Analyzing and Improving Adversarial Training for Generative Modeling

Xuwang Yin  
Department of Electrical and Computer Engineering  
University of Virginia  
xy4cm@virginia.edu

Shiying Li  
Department of Biomedical Engineering  
University of Virginia  
sl8jx@virginia.edu

Gustavo K. Rohde  
Department of Electrical and Computer Engineering  
University of Virginia  
gustavo@virginia.edu

Abstract

We study a new generative modeling technique based on adversarial training (AT). We show that in a setting where the model is trained to discriminate in-distribution data from adversarial examples perturbed from out-distribution samples, the model learns the support of the in-distribution data. The learning process is also closely related to MCMC-based maximum likelihood learning of energy-based models (EBMs), and can be considered as an approximate maximum likelihood learning method. We show that this AT generative model achieves competitive image generation performance to state-of-the-art EBMs, and at the same time is stable to train and has better sampling efficiency. We demonstrate that the AT generative model is well-suited for the task of image translation and worst-case out-of-distribution detection.

1. Introduction

In unsupervised learning, energy-based models (EBMs) [45] are a class of generative model that uses an energy function to model the probability distribution of the observed data. Unlike explicit density models, EBMs model the unnormalized density function, which makes it difficult to evaluate the likelihood function. Maximum likelihood learning of EBMs hence makes use of the likelihood function’s gradient which can be approximated using Monte Carlo methods. Each iteration of the learning process consists of first generating synthesized data by sampling from the current model, and then updating the model to maximize the energy difference between synthesized data and observed data. This process leads to an energy function that outputs low energies on the data manifold and high energies on other regions. Important applications of EBMs include sample generation, image restoration (e.g., denoising, inpainting), and out-of-distribution detection. The main difficulties of training EBMs lie in the computational challenges from the sampling procedure and some training stability issues [15, 16, 20, 21, 55, 56, 81, 87].

Another line of work on adversarial training (AT) show that adversarially robust classifiers learn high-level, interpretable features, and thus can be utilized to solve various computer vision tasks including generation, inpainting, super-resolution, and image-to-image translation [17, 68]. Compared to state-of-the-art generative models, this AT approach does not provide a competitive generation performance and is therefore of limited value in many of these tasks. Nonetheless, the generative properties of the robust classifier suggest that the model has captured the distribution of the training data, although the underlying learning mechanism is not yet well understood.

At a high level, both EBMs training and AT are based on the idea of first using gradient-based optimization to generate samples that reach high activations under the current model, and then optimizing the model to minimize its activations on the generated samples. In addition, both approaches synthesize new samples by performing gradient descent on the trained model. These similarities suggest that there are some connections between these two methods.

In this work we investigate the mechanism by which AT learns data distributions, and propose improved techniques for generative modeling with AT. Our focus is on the binary AT objective [85] which is simpler to analyze and naturally fits the generative modeling task. We first analyze the binary AT objective and the corresponding training algorithm, and show that binary AT learns a special kind of energy function that models the support of the observed data. We then draw a connection between AT and MCMC-base maximum likelihood learning of EBMs by showing that the binary AT objective can be interpreted as a gradient-scaled version
of the likelihood objective in EBMs training, and the PGD attack can be viewed as an non-convergent sampler of the model distribution. This connection provides us with intuition of how AT learns data distributions from the maximum likelihood learning perspective, and suggests that binary AT can be viewed as an approximate maximum likelihood learning algorithm.

Based on the above analysis we propose improved techniques for generative modeling with AT. Our empirical evaluation shows that this AT approach provides competitive generation performance to explicit EBMs, and at the same time is stable to train (just like regular adversarial training), is well-suited for image translation tasks, and exhibits strong out-of-distribution adversarial robustness. The main limitation of the studied approach is that it cannot properly learn the underlying density function of the observed data. However, this problem is not unique to the studied approach - most existing work on learning EBMs relies on short-run non-convergent sampler to improve the training efficiency, and the learned model typically does not have a valid steady-state that reflects the distribution of the observed data [55, 56].

