Implicit Generation and Generalization with Energy-Based Models

Yilun Du
OpenAI

Igor Mordatch
OpenAI

Abstract

Energy based models (EBMs) are appealing due to their generality and simplicity in likelihood modeling, but have been traditionally difficult to train. We present techniques to scale EBM training through a MCMC framework to modern architectures. We show that MCMC sampling on EBMs generate significantly better samples than other likelihood models (with significantly lower generation cost and parameters), so much so that they are competitive with GANs. We show that EBMs are good likelihood models, able to both reliably restore test CIFAR-10 images and interconvert between classes of CIFAR-10 images. Finally show that EBMs generalize well. On CIFAR10, we achieve better out of distribution generalization than other state of the art generative models (such as assigning high likelihood to CIFAR-10 images than SVHN images). For time series modeling, EBMs generalize much better for long term time series prediction than corresponding feed-forward networks. Compositionaly, we find EBMs generalize to effectively generate samples when jointly conditioned on independent conditional EBMs for different latents.

1 Introduction

Two fundamental problems with deep learning are data efficiency and out of distribution generalization. Generative models capture world knowledge and enable faster learning by fine-tuning existing features. At the same time, generative modeling helps prevent catastrophical failure in out of distribution cases.

Generative modeling has seen a flux of interest. Many approaches reply on directly maximizing the likelihood. Modeling a correlated high dimensional data distribution is difficult. Auto-regressive models [Van Oord et al., 2016, Graves, 2013] solve this by completely factorizing the underlying distribution, but such an approach leads to accumulation of error and dispersal of underlying structural information. Other approaches such as the variational auto-encoder [Kingma and Welling, 2014] or flow based models [Dinh et al., 2014, Kingma and Dhariwal, 2018] rely on a factorized prior distribution to simplify likelihood estimation. Flow models require invertible Jacobian transformations, which can limit model capacity and have difficulty fitting discontinuous data. Such approximations have prevented likelihood models from generating high quality images in diverse domains. In contrast, approaches based on generative adversarial networks [Goodfellow et al., 2014] put no constraints on the latent space and have generated high-quality images but do not cover the entire data distribution.

Energy based models (EBMs) are flexible likelihood model with no latent constraints [LeCun et al., 2006]. EBMs received attention in the past [Hinton et al., 2012, Dayan et al., 1995] but have not seen wide adoption due to an expensive negative sampling phase.

Third workshop on Bayesian Deep Learning (NeurIPS 2018), Montréal, Canada.

Figure 1: Images of Lengevin Dynamic Sampling (gradient based) on a class condition EBMs initialized with random image and image of different class.

We propose methods to scale up EBMs to modern day architectures. We find that these EBMs are able to generate significantly better samples than other likelihood models and are competitive with GANs on CIFAR10 (inception score) in image quality and better than corresponding feedforward networks in time series modeling.

We find that our EBMs are able to generate significantly better samples (Figure 1 shows images of the sampling process) than other likelihood models, so much so that they are competitive with GANs. Through several image completion experiments and likelihood evaluation, we find that the sample quality does not come at the expense of mode collapse. Finally, we show that EBMs generalize well. On images, EBMs exhibit better distribution generalization [Hendrycks and Gimpel, 2016] than other state of the art likelihood models, such as being able to assign higher log likelihood to CIFAR10 test images than SVHN images. For time series modeling, EBMs generalize much better for long term time series prediction than corresponding feed-forward networks. Compositionally, we find EBMs generalize to effectively generate samples when jointly conditioned on independent conditional EBMs for separate latents.

2 Related Work

Energy based models have seen large amounts of attention in the past [LeCun et al., 2006, Hinton et al., 2012]. Previous methods have also relied on MCMC training to sample from the partition function, but have primarily relied on Gibb's Sampling on older architectures [Nair and Hinton, 2010]. We instead use Langevin Dynamics (also used in [Mnih and Hinton]) or MPPI([Williams et al., 2017]) to more efficiently sample negative samples on modern architectures.

While there are been other recent works that have focused on training EBMs, sampling is done through a sampling network [Zhao et al., 2016, Haarnoja et al., 2017]. Our contrast, our sampling is done entirely through an MCMC approximation of the original distribution. Since we use our original network to sample, we are able to explore all modes in the probability density. Furthermore, this allows us to easily use our sampling procedure down-stream to restore and cross map images in the energy landscape.

We show a connection of energy based training to GANS which has also been shown in [Finn et al., 2016, Zhao et al., 2016, Kim and Bengio, 2016]. Finn et al. [2016] show a direct connection between training a GAN and energy functions, using a separate proposal distribution $q(x)$ to estimate the partition function. Zhao et al. [2016], Kim and Bengio [2016] and many related works use adversarial training on a separate $q(x)$ to provide fast estimation of the partition function. Our work is separate from these models as we use a MCMC approximation of the original function to estimate the partition function and use this MCMC approximation as our generator. Since our "generator" is then dependent on our original function, the generator adapts implicitly while only training our energy function (discriminator). This then removes the need to train the generator, which combined with the fact the model itself has modes of probability at all training data points, reduces the likelihood of mode collapse. Our derivation for EBMs further shows that maximizing likelihood under an optimal "generator" corresponds exactly to the Wassterstein GAN criterion [Arjovsky et al., 2017].

3 Scaling EBM Training

In the section, we formulate our method for training EBMs. We outline our likelihood objective and sampling distributions. We then detail architectural changes and sample tricks allowing better overall sampling. Finally, we provide our overall loss and show a connection GAN based training.

