality. Note that the perceptual quality of the samples improves by using the proposed MCMC sampler.

The authors would also like to mention that the objective of the proposed method is not to beat the best GANs (which do not provide an energy function) but to show that it is possible to have both an energy function and good samples by appropriately fixing issues from (Kim & Bengio, 2016).

6.4. Application to Anomaly Detection

Apart from the usefulness of energy estimates for relative density estimation (up to the normalization constant), energy functions can also be useful to perform unsupervised anomaly detection. Unsupervised anomaly detection is a fundamental problem in machine learning, with critical applications in many areas, such as cyber-security, complex system management, medical care, etc. Density estimation is at the core of anomaly detection since anomalies are data points residing in low probability density areas. We test the efficacy of our energy-based density model for anomaly detection using two popular benchmark datasets: KDDCUP and MNIST.

KDDCUP We first test our generative model on the KDDCUP99 10 percent dataset from the UCI repository (Lichman et al., 2013). Our baseline for this task is Deep Structured Energy-based Model for Anomaly Detection (DSEBM) (Zhai et al., 2016), which trains deep energy models such as Convolutional and Recurrent EBMs using denoising score matching instead of maximum likelihood, for performing anomaly detection. We also report scores on the state of the art DAGMM (Zong et al., 2018), which learns a Gaussian Mixture density model (GMM) over a low dimensional latent space produced by a deep autoencoder. We train our model on the KDD99 data and use the score norm $||\nabla_x E_\theta(x)||_2^2$ as the decision function, similar to (Zhai et al., 2016).

Table 3. Performance on the KDD99 dataset. Values for OC-SVM, DSEBM, Efficient GAN values were obtained from (Zong et al., 2018). Values for our model are derived from 5 runs. For each individual run, the metrics are averaged over the last 10 epochs.

Model	Precision	Recall	F1
Kernel PCA	0.8627	0.6319	0.7352
OC-SVM	0.7457	0.8523	0.7954
DSEBM-e	0.8619	0.6446	0.7399
DAGMM	0.9297	0.9442	0.9369
Our model	**0.9307 ± 0.0146**	**0.9472 ± 0.0153**	**0.9389 ± 0.0148**

From Table 3, we can see that our MEG energy function outperforms the previous SOTA energy-based model (DSEBM) by a large margin (+0.1990 F1 score) and is comparable to the current SOTA model (DAGMM) specifically designed for anomaly detection. Note that DAGMM is an engineered model to perform well on anomaly detection, and doesn't possess the advantages of our energy-based model to draw

Figure 4. **Left:** Samples at the beginning of the chain (i.e. simply from the ordinary generator, $z \sim N(0, I)$). **Right:** Generated samples after 100 iterations of MCMC using the MALA sampler. We see how the chain is smoothly walking on the image manifold and changing semantically meaningful and coherent aspects of the images.

sharp samples.

MNIST Next we evaluate our generative model on anomaly detection of high dimensional image data. We follow the same experiment setup as (Zenati et al., 2018) and make each digit class an anomaly and treat the remaining 9 digits as normal examples. We also use the area under the precision-recall curve (AUPRC) as the metric to compare models.

Table 4. Performance on the unsupervised anomaly detection task on MNIST measured by area under precision recall curve. Numbers except ours are obtained from (Zenati et al., 2018). Results for our model are averaged over last 10 epochs to account for the variance in scores.

Heldout Digit	VAE	MEG	BiGAN-σ
1	0.063	0.281 ± 0.035	0.287 ± 0.023
4	0.337	0.401 ± 0.061	0.443 ± 0.029
5	0.325	0.402 ± 0.062	0.514 ± 0.029
7	0.148	0.29 ± 0.040	0.347 ± 0.017
9	0.104	0.342 ± 0.034	0.307 ± 0.028

From Table 4, it can be seen that our energy model outperforms VAEs for outlier detection and is comparable to the SOTA BiGAN-based anomaly detection methods for this dataset (Zenati et al., 2018) which train bidirectional GANs to learn both an encoder and decoder (generator) simultaneously and use a combination of the reconstruction error in output space as well as the discriminator's cross entropy loss as the decision function.

6.5. MCMC Sampling

To show that the Metropolis-Adjusted Langevin Algorithm (MALA) performed in latent space produces good samples in observed space, we attach samples from the beginning (with z sampled from a Gaussian) and end of the chain for visual inspection. From the attached samples, it can be seen that the MCMC sampler appears to perform a smooth walk on the image manifold, with the initial and final images only differing in a few latent attributes such as hairstyle, background color, face orientation, etc. Note that the MALA sampler run on E_θ in visible space 1 did not work well and tends to get attracted to spurious modes (which G eliminates, hence the advantage of the proposed p_{EG} sampling scheme).