In summary, the contributions of this paper are: 1) We show that binary AT learns a special kind of energy function that models the support of the data distribution, and the learning process is closely related to MCMC-based maximum likelihood learning of EBMs. 2) We propose improved techniques for generative modeling with AT, and demonstrate competitive image generation performance to state-of-the-art explicit EBMs. 3) We show that the studied approach is stable to train, has competitive training and test time sampling efficiency, and can be applied to denoising, inpainting, image translation, and worst-case out-of-distribution detection.

2. Related Work

Learning EBMs Due to the intractability of the normalizing constant, maximum likelihood learning of EBMs makes use of the gradient of the log-likelihood which can be approximated using MCMC sampling. Recent work [16, 55, 56, 83] scaling EBMs training to high-dimensional data performs sampling using SGLD [80] and initialize the chain from a noise distribution. The sampling process involves estimating the model’s gradient with respect to the current sample at each step and thus has high computational cost. To improve the sampling efficiency, many authors consider short-run non-convergent SGLD sampler in combination with a persistent sampling buffer [15, 16, 20, 21, 55, 81]. Although a short-run sampler is sufficient for learning a generation model, the resulting energy function typically does not have a valid steady-state [55, 56]. The mixing time of the sampling procedure also depends on how close the chain-initialization distribution is to the model distribution. A recent trend hence considers initializing the sampling chain from samples produced by a generator model fitted on the observed data [2, 21, 24, 25, 42, 58, 81, 82, 84]. There are also works focusing on addressing the training stability issues of EBMs [15, 87]. Due to the computational challenge and stability issues, the successful application of EBMs to modeling high-dimensional data such as 256 × 256 images is only achieved in some very recent works [81, 87]. The generation performance of these state-of-the-art EBMs is also not as competitive as some other approaches such as GANs.

Apart from MCMC-based maximum likelihood learning of EBMs, alternative approaches for learning EBMs exist. Score matching [33] circumvents the difficulty of estimating the partition function by directly modeling the derivatives of the data distribution. This method has recently been successfully applied to modeling large natural images and achieves competitive performance to state-of-the-art generative models such as GANs [31, 74–76]. Noise contrastive estimation (NCE) [23] learns data distributions by contrasting the observed data with data from a known noise distribution. Similar to our approach, NCE makes use of a logistic regression model. The main difference is that in NCE, the logit of the classifier is the difference in log probabilities of the model distribution and the noise distribution, whereas in our approach the logit directly defines the estimator (i.e., the energy function). Unlike other EBMs, NCE typically does not scale well to high-dimensional data [6, 23, 64].

Worst-case out-of-distribution detection We provide a discussion of related work in the supplementary materials.

3. Background

3.1. Energy-Based Models

Energy-based models (EBMs) [45] represent probability distributions by converting the outputs of a scalar function \( f_\theta \) into probabilities through a Gibbs distribution:

\[
p_\theta(x) = \frac{\exp(f_\theta(x))}{Z(\theta)},
\]

where the normalizing constant \( Z(\theta) \), also known as the partition function, is an integral over the unnormalized probability of all states: \( Z(\theta) = \int \exp(f_\theta(x))dx \). The energy function can be defined as \( E_\theta(x) = -f_\theta(x) \), and thus has the property of attributing low energy outputs on the support of the modeled data and high energy outputs on other regions.

For many interesting models, the partition function \( Z(\theta) \) is intractable, and thus maximum likelihood estimation (MLE) of the model parameters \( \theta \) is not straightforward. Standard maximum likelihood learning of EBMs makes use of the gradient of the log likelihood function. Denote the distribution of the observed data as \( p_{\text{data}} \), the gradient of the
log likelihood takes the following form:
\[
\nabla \theta \mathbb{E}_{x \sim \text{data}}[\log p_\theta(x)] = \mathbb{E}_{x \sim \text{data}}[\nabla \theta f_\theta(x)] - \mathbb{E}_{x \sim p_\theta(x)}[\nabla \theta f_\theta(x)].
\] (2)

Intuitively, maximum likelihood learning with this gradient causes \(f_\theta(x)\) to increase on \(p_{\text{data}}\) samples and decrease on samples drawn from \(p_\theta\), when \(p_\theta\) matches \(p_{\text{data}}\), the gradient cancels out and the training terminates.