3.1 Likelihood Objective

Our overall algorithm for training algorithm for EBMs follows the contrastive divergence algorithm [Hinton, 2002]. Given an energy function $E(x;\theta) \in \mathbb{R}$, represented as a a neural network *, we model the probability distribution as $p(x)$ through the Gibb's distribution.

$$p(x) = \frac{e^{-E(x;\theta)}}{Z(\theta)}; \quad Z(\theta) = \int e^{-E(y;\theta)} dy$$

Given data distribution $p_d(x)$, we seek to minimize negative likelihood given by

$$\mathbb{E}_{x\sim p_d(x)}\left[E(x;\theta) + \log\left(\int e^{-E(y;\theta)}dy\right)\right] = \mathbb{E}_{x\sim p_d(x)}\left[E(x;\theta) + \log(\mathbb{E}_{y\sim q(x)}[e^{-E(y;\theta)}/q(y)])\right] \tag{1}$$

where $q(x)$ is a proposal distribution which we choose to be a finite step MCMC approximation of $p(x)$. The exact likelihood $q(y)$ is difficult to calculate. We choose to use one of either two approximations: either that $q(y)$ has uniform probability or if $q(y)$ matches the energy distribution $p(y)$. If we assume that all $q(y)$ are equal, we have an *approximate* negative log likelihood objective of

$$\min_\theta \mathbb{E}_{x\sim p_d(x)}\left[E(x;\theta) + \log(\mathbb{E}_{y\sim q(x)}[e^{-E(y;\theta)}])\right] \tag{2}$$

Alternatively, if we assume that $q(y) = p(y)$ then we obtain the **exact** negative log likelihood objective below from Equation 1 (derivation in Section 9.1)

$$\min_\theta \mathbb{E}_{x\sim p_d(x)}\left[E(x;\theta) - \mathbb{E}_{y\sim q(x)}[E(y;\theta)]\right]. \tag{3}$$

Empirically, we find that $q(y)$ appears to have almost equal probability distribution to $p_d(x)$ (Figure 2), indicating that its likely that $q(y) = p(y)$, indicating a preference of Equation 3. However, during training, we found that empirically Equation 3 was slightly less stable than Equation 2, so we choose to use Equation 2 for our training procedures.

3.2 Proposal Distributions

To we use a proposal distribution $q(x)$ to gather samples for the likelihood objective. In contrast to recent works, our proposal distributions are MCMC approximations of the original distribution instead of a separate network. We either use Langevin dynamics (LD) [Welling and Teh, 2011]

$$\tilde{x}^q = \tilde{x}^k, \quad \tilde{x}^k = \tilde{x}^{k-1} - \lambda(\nabla_{x^{k-1}} E(\tilde{x}^{k-1}) + \omega^k; \theta), \quad \omega \sim N(0, \sigma) \tag{4}$$

or MPPI based MCMC sampling [Williams et al., 2017]

$$\tilde{x}^k = \sum_i w_i x_i^k, \quad x_i^k \sim N(0,\sigma) + x^{k-1}, \quad w_i = \left(\frac{e^{-E(x_i^k;\theta)}}{\sum_j e^{-E(x_i^j;\theta)}}.\right) \tag{5}$$

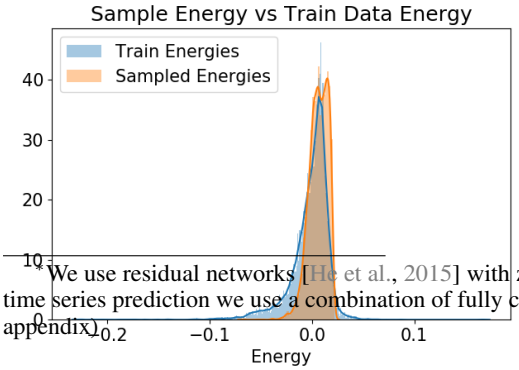


Figure 2: Relative energy of points sampled from $q(x)$ compared to CIFAR10 train data points. We find that $q(x)$ exhibits a relatively similar distribution to $p_d(x)$.

For stability, we further clip gradients in LD sampling. We find that either MCMC proposals lead to good training performance, with LD scaling better on high dimensional data, but MPPI sampling working better on lower dimensional data. The gradient descent direction given by an energy function in LD allows quick mode generation, but may also make it difficult to find all nearby modes, which MPPI sampling allows. We demonstrate that both methods perform well

*We use residual networks [He et al., 2015] with zero initialization [Anonymous, 2019b] for images. For time series prediction we use a combination of fully connected, self-attention and 1D convolutions layers (see appendix)

on time series data but for images, only LD was viable.

At the start of training, we initialize MCMC chains with uniform random noise. To improve mixing of MCMC chains, as training progresses, we maintain a replay buffer of past generated samples, re-initializing 5% of samples with random noise. This is similar to PCD in [Tieleman, 2008], but a replay buffer has the added benefit of discouraging models from exhibiting cyclical behavior. Furthermore, a replay buffer encourages models to not have spurious modes at all stages of sampling, and we found replay buffers crucial for sampled images to be reasonable, if the sampling procedure is significantly longer than during training.

3.3 Model Constraints

Arbitrary energy models can have sharp changes in gradients that make sampling with LD very difficult. We find that simply constraining the Lipschitz constant remove many of these problems and allow most architectures blocks (residual, self attention,) to be samplable and thus trainable, as long as activations were kept piecewise linear without activation normalization. To constrain the Lipschitz constant, we follow the method of [Miyato et al., 2018] and add spectral normalization to all layers of the model.

Constraining the Lipschitz constant in models can significantly reduce the capacity of models and there may exist blocks that are not easily samplable even with spectral normalization. A more general solution we found was to then add an additional loss to minimize KL distance between proposal distribution and the original distribution (with the drawback of computational expense).

Specifically, we minimize the KL objective

$$\min_{\theta} \mathrm{KL}(q(x;\theta), p(x; \texttt{stop\_gradient}(\theta))) = \min_{\theta} -\mathbb{E}_{x_i \sim q(x;\theta)}[E(x_i; \texttt{stop\_gradient}(\theta))] \quad (6)$$

. We note that both MCMC sampling distributions are differentiable, allowing us to change our model's landscape to be more samplable. We choose to ignore the first entropy term of the KL divergence, as it is difficult to evaluate the likelihood of $q(x)$. Methods such as [Liu et al., 2017] may be helpful in simultaneously minimizing both terms.

In our tasks, we found that constrained Lipschitz residual networks were sufficiently expressive to model images. In cases of particle dynamics, we found the Lipschitz constant constraint overly restrictive and chose to add the above loss to impose sampleability.