7. Conclusions

We proposed MEG, an energy-based generative model that produces energy estimates using an energy model and a generator that produces fast approximate samples. This takes advantage of novel methods to maximize the entropy at the output of the generator using a GAN-like technique. We have shown that our energy model learns good energy estimates using visualizations in toy 2D data and through performance in unsupervised anomaly detection. We have

also shown that our generator produces samples of high perceptual quality by measuring Inception and Fréchet scores and shown that MEG is robust to the respective weaknesses of GAN models (mode dropping) and maximum-likelihood energy-based models (spurious modes). We found that running an MCMC in latent space rather than in data space (by composing the generator and the data-space energy to obtain a latent-space energy) works substantially better than running the MCMC in data-space.

8. Acknowledgements

The authors acknowledge the funding provided by NSERC, Canada AI Chairs, Samsung, Microsoft, Google and Facebook. We also thank NVIDIA for donating a DGX-1 computer used for certain experiments in this work.

References

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Belghazi, I., Baratin, A., Rajeswar, S., Ozair, S., Bengio, Y., Courville, A., and Hjelm, R. D. Mine: mutual information neural estimation. *arXiv preprint arXiv:1801.04062, ICML'2018*, 2018.

Bell, A. J. and Sejnowski, T. J. An information-maximization approach to blind separation and blind deconvolution. *Neural computation*, 7(6):1129–1159, 1995.

Bengio, Y. *Learning deep architectures for AI*. Now Publishers, 2009.

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. Greedy layer-wise training of deep networks. In *NIPS'2006*, 2007.

Bengio, Y., Mesnil, G., Dauphin, Y., and Rifai, S. Better mixing via deep representations. In *ICML'2013*, 2013.

Brakel, P. and Bengio, Y. Learning independent features with adversarial nets for non-linear ica. *arXiv preprint arXiv:1710.05050*, 2017.

Dai, Z., Almahairi, A., Bachman, P., Hovy, E., and Courville, A. Calibrating energy-based generative adversarial networks. *arXiv preprint arXiv:1702.01691*, 2017.

Dumoulin, V., Belghazi, I., Poole, B., Mastropietro, O., Lamb, A., Arjovsky, M., and Courville, A. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

Goodfellow, I. On distinguishability criteria for estimating generative models. *arXiv preprint arXiv:1412.6515*, 2014.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks. In *NIPS'2014*, 2014.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. Improved training of Wasserstein GANs. In *NIPS'2017*, pp. 5767–5777, 2017.

Gutmann, M. and Hyvarinen, A. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS'2010*, 2010.

Hinton, G. E. Training products of experts by minimizing contrastive divergence. Technical Report GCNU TR 2000-004, Gatsby Unit, University College London, 2000.

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

Hjelm, R. D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Trischler, A., and Bengio, Y. Learning deep representations by mutual information estimation and maximization. *arXiv:1808.06670*, 2018.

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

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

Larochelle, H., Bengio, Y., Louradour, J., and Lamblin, P. Exploring strategies for training deep neural networks. *J. Machine Learning Res.*, 10:1–40, 2009.

LeCun, Y., Chopra, S., Hadsell, R., Ranzato, M.-A., and Huang, F.-J. A tutorial on energy-based learning. In Bakir, G., Hofman, T., Scholkopf, B., Smola, A., and Taskar, B. (eds.), *Predicting Structured Data*, pp. 191–246. MIT Press, 2006.

Lichman, M. et al. Uci machine learning repository, 2013.

Lin, Z., Khetan, A., Fanti, G., and Oh, S. Pacgan: The power of two samples in generative adversarial networks. *arXiv preprint arXiv:1712.04086*, 2017.

Linsker, R. Self-organization in a perceptual network. *Computer*, 21(3):105–117, 1988.

Metz, L., Poole, B., Pfau, D., and Sohl-Dickstein, J. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

Nowozin, S., Cseke, B., and Tomioka, R. f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in Neural Information Processing Systems*, pp. 271–279, 2016.

Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Srivastava, A., Valkoz, L., Russell, C., Gutmann, M. U., and Sutton, C. Veegan: Reducing mode collapse in gans using implicit variational learning. In *Advances in Neural Information Processing Systems*, pp. 3308–3318, 2017.

Tieleman, T. Training restricted Boltzmann machines using approximations to the likelihood gradient. In *ICML'2008*, pp. 1064–1071, 2008.

van den Oord, A., Kalchbrenner, N., and Kavukcuoglu, K. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016.

Younes, L. On the convergence of Markovian stochastic algorithms with rapidly decreasing ergodicity rates. In *Stochastics and Stochastics Models*, pp. 177–228, 1998.

Zenati, H., Foo, C. S., Lecouat, B., Manek, G., and Chandrasekhar, V. R. Efficient gan-based anomaly detection. *arXiv preprint arXiv:1802.06222*, 2018.

Zhai, S., Cheng, Y., Lu, W., and Zhang, Z. Deep structured energy based models for anomaly detection. *arXiv preprint arXiv:1605.07717*, 2016.

Zhao, J., Mathieu, M., and LeCun, Y. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126*, 2016.

Zong, B., Song, Q., Min, M. R., Cheng, W., Lumezanu, C., Cho, D., and Chen, H. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. 2018.