Evaluating \(\mathbb{E}_{x \sim p_\theta(x)}[\nabla \theta f_\theta(x)]\) requires sampling from the model distribution, which is typically done using Markov chain Monte Carlo (MCMC) methods. Stochastic Gradient Langevin Dynamics (SGLD) \([80]\) is a widely used sampler in recent work that scales EBM training to high-dimensional data \([16, 55, 56, 83]\). SGLD performs sampling by projecting onto the feasible set \(\mathbb{B}(x, \epsilon)\). Because the gradient vector in Eq. (6) is normalized to have unit norm, we can equivalently implement the attack by directly performing gradient ascent on \(f_\theta\):
\[
x_0 \sim p_\theta, \quad x_{i+1} = \text{Proj} \big( x_i + \lambda \frac{\nabla_x f_\theta(x_i)}{\|\nabla_x f_\theta(x_i)\|_2} \big), \quad (7)
\]

Let \(p_\theta\) be some random noise distribution. A proper SGLD sampler requires a large number of update steps in order for the distribution of sampled data to match \(p_\theta\). Due to the high computational cost of the sampling procedure, many authors resort to short-run non-convergent MCMC to improve the sampling efficiency \([16, 20, 55, 56, 83]\). The resulting model typically does not have a valid steady-state that reflects the distribution of the observed data, but is still capable of generating realistic and diverse samples \([55, 56]\).

3.2. Binary Adversarial Training

Binary adversarial training \([85]\) is a method for detecting adversarial examples. In a \(K\) class classification problem, the detection system consists of \(K\) individual base detectors (binary classifiers), with the \(k\)-th detector trained to discriminate between clean data of class \(k\) and adversarial examples perturbed from data of other classes. A committee of \(K\) base detectors then provides a complete solution for detecting any adversarial examples.

Denote the data distribution of class \(k\) as \(p_{\text{data}}\), the mixture distribution of other classes as \(p_0 = \frac{1}{K-1} \sum_{i=1, \ldots, K, i \neq k} p_i\), the \(k\)-th detector is trained by maximizing the objective
\[
J(D) = \mathbb{E}_{x \sim p_{\text{data}}} [\log D(x)] + \mathbb{E}_{x \sim p_0} [\min_{x' \in \mathbb{B}(x, \epsilon)} \log(1 - D(x'))], \quad (4)
\]

where \(D : \mathcal{X} \subseteq \mathbb{R}^d \rightarrow [0, 1]\) is the detector function, and \(\mathbb{B}(x, \epsilon)\) is a neighborhood of \(x\): \(\mathbb{B}(x, \epsilon) = \{x' \in \mathcal{X} : \|x' - x\|_2 \leq \epsilon\}\). In practice, \(D\) is defined by applying a logistic sigmoid function to the output of a neural network:
\[
D(x) = \sigma(f_\theta(x)), \quad (5)
\]

where \(f_\theta\) is a neural network with a single output channel and parameters \(\theta\). Eq. (4) is characterized by an inner minimization problem and an outer maximization problem; when the inner minimization is perfectly solved and the training achieves a vanishing loss, \(D\) becomes a perfectly robust model capable of separating \(p_{\text{data}}\) samples from any \(\epsilon\)-bounded adversarial examples perturbed from \(p_0\) data.