3.4 Loss Functions

For training EBMs we use three different loss functions, L_{ML}, a maximum likelihood loss described in Section 3.1, L_{KL}, an optional sampling loss described in Section 3.3 and a regularization loss L_{Reg}. Our overall loss function is given by

$$L_{\text{total}} = L_{\text{ML}} + L_{\text{KL}} + L_{\text{Reg}}$$

For L_{ML} we approximate Equation 2 over N points, where x_i^- and sampled from proposal distribution $q(x;\theta)$ and x_i^+ are real data points to obtain the following loss

$$L_{\text{ML}} = \frac{1}{N}\sum_{i=0}^{N} E(x_i^+;\theta) + \log\left(\sum_{i=0}^{N} e^{-E(x_i^-;\theta}\right). \quad (7)$$

In practice, we found that gradient of the log term $\sum_{i=0}^{N}(\frac{e^{-E_\theta(x_i^-)}}{\sum_i e^{-E_\theta(x_i^-)}})E(x_i^-)$ to be numerically unstable due to close to zero denominators. Therefore, we rewrite our objective in an equivalent form which removes this issue

$$L_{\text{ML}} = \frac{1}{N}\sum_{i=0}^{N} E_\theta(x_i^+) - \sum_{i=0}^{N} -\texttt{stop\_gradient}\left(\frac{e^{-E_\theta(x_i^-)}}{\sum_i e^{-E_\theta(x_i^-)} + \epsilon}\right) E(x_i^-). \tag{8}$$

When using a sampling loss (only on time series data), we approximate Equation 6 over N points to get

$$L_{\text{KL}} = \frac{1}{N}\sum_{x_i \sim q(x)} E_{\texttt{stop\_gradient}(\theta)}(q(x_i)). \tag{9}$$

Finally, we add a regularization loss to prevent energy explosion during training, as above losses only enforce relative energy difference between points and not absolute energy magnitude

$$L_{\text{Reg}} = \frac{1}{N}\sum_i E_\theta(x_i^+)^2 + E_\theta(x_i^-)^2. \tag{10}$$

We include a detail of our algorithm in Algorithm 1

3.5 Relationship to GAN training

We note that our likelihood objective is similar to GAN training objective training, where the generator $G(x)$ is our MCMC proposal distribution $q(x)$ and the discriminator is the energy function $p(x)$. In fact, when setting $N = 1$ in Equation 7 or assuming $q(x) = p(x)$ Equation 3, we precisely obtain the Wasserstein objective [Arjovsky et al., 2017]. Furthermore, in cases where proposal distribution is LD sampling on residual network based EBM, we note each step of sampling is operationally equivalent to a forward pass through a generator, since the gradient of a convolution is a deconvolution.

However an important difference between our likelihood training and GAN based training is that our "generator" is **implicitly** a function of our "discriminator" (and in the infinite limit of number of the steps the "discriminator"). As a result, the "generator" is able to co-adapt with training of the discriminator and there is no need to explicitly train the "generator". This reduces the likelihood of over-fitting of the "generator" and makes it exhibit the same modes that a discriminator exhibits. Furthermore, this allows our framework to work well on discrete values in which training generators is difficult.

This connections may explain one reason in which EBMs are able to sharper samples than other likelihood models. However, we note that unlike in GANs, our generation procedure doesn't appear to exhibit significant mode collapse as seen in Figure 5.

4 Images Modeling

We measure EBM's ability to model complex distribution on both class-conditional and unconditional CIFAR10 datasets. Our model is based on the ResNet architecture (using conditional gains and biases per class) with details in Section 9.5. Our models have around 7-20 million parameters, comparatively smaller than other state of the art models. We preprocess images to be between 0 and 1 with details in Section 9.6. We evaluate EBMs ability to generate images, show that it constructs a good likelihood model of the underlying distribution, measure representation learning, and show that exhibits good out of distribution generalization

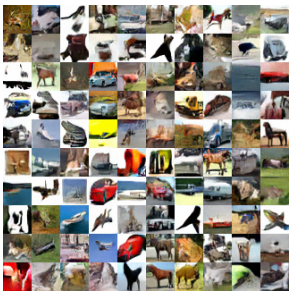
4.1 Image Generation

We provide unconditional generated images in Figure 3 and conditional generated images in Figure 4a. In Figure 3, we see that compared to Glow, are model is able to make much ore object like images. We show in Figure 13 that our generated images are not merely copies of images in the dataset.

We evaluate image quality of EBMs with Inception score [Salimans et al., 2016] and report comparison with other models Table 4b. As evidenced by qualitative examples, we find higher inception scores than PixelCNN (4.60) and is similar to performance to DCGAN (6.4) [Radford et al., 2016] in unconditional generation (6.43). One issue we found during evaluation is that LD sampling takes large amounts of time to explore all modes from random noise at test time, a problem mitigated by a replay buffer during training time. To mimic a replay buffers ability to increase sample diversity, we consider alternate sampling from the last 10 snapshots of a model. Under this scheme, we are able to improve scores to 6.79 and believe that additional improvements can be obtained by more explicit

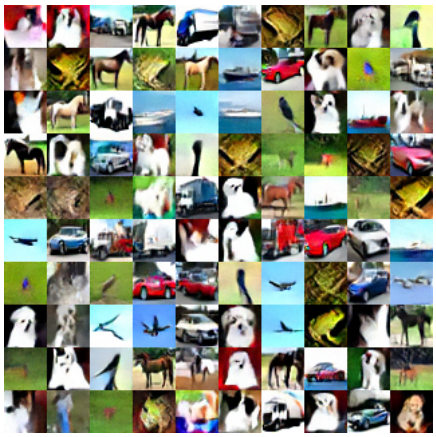
(a) GLOW [Kingma and Dhariwal, 2018] samples

(b) Unconditional EBM samples



(c) Historical ensemble(10) EBM unconditional samples

Figure 3: Illustrations of image generation from GLOW as compared to our EBM models. Our models are able to more accurately generate objects.



(a) conditional CIFAR10 EBM samples

Model	Inception Score
Unconditional CIFAR10	
PixelCNN	4.60
DCGAN	6.40
Ours (single)	6.43
Ours (10 historical ensemble)	**6.79**
Conditional CIFAR10	
Improved GAN	8.09
Ours	8.52
Spectral Normalization GAN	**8.59**

(b) Table of Inception Scores

exploration. We note that numbers are still lower than current state of the art unconditional GAN models, likely due to reduced mode exploration.