The inner minimization is solved using the PGD attack \([43, 48]\), a first-order method that employs an iterative update rule of (\(l^2\)-based attack):
\[
x_0 \sim p_0, \quad x_{i+1} = \text{Proj} \big( x_i - \lambda \frac{\nabla_x \log(1 - D(x_i))}{\|\nabla_x \log(1 - D(x_i))\|_2} \big), \quad (6)
\]

where \(\lambda\) is some step size, and \(\text{Proj}\) is the operation of projecting onto the feasible set \(\mathbb{B}(x, \epsilon)\). Because the gradient vector in Eq. (6) is normalized to have unit norm, we can equivalently implement the attack by directly performing gradient ascent on \(f_\theta\):
\[
x_0 \sim p_0, \quad x_{i+1} = \text{Proj} \big( x_i + \lambda \frac{\nabla_x f_\theta(x_i)}{\|\nabla_x f_\theta(x_i)\|_2} \big). \quad (7)
\]

Similar to standard adversarial training, a model trained with binary AT has strong interpretability — an unbounded attack that maximizes the model’s output results in samples that resemble data from \(p_{\text{data}}\), suggesting that the model has captured the distribution of the observed data of \(p_{\text{data}}\).

4. Analyzing and Improving Binary AT for Generative Modelling

In this section we analyze the generative properties of binary AT, with the goal of understanding the mechanism by which this method learns data distributions, and improving it for generative modeling. We begin with analyzing the binary AT objective and the training algorithm, then discuss the connection between binary AT and MCMC-based EBM learning, and finally propose improved techniques for generative modeling with AT.

4.1. Properties of Binary Adversarial Training

4.1.1 Optimal Solution

We consider the optimal solution of Eq. (4) under the scenario of unbounded perturbation: \(\mathbb{B}(x, \epsilon) = \mathcal{X}\). Perturbing \(p_0\) samples can be thought of as moving \(p_0\) samples via a translation function \(T(x) = x + \Delta_x\), with \(\Delta_x\) being the perturbation computed on sample \(x\). We can write the density function of the perturbed distribution \(p_T\) using random variable transformation:
\[
p_T(y) = \int_{\mathcal{X}} p_0(x) \delta(y - T(x)) dx. \quad (8)
\]

The inner problem in Eq. (4) can then be interpreted as determining the distribution that has the lowest expected
value of \( \log(1 - D(x)) \):

\[
p_T^* = \arg \min_{p_T} \mathbb{E}_{x \sim p_T} [\log(1 - D(x))]. \tag{9}
\]

The objective of the outer problem is then the log-likelihood in a logistic regression model which discriminates \( p_{\text{data}} \) samples from \( p_T^* \) samples:

\[
J(D) = \mathbb{E}_{x \sim p_{\text{data}}} [\log D(x)] + \mathbb{E}_{x \sim p_T^*} [\log(1 - D(x))]. \tag{10}
\]

We can equivalently formulate Eq. (4) as a maximin problem:

\[
\max_{p_T} \min_D U(D, p_T) = \mathbb{E}_{x \sim p_{\text{data}}} [\log D(x)] + \mathbb{E}_{x \sim p_T^*} [\log(1 - D(x))]. \tag{11}
\]

**Proposition 1.** The optimal solution of \( \max_D \min_{p_T} U(D, p_T) \) is \( U(D^*, p_T^*) = -\log(4) \), where \( D^* \) outputs \( \frac{1}{2} \) on \( \text{Supp}(p_{\text{data}}) \) and \( \leq \frac{1}{2} \) outside \( \text{Supp}(p_{\text{data}}) \), and \( p_T^* \) has its mass distributed in locations where \( D \) outputs \( \frac{1}{2} \).

**Proof.** See the supplementary materials. \( \square \)

The above maximin problem can also be interpreted as a two-player zero-sum game, and is closely related to the \emph{minimax} game in GANs [19]. The game-theory point of view provides a convenient way to understand their differences. We include a game theory-based analysis of \( \max_D \min_{p_T} U(D, p_T) \) and a comparative analysis of GANs in the supplementary materials.