For conditional generation, we find that our inception score of 8.52 is higher than 8.09 in [Salimans et al., 2016] and is close to 8.59 in [Miyato et al., 2018]. Our conditional CIFAR10 scores are very similar to state of the art GAN model scores. We believe a large reason for the increase in competitiveness of conditional EBMs relative to unconditional EBMs is increased mode exploration during evaluation / training time. With conditional EBMs, during test time evaluation, we able to able to initialize generation of class images from images initially generated from other classes, allowing more mode exploration.

4.2 Likelihood Evaluation

Quantitative Evaluation We found it difficult to estimate the partition function of EBMs to measure exact likelihood [†]. However, relative probability of data points can be evaluated by computing energies of points. We found that our unconditional model had average energies of -0.00169 ± 0.0196 on the train dataset and 0.001454 ± 0.0176 on the test dataset. For conditional model, we found energies of 0.00198 ± 0.0369 on the train dataset and 0.00751 ± 0.0374 on the test dataset. The small mean difference relative to individual standard deviation indicates EBMs assigns close likelihoods on train and test sets are not over-fitting to training images.

Image Restoration While unable to evaluate exact likelihood, we can measure the relative likelihood modeling of models through image decorruption on test images. If a model is a able to reliably

[†]After training, we founding that using AIS [Neal, 2001] with HMC transitions took too long to explore modes.

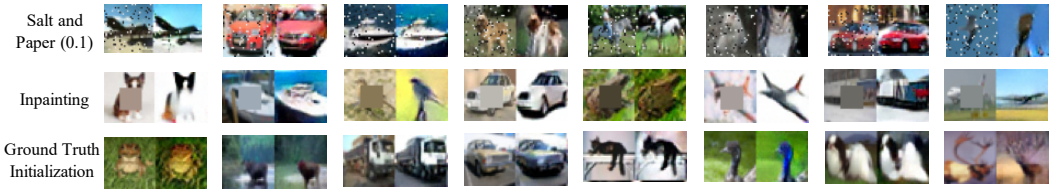


Figure 5: Conditional EBM image restoration on images in the **test** set through MCMC. The right column shows failure (approx. 10% objects change with ground truth initialization and 30% of objects change in salt/pepper corruption or in-painting. Right column shows worst case of change.)

Figure 6: Illustration of cross-class mapping using MCMC on a conditional EBM. The EBM is conditioned on a particular class but is initialized with an image from a separate class(left). Target classes of ship, car, ship. Additional images in Figure 12

restore and maintain test images, its likely that we have a good likelihood model of overall data. In Figure 5, we find that if we initialize sampling with images from the test set, images do not move significantly, indicating modes of probability at all test images. In Figure 5, we also test models ability to inpaint and decorrupt images. We found in a large majority of cases, we are also able to reliably decorrupt images, indicating relatively little mode collapse. In comparison, GANs have been shown to miss many modes of data and cannot reliably reconstruct many different test images [Yeh et al.].

As an additional measure of likelihood modeling, we initialize conditional models with images from images from another class. We find in Figure 6 that energy models are still able to reliably convert these images to images of the target class, indicating good likelihood modeling and generalization.

4.3 Representation Learning

We further investigate representation learning in EBMs, which we measure by fine-tuning energy models to a supervised classification task. We remove the last linear layer of our model and replace it with a classification layer after pretraining in an unsupervised way. During training, we backpropogate through all weights and get results found in Table 1. We find EBMs learns representations that allow better generalization on CIFAR10. We believe even larger gains may be achieved by pre-training on larger dataset or joint training.

4.4 Out-of-Distribution Generalization

An important evaluation metric for generative modeling is to measure how well models generalize to out-of-distribution(OOD) images. If a generative model has learned a good probability data distribution, the model should be able to assign lower probability to data from all other disjoint distributions. Curiously, however, as found in [Anonymous, 2019a], it appears current likelihood models, such VAE, PixelCNN, and Glow models, are unable to distinguish data from disjoint

#	Baseline	FT	Baseline + DA	FT + DA
Accuracy	83.6	86.5	89.7	90.2

Table 1: Test Accuracy on CIFAR10 with or without finetuning (FT) with or without data augmentation (DA). We using horizontal flip and random crop data augmentations. Energy based fine-tuning allow better generalization.

Figure 7: Illustration of images from each of the out of distribution dataset.

Model	SVHN	Textures	Monochrome Uniform	Uniform	CIFAR10 Interpolation	Average
PixelCNN++	0.32	0.33	0.0	1.0	**0.71**	0.47
Glow	0.24	0.27	0.0	1.0	0.59	0.42
EBM (ours)	**0.63**	**0.48**	**0.30**	**1.0**	0.70	**0.62**

Table 2: AUROC scores of out of distribution classification on different datasets, only our model gets better than chance classification.

distribution, and actual assigns higher likelihood to certain OOD images (which ameliorated somewhat in [Hendrycks et al., 2018]).

Similar to [Hendrycks and Gimpel, 2016], we propose a OOD metric where we take generative models trained on CIFAR10 and evaluate the AUROC score for classifying CIFAR10 test images compared to OOD images using log probability. We choose to evaluate on SVHN, Textures [Cimpoi et al., 2014], monochromatic uniform noise(all image pixels are the same value), uniform noise and interpolations of separate CIFAR10 images. We choose the SVHN dataset for comparison to previous works, Textures to test memorization of textures, monochromatic uniform noise to test for memorization of smoothness, uniform noise as a sanity test for likelihood modeling, and image CIFAR10 interpolation (where we mix two different CIFAR10 images) as a test of memorization of low level image statistics. We provide illustration of out-of-distribution images in Figure 7.

As seen in Table 2, EBMs perform better out-of-distribution than other models. We provide histograms of relative likelihoods for SVHN in Figure 8 which is also discussed in [Anonymous, 2019a, Hendrycks et al., 2018]. We believe that reason for better generalization is two-fold. First, we believe that EBMs have flexible structure allowing global context when estimating probability without imposing constraints on latents. In contrast, auto-regressive models model likelihood sequentially, making global coherence difficult. In a different vein, flow based models must apply continuous transformations onto a continuous connected probability distribution which makes it very difficult to model disconnected modes, consequently making it likely that large amounts of probability are wasted at connections between modes. Second, we believe that energy models have a negative sampling procedure to estimate the partition function, which allows the model to exhibit less bad local minima. However, we note that there is till much work that can be done to improve the out-of-distribution robustness of generative models.

5 Time Series Prediction

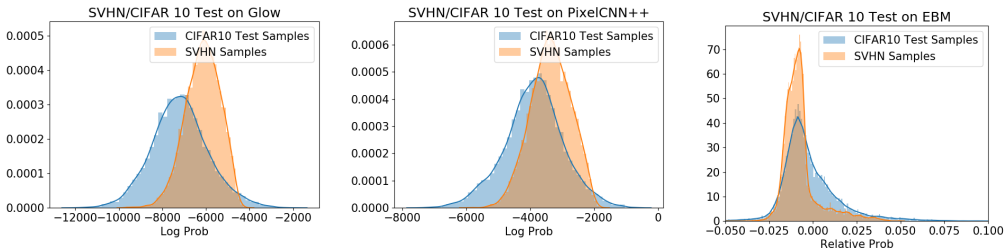


Figure 8: Histogram of relative likelihoods for SVHN images vs CIFAR10 Test Images for Glow, PixelCNN++ and EBM models.

Figure 10: Images generated by joint distribution of 4 conditional EBMs trained independently on scale, position, rotation and shape (left) with associated ground truth rendering (right). Despite never being trained to sample with other conditional models, EBMs are able to compositionaly combine.

To demonstrate the generality of our technique, we also explore the ability of energy functions to model future predictions on time series. We consider two time series tasks; either predicting the trajectory of a single moving ball.

5.1 Particle Dynamics

To test particle dynamics modeling, we simulate one moving ball with wall collisions, drag and friction. We train models to predict the next ball positions given the past 3 positions using 4500 training trajectories with 500 time-steps and evaluate MSE of future state predictions on 500 test trajectories. We compare training an energy model with directly doing a feed-forward prediction of state by modifying the last layer to predict the next state. We use the architecture given in Figure 15c and use either MPPI of LD to sample from. Details can be found in Section 9.6. The train feed-forward prediction on MSE error.

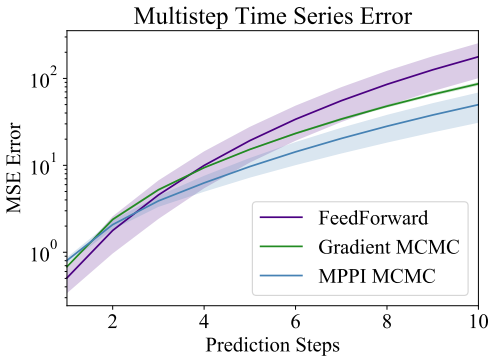


Figure 9: Multistep time series prediction MSE errors (**log** scale). EBMs show out of distribution generalization by reduced long term rollout error.

We present results of multistep time series predictions at test time in Figure 9. When using model roll-outs, we find that energy functions have significantly lower error for multistep prediction despite higher initial error, compared to feed forward networks. We believe this is due to EBMs being able to generalize better in out of distribution situations that occur after a couple model rollouts. Interestingly, we find the MPPI based sampling methods lead to better generalization than LD based sampling methods, probably partially due to better mode exploration during training time as the gradient may be biased to certain minima.

6 Combinatorial Generalization

To further evaluate generalization of EBMs, we consider sampling from the joint distribution of several separately trained conditional EBMs on different latents. Due to the functional form of EBMs, sampling from the joint distribution using LD is equivalent to taking gradient descent on each respective conditional EBM with added noise. Sampling from the joint distribution tests generalization by leading to constraints/mode exploration likely not seen during training. We evaluate on the DSprites dataset [Higgins et al., 2017], which consists images of a single object varied by scale, position, rotation, and shape.

Latent Conditioning We found that it to we could effectively model conditional latent with conditional gains and biases following each convolution. Latents can be either continuous or discrete and are projected to the size of each gain or bias. We found latents of scale, position and rotation were well. The latent of shape was difficult to learn, and we found that even our unconditional models were not able to reliably generate different shapes which is also the case in [Higgins et al., 2017], perhaps due to the combinatorical explosion of different shapes at different scales. We further found by incorporating latent sampling into proposal sampling during training, latents could also be effectively inferred from images at test time.

Joint Conditioning In Figure 10, we provide generated images where we condition on 4 separate EBMs trained on conditioning of scale, position, rotation and shape on the entire DSprites dataset. We find that we are able to effectively sample from the joint distribution without significant loss in

sample quality (all factors except shape appeared to be preserved; the original conditional shape model is also unable to generate definitive shapes). We believe a unique advantage of EBMs is the ability for sampling cost to scale linearly with the number separate conditional distribution as opposed to exponentially in the case of rejection sampling.

7 Conclusion

We have presented techniques to scale up EBM training. We show that EBMs are class of generative models worth exploring as EBMs are able to generate much better samples than other state-of-the-art models while simultaneously maximizing likelihood and maintaining modes of probability on test images. Furthermore, EBMs generalize well, achieving better out-of-distribution generalization than other likelihood models, compositionally combining at test time seperately condition energy functions, and achieving significantly reduced long term trajectory roll-out error.

We believe that the EBM formulation of generative modeling is flexible, able to accommodate any existing modern architecture and relatively fast, as models are smaller and image generation occurs at once. Our current formulation of EBMs undergoes generation by inference over input images, but we believe an identical method can be done to infer latents, allowing EBMs to simultaneously generate, maximize likelihood, and classify images.

8 Acknowledgements

We would like to thank Ilya Sutskever, Alec Radford, Prafulla Dhariwal, Dan Hendrycks, Johannes Otterbach and everyone at OpenAI for helpful discussions.