### 4.1.2 Training Algorithm Analysis

Proposition 2 states that by solving \( \max_D \min_{p_T} U(D, p_T) \) we can obtain a \( D \) that outputs \( \frac{1}{2} \) on the support of \( p_{\text{data}} \) and \( \leq \frac{1}{2} \) on other regions. This result is obtained by assuming that for any \( D \), the inner problem Eq. (9) is always perfectly solved. In practice, the inner problem is solved by performing PGD attacks on \( p_0 \) samples; when \( D \) is a non-concave function, this process could get trapped in different local maxima of \( D \). Given that an optimal \( p_T^* \) is not always attainable, the actual \( D \) solution obtained from an algorithm could be different from the one predicted by Proposition 2.

**Algorithm 1** Binary Adversarial Training

1. \( \text{repeat} \)
2. Draw samples \( \{x_i\}_{i=1}^m \) from \( p_{\text{data}} \), and samples \( \{x_i^0\}_{i=1}^m \) from \( p_0 \).
3. Update \( \{x_i^0\}_{i=1}^m \) by performing \( K \) steps PGD attack Eq. (7) on each sample. Denote the resulting samples as \( \{x_i^1\}_{i=1}^m \).
4. Update \( D \) by maximizing \( \frac{1}{m} \sum_{i=1}^m \log D(x_i) + \frac{1}{m} \sum_{i=1}^m \log(1 - D(x_i^1)) \) (single step).
5. \( \text{until} \) \( D \) convergences

The pseudo code for binary AT is outlined in Algorithm 1. The algorithm starts by drawing sampling from \( p_{\text{data}} \) and \( p_0 \). Step 3 then solves the inner problem Eq. (9) by performing PGD attacks on \( p_0 \) samples. When \( D \) is a non-concave function, the resulting samples \( \{x_i^1\}_{i=1}^m \) are in local maxima of \( D \). In step 4, \( D \) is updated by maximizing its outputs on \( \{x_i\}_{i=1}^m \) and minimizing its outputs on \( \{x_i^1\}_{i=1}^m \). A key observation is that when \( \{x_i^1\}_{i=1}^m \) are in local maxima, the update in step 4 causes these local maxima values to decrease. Hence alternating between these two optimization procedures causes local maxima of \( D \) to be constantly suppressed.

Fig. 1 top panel shows the 2D simulation result of the algorithm when \( p_0 \) data are some random samples from a uniform distribution. It can be seen that when the algorithm converges, local maxima outside the support of \( p_{\text{data}} \) are suppressed, and \( D \) (approximately) outputs \( \frac{1}{2} \) on the support of \( p_{\text{data}} \), which is consistent with Proposition 2. Meanwhile, \( D \) retains gradient information for transforming out-distribution samples to the support of \( p_{\text{data}} \). Fig. 1 bottom panel shows that when \( p_0 \) data is concentrated in the bottom left corner, the final \( D \) has local maxima outside the support of \( p_{\text{data}} \). This result suggests that the \( D \) solution is affected by the setting of \( p_0 \), and in order to learn a valid energy function, the support of \( p_0 \) should span as much space as possible.

![Figure 1](image-url)

Figure 1. Plots of contours and (normalized) gradient vector fields of the \( D \) functions learned with different \( p_0 \) data. Left images: \( p_0 \) data (red points) and initial states of the \( D \) functions. Right images: \( p_1^* \) data (red points) and the final states of the \( D \) functions when the algorithm converges. \( p_{\text{data}} \) is a Gaussian distribution centered at \((0, 0)\) (blue points). \( D \) is parameterized by a multilayer perceptron.

The above analysis reveals how binary AT learns data distributions: the learning process starts with a randomly-initialized \( D \) solution, and iteratively refine the solution by suppressing local maxima outside the support of the observed data. This process is similar to EBMs training where the model distribution’s spurious modes are constantly lo-
ated by MCMC sampling and subsequently suppressed in the model update stage. However, unlike the EBMs likelihood objective Eq. (2), the AT objective Eq. (11) does not support the learning of the observed data’s density function. This can also be observed in the 2D experiment where D outputs \( \frac{1}{2} \) uniformly on the support of \( p_{\text{data}} \) (blue points).