References

Anonymous. Do deep generative models know what they don't know? In *Submitted to International Conference on Learning Representations*, 2019a. URL https://openreview.net/forum?id=H1xwNhCcYm. under review. 7, 8

Anonymous. The unreasonable effectiveness of (zero) initialization in deep residual learning. In *Submitted to International Conference on Learning Representations*, 2019b. URL https://openreview.net/forum?id=H1gsz30cKX. under review. 3, 14

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017. 2, 5

M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014. 8

Peter Dayan, Geoffrey E Hinton, Radford M Neal, and Richard S Zemel. The helmholtz machine. *Neural Comput.*, 7(5):889–904, 1995. 1

Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014. 1

Chelsea Finn, Paul Christiano, Pieter Abbeel, and Sergey Levine. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. In *NIPS Workshop*, 2016. 2

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014. 1

Alex Graves. Generating sequences with recurrent neural networks. *arXiv:1308.0850*, 2013. 1

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. *arXiv preprint arXiv:1702.08165*, 2017. 2

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2015. 3

Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016. 2, 8

Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. *arXiv preprint*, 2018. 8

Irina Higgins, Loic Matthey, Arka Pal, Christopher P Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. Beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017. 9

Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural Comput.*, 14(8): 1771–1800, 2002. 3

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv:1207.0580*, 2012. 1, 2

Taesup Kim and Yoshua Bengio. Deep directed generative models with energy-based probability estimation. *arXiv preprint arXiv:1606.03439*, 2016. 2

Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039*, 2018. 1, 6

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014. 1

Yann LeCun, Sumit Chopra, and Raia Hadsell. A tutorial on energy-based learning. 2006. 1, 2

Yang Liu, Prajit Ramachandran, Qiang Liu, and Jian Peng. Stein variational policy gradient. *arXiv preprint arXiv:1704.02399*, 2017. 4

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018. 4, 6, 14

Andriy Mnih and Geoffrey Hinton. *Learning nonlinear constraints with contrastive backpropagation*. Citeseer. 2

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010. 2

Radford M Neal. Annealed importance sampling. *Stat. Comput.*, 11(2):125–139, 2001. 6

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016. 5

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *NIPS*, 2016. 5, 6

Tijmen Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th international conference on Machine learning*, pages 1064–1071. ACM, 2008. 4

Aaron Van Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *ICML*, 2016. 1

Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 681–688, 2011. 3

Grady Williams, Andrew Aldrich, and Evangelos A Theodorou. Model predictive path integral control: From theory to parallel computation. *Journal of Guidance, Control, and Dynamics*, 40(2):344–357, 2017. 2, 3

Raymond A Yeh, Chen Chen, Teck-Yian Lim, Alexander G Schwing, Mark Hasegawa-Johnson, and Minh N Do. Semantic image inpainting with deep generative models. 7

Junbo Zhao, Michael Mathieu, and Yann LeCun. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126*, 2016. 2

9 Appendix

9.1 Derivation

Assuming that we have the following equation NLL objective with $q(y) = p(y)$

$$\mathbb{E}_{x \sim p_d(x)} \left[E(x; \theta) + \log(\mathbb{E}_{y \sim q(x)}[e^{-E(y;\theta)}/q(y)]) \right]$$

is equal to

$$\mathbb{E}_{x \sim p_d(x)} \left[E(x; \theta) + \log(\mathbb{E}_{y \sim q(x)}[e^{-E(y;\theta)}/\texttt{stop\_gradient}(e^{-E(y;\theta)})]) \right] \tag{11}$$

Taking the gradient of the above expression gets

$$\mathbb{E}_{x \sim p_d(x)} \left[\nabla_\theta E(x; \theta) - \mathbb{E}_{y \sim q(x)}[\nabla_\theta E(y; \theta)] \right] \tag{12}$$

Getting us an original objective of

$$\mathbb{E}_{x \sim p_d(x)} \left[E(x; \theta) - \mathbb{E}_{y \sim q(x)}[E(y; \theta)] \right]$$

Alternatively, assuming $q(y) = p(y)$ we can also directly derive the NLL gradient as

$$\mathbb{E}_{x \sim p_d(x)} \left[\nabla_\theta E(x; \theta) \right] - \frac{\nabla_\theta Z(\theta)}{Z(\theta)}$$

Focusing on the second term, assuming suitable regularity conditions, we bring the gradient inside the integral to obtain

$$-\frac{\int (\nabla_\theta E(x; \theta)) * e^{E(x;\theta)} dx}{Z(\theta)} = \mathbb{E}_{x \sim p(x)}[\nabla_\theta E(x; \theta)]$$

Giving the other gradient of NLL of

$$\mathbb{E}_{x \sim p_d(x)} \left[\nabla_\theta E(x; \theta) \right] - \mathbb{E}_{x \sim p(x)}[\nabla_\theta E(x; \theta)]$$

which is equivalent to Equation 12.

9.2 Algorithm Pseudocode

We present the pseudo-code for training EBMs with $q(x)$ based off Langevin Dynamics in Algorithm 1. For training with MPPI, the MCMC step can be suitably changed.

9.3 Additional Qualitative Evaluation

We present images from a unconditional generation on ImageNet in Figure 11, which we generate using the last 10 model snapshots of energy models. We find the presence of objects and scenes in some of the generated image with occasional hybrids (such as a presence of a toaster cat in middle bottom row).

We provide further images of cross class conversions using a conditional EBM model in Figure 12. Our model is able to convert images from different classes into reasonable looking images of the target class while sometimes preserving attributes of the original class.

Finally, we analyze nearest neighbors of images we generate in Figure 13.

9.4 Test Time Sampling Process

We provide illustration of image generation from conditional and unconditional EBM models starting from random noise in Figure 14 with small amounts of random noise added. Dependent on the image generated there is slight drift from some start image to a final generated image. We typically observe that as sampling continues, much of the background is lost and a single central object remains.