### 4.2. Connection to MCMC-based Maximum Likelihood Learning of EBMs

We next consider the learning process of binary AT from a maximum likelihood learning point of view. Both binary AT and MCMC-based EBMs learning employ an iterative optimization algorithm, where in each iteration the contrastive data is computed by performing gradient ascent on the current model, and then the model is updated by maximizing its outputs on the observed data and minimizing its outputs on the contrastive data. The following analysis shows that the PGD attack can be viewed as a non-convergent sampler of the model distribution, and the binary AT objective can be interpreted as a gradient-scaled version of the likelihood objective used in EBMs training. Tab. 1 summaries the main differences between these two methods.

| Objective | EBMs: \( E_{x \sim p_{\text{data}}} [\nabla_{\theta} f_0(x)] - E_{x \sim p_\theta(x)} [\nabla_{\theta} f_0(x)] \) |
|-----------|-------------------------------------------------|
| Binary AT: \( E_{x \sim p_{\text{data}}} [(1 - \sigma(f_0(x))) \nabla_{\theta} f_0(x)] - E_{x \sim p_\theta(x)} [\sigma(f_0(x)) \nabla_{\theta} f_0(x)] \) |

| Contrastive data | EBMs: \( x_0 \sim p_0, x_{i+1} = x_i + \frac{\lambda}{\left\| \nabla_{x_i} f_0(x_i) \right\|^2} \nabla_{x_i} f_0(x_i) \) |
|------------------|-------------------------------------------------------------------|
| Binary AT: \( x_0 \sim p_0, x_{i+1} = x_i + \lambda \frac{\nabla_{x_i} f_0(x_i)}{\left\| \nabla_{x_i} f_0(x_i) \right\|^2} \) |

| \( p_0 \) data | EBMs: A noise distribution or a distribution close to \( p_{\text{data}} \) |
|----------------|-------------------------------------------------------------------|
| Binary AT: A real and diverse out-distribution dataset (80 million tiny images [77] for CIFAR-10 [41] and ImageNet [13] for other 256 × 256 datasets, see Sec. 4.3) |

Table 1. The differences between binary AT and maximum likelihood EBMs in terms of training objective gradient, how contrastive data is computed, and the setting of \( p_0 \) data.

### Contrastive Data Computation

In EBMs training, the contrastive data is computed by MCMC-sampling, typically with Langevin dynamics Eq. (3). In binary AT, the contrastive data is computed using PGD attacks Eq. (7). As we are considering the unconstrained scenario \( \mathcal{B}(x, \epsilon) = \mathcal{X} \), we can simplify the attack by removing the \( \mathcal{P} \circ \mathcal{O} \) operator:

\[
x_0 \sim p_0, \quad x_{i+1} = x_i + \lambda \frac{\nabla_{x} f_0(x_i)}{\left\| \nabla_{x} f_0(x_i) \right\|^2} \tag{12}\]

Comparing Eq. (12) with Eq. (3), we see that both methods compute the contrastive data by first initializing from some out-distribution data, and then performing gradient ascent on \( f_\theta \). The main differences are that the PGD attack does not include the noise term, and makes use of normalized gradient. We first note that although Langevin dynamics requires the noise term to be a valid sampler, the noise term is not absolutely necessary when learning short-run EBMs [55].

In the PGD attack, as the normalized gradient vector has unit norm, the perturbation imposed on \( x_i \) is \( \lambda \); in a \( K \) iterations of the update, the overall perturbation \( \| x_i^T - x_i \|_2 \) is always \( \leq K \lambda \). Hence with the PGD attack we can more easily control the distribution of the contrastive data. In contrast, Langevin dynamics adjusts \( x_i \) in a scale that corresponds to the magnitude of the gradient of \( f_\theta \) at \( x_i \); when \( f_\theta \) is updated during training, the overall perturbation may undergo a large change. This behavior of Langevin dynamics can be a source of some training stability issues [55].