We find that if small amounts of random noise are added, all sampling procedures generate a large initial set of diverse, reduced sample quality images before converging into a small set of high

Algorithm 1 Training Algorithm for EBMs with Langevin Dynamics(LD)

INPUT: number of proposal steps n, train dataset D, gradient clip threshold of V
INITIALIZE: parameter θ of network and replay buffer $B \leftarrow \{\}$
while Training **do**
    $x^+, l^+ \leftarrow D$
    $x^- \leftarrow B$
    Replace 5% of the sample images in x^- with $U(0, 1)$
    for i={1..n} **do**
        x^- = x^- - $\lambda * \text{clip}(\nabla_\theta(E_\theta(x^-, l^+; \theta)), -V, V) + N(0, \epsilon)$
    end for
    $e\_pos = E(x^+, l^+; \theta), e\_neg = E(\texttt{stop\_gradient}(x^-), l^+; \theta)$
    $\text{Loss}_{ml} = e\_pos - \text{softmax}(\texttt{stop\_gradient}(e\_neg)) \cdot e\_neg$
    $\text{Loss}_{kl} = E(x^-, l^+; \texttt{stop\_gradient}(\theta))$ ▷ Only for time series data, otherwise set to 0
    $\text{Loss}_{reg} = e\_pos^2 + e\_neg^2$
    Update $E(x; \theta)$ with $\nabla_\theta(\text{Loss}_{ml} + \text{Loss}_{kl} + \text{Loss}_{reg})$
    $B \leftarrow x^-$
end while

probability/quality image modes that are modes of images in CIFAR10. However, we find that if sufficient noise is added during sampling, we are able to slowly cycle between different images with larger diversity between images (indicating successful distribution sampling) but with reduced sample quality.

Due to this tradeoff, we use a replay buffer to sample images at test time, with slightly high noise then used during training time. For conditional energy models, to increase sample diversity, during initial image generation, we flip labels of images early on in sampling.

9.5 Model

We use the residual model in Figure 15a for conditional CIFAR10 images generation and the residual model in Figure 15b for unconditional CIFAR10 and Imagenet images. We found unconditional models need additional capacity. Our conditional and unconditional architectures are similar to architectures in [Miyato et al., 2018].

We found definite gains with additional residual blocks. We further found that replacing global sum pooling with a fully connected network also worked but did not lead to substantial benefits. We use the zero init in [Anonymous, 2019b] and spectral normalization on all weights. We use conditional bias and gains in each residual layer for a conditional model. We found it important when down-sampling to do average pooling as opposed to strided convolutions. We use leaky ReLUs throughout the architecture.

We use the architecture in Figure 15c for particle time series regression.

9.6 Training Hyperparameters

For CIFAR10 experiments, we use 60 steps of LD to generate negative samples. We use a replay buffer of size of 10000 image. We scale images to be between 0 and 1. We clip gradients to have magnitude of 0.01 and use a step size of 10 for each gradient step of LD. We use random noise with standard deviation of 0.005. We train our model on 1 GPU for 2 days. We use the Adam Optimizer with $\beta_1 = 0.0$ and $\beta_2 = 0.999$ with a training learning rate of 1e-4. We use a batch size during training of 128 positive and negative samples. For both experiments, we clip all training gradients that are more than 3 standard deviations from the 2nd order Adam parameters. We use spectral normalization on networks without backpropagating through the sampling procedure. We use the identical setup for ImageNet 32x32 images, but train for 3 days on 1 GPU.

For trajectories, we use 20 steps of LD to generate negative samples. We use a noise standard deviation 0.005. We use a batch size of 256 positive and negative samples. We found that a replay buffer was not necessary. We use the Adam Optimizer with $\beta_1 = 0.0$ and $\beta_2 = 0.999$. For MPPI, we use 30 steps of simulation with 5 noise simulations per step. We found spectral normalization to be overly restrictive on trajectories so we instead backpropogate through the sampling procedure.

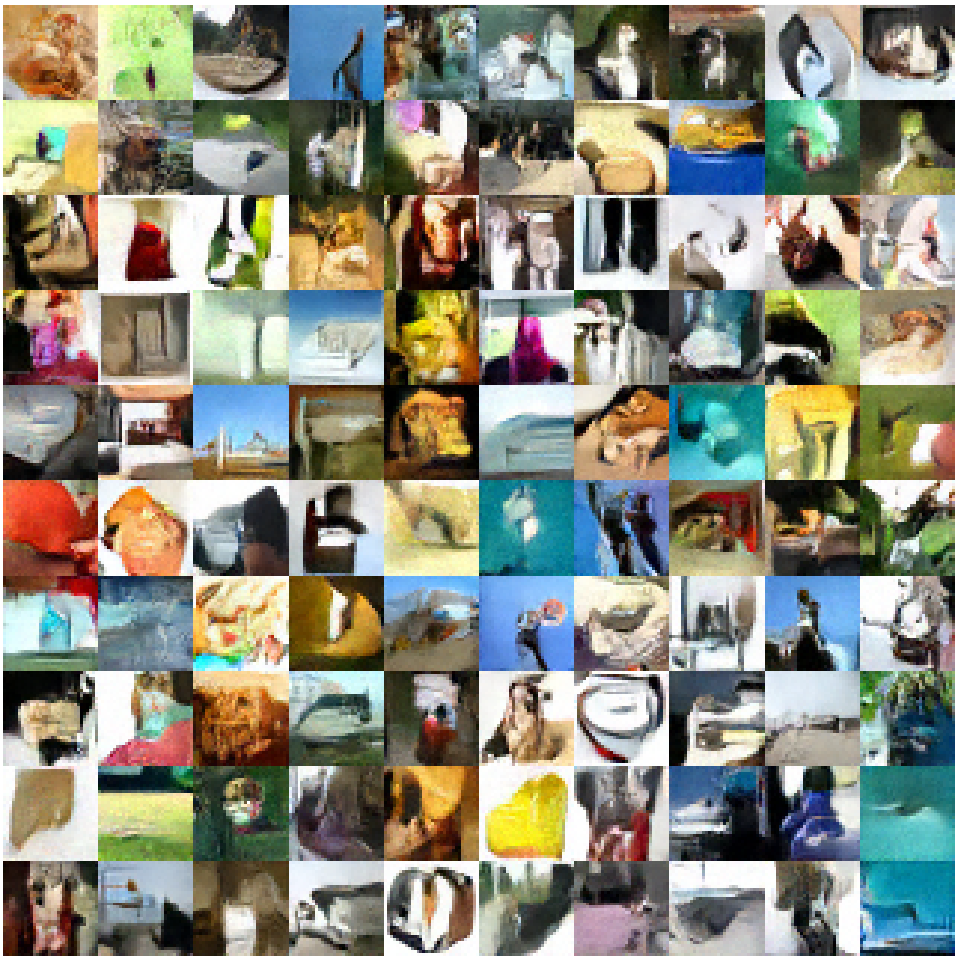


Figure 11: MCMC samples from unconditional ImageNet 32x32 EBM model

9.7 Tips And Failures

We provide a list of tips, observations and failures that we observe when trying to train energy based models. We found evidence that suggest the following observations, though in no way are we certain that these observations are correct.

We found the following tips useful for training.

- When training EBMs, we found the most important hyper-parameters to tune are MCMC transition step sizes. We found that as long as this hyper-parameter was tuned correctly models would train stably.
- We found that it is important to use piecewise linear activations in EBMs (either ReLU or LeakyReLU). We found that other activations gave poor results and instability.
- When using residual networks, we found that performance can be improved by using 2D average pooling as opposed to transposed convolutions
- We found that group, layer, batch, pixel or other types of normalization appeared to significantly hurt sampling, likely due to making MCMC steps dependent on surrounding data points.
- During a typical training run, we keep training until the sampler is unable to generate effective samples (when energies of proposal samples are much larger than energies of data points from the training dataset). Therefore, to extend training for longer time periods, the number of sampling steps can be increased during long time periods for better generation.

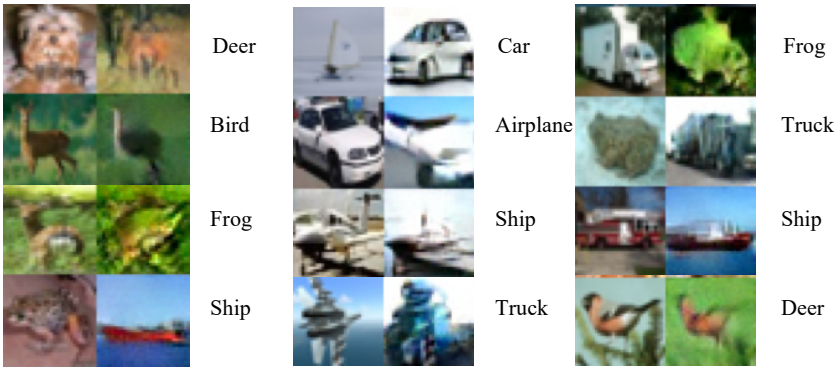


Figure 12: Illustration of more cross class conversion applying MCMC on a conditional EBM. We condition on a particular class but is initialized with an image from a another class(left). We are able to preserve certain aspects of the image while altering others

- Generally, we would recommend first trying to train EBMs without a sampling loss with spectral normalization. Only if this doesn't given satisfactory results would we recommend using sampling loss, as this causes models to train slowly.
- We find that there appears to be a direct relationship between depth and sample quality. Simply increase model depth can easily increase generation quality.
- When adding noise when using MCMC sampling, we found that very low levels of noise led to poor results. We found that high levels of noise allowed large amounts of mode exploration initially but quickly led to early collapse of sample (failure to explore modes).

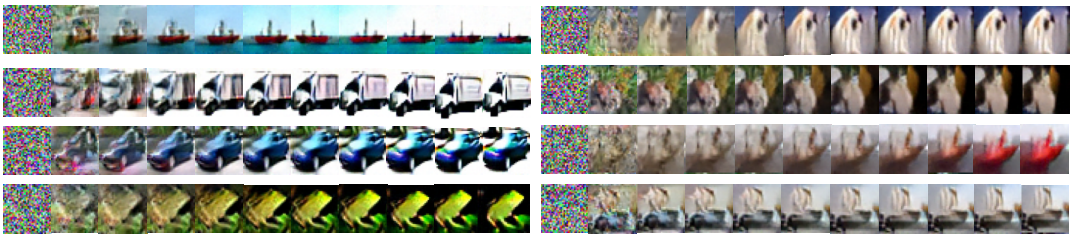
We also tried the approaches below with the relative success levels indicated. For training of models in this paper, we do not use any of the additions listed below.

- We found that training ensembles of energy functions (sampling and evaluating on ensembles) to help a bit, but was not worth the added complexity.
- We found that multistep HMC or Adam based updates didn't work well with sampling as the momentum term appeared to add a large amount of noise to the sampling procedure. We did observe that include second order information helped training.
- We didn't find much success with adding a gradient penalty term as it seemed to destablize sampling from the proposal distribution through LD.
- We found that a version of label discovery, where we assigned each data point a label with the lowest energy (normalized by the average energy the label assigned to other point) to provide some benefit. However, we found this gain in performance could also be obtained by simply increase model parameters.
- We tried a version of proposal distillation where we tried to make each proposal step equal to the final outcome after a large number of proposal steps. We found small benefits but did not found this computationally expensive.
- We tried training a separate network to help parametrize MCMC sampling but found that this made training very unstable. However, we did find that using some part of the original model to parametrize MCMC (such as using the magnitude to energy to control step size) to help performance.

(a) Nearest neighbor images in CIFAR10 for conditional energy models (leftmost generated, seperate class per row).

(b) Nearest neighbor images in CIFAR10 for unconditional energy model (leftmost generated)

Figure 13: Nearest neighbor images for images generated with GEO



(a) Illustration of GEO on conditional model of CIFAR10

(b) Illustration of GEO on unconditional model on CIFAR10

Figure 14: Generation of images from random noise.

3x3 conv2d, 128
ResBlock down 128
ResBlock 128
ResBlock down 256
ResBlock 256
ResBlock down 256
ResBlock 256
Global Sum Pooling
dense $\rightarrow$ 1

(a) Architecture used for conditional CIFAR10 experiments

3x3 conv2d, 128
ResBlock down 128
ResBlock 128
ResBlock 128
ResBlock down 256
ResBlock 256
ResBlock 256
ResBlock down 256
ResBlock 256
ResBlock 256
Global Sum Pooling
dense $\rightarrow$ 1

(b) Architecture used for unconditional CIFAR10 experiments

FC 32
Self Attention Block
Conv1D 128
Conv1D 128
dense $\rightarrow$ 1

(c) Architecture used for Time Series Experiments