The PGD attack may not correspond exactly to a valid sampler, but in practice we find it capable of producing samples that follow the distribution of the modeled data. We also note that in MCMC-based EBMs learning, due to the high computational cost of MCMC sampling, it is not uncommon to use an invalid sampler which is short-run and non-convergent [16, 20, 55, 56, 83].

### Gradient of the training objective

With the definition in Eq. (5), the gradient of \( D \)'s training objective Eq. (10) takes the form

\[
\nabla_{\theta} J(D) = E_{x \sim p_{\text{data}}} [(1 - \sigma(f_\theta(x))) \nabla_{\theta} f_\theta(x)] - E_{x \sim p_\theta(x)} [\sigma(f_\theta(x)) \nabla_{\theta} f_\theta(x)]. \tag{13}\]

Comparing the above equation with Eq. (2) we find both equations consisting of gradient terms that yield similar effects: the first term causes \( f_\theta \) outputs on \( p_{\text{data}} \) samples to increase, and the second causes \( f_\theta \) outputs on the contrastive samples to decrease. Specifically, as \( \sigma(f_\theta(x)) \) and \( \sigma(f_0(x)) \) are scalars in the range 0 to 1, the gradient terms in Eq. (13) are scaled versions of the gradient terms in Eq. (2). Although Eq. (13)'s gradient update direction is still the same as Eq. (2), these scalars could cause the objective to converge to different points than the maximum likelihood estimator.

Eq. (13) also helps us understand why binary AT can only learn the support of the observed data. In Eq. (2), when \( p_\theta(x) \) matches \( p_{\text{data}} \), the gradient cancels out and training terminates, whereas in Eq. (13), when \( p_\theta(x) \) matches \( p_{\text{data}} \), the gradient becomes \( E_{x \sim p_{\text{data}}} [(1 - 2\sigma(f_\theta(x))) \nabla_{\theta} f_\theta(x)] \) and only vanishes when \( \sigma(f_\theta(x)) = \frac{1}{2} \) everywhere on the support of \( p_{\text{data}} \). This result is consistent with Proposition 2 and the 2D experiment result.

### 4.3. Improving Binary AT for Generative modeling

Building on insights from the analysis in Sec. 4.1, we propose improved techniques for generative modeling with AT. Specifically, we propose to use a realistic and diverse out-distribution dataset as the \( p_0 \) dataset, and train with an unconstrained objective. We also address a failure mode of the training algorithm that we observe on CelebA-HQ [35].
Progressive Binary Adversarial Training

Algorithm 2

1: for \( K \) in \([0, 1, \ldots, N]\) do
2:     for number of training iterations do
3:         Draw samples \( \{x_i\}_{i=1}^m \) from \( p_{\text{data}} \), and samples \( \{x^0_i\}_{i=1}^m \) from \( p_0 \).
4:         Update \( \{x^0_i\}_{i=1}^m \) by performing \( K \) steps unconstrained PGD attack Eq. (12) on each sample. Denote the resulting samples as \( \{x^*_i\}_{i=1}^m \).
5:         Update \( D \) by maximizing \( \frac{1}{m} \sum_{i=1}^m \log D(x_i) + \frac{1}{m} \sum_{i=1}^m \log(1 - D(x^*_i)) \) (single step).
6:     end for
7: end for

Failure mode When we use Algorithm 2 to train on CelebA-HQ 256 [35], we observe that the binary classification accuracy quickly reaches 100%; increases \( K \) causes the accuracy to drop before it quickly returns 100%. Meanwhile the \( l^2 \) distance between \( x^0_i \) and \( x^*_i \) is only a small fraction of \( \lambda K \), and \( x^*_i \) shows no meaningful high-level features of human faces. This failure happens because the \( D \) model has only learned a handful of low-level features which correlate well with the labels. These features are sufficient for the classification task but do not provide enough informative gradient for \( p_0 \) data to move to the manifold of \( p_{\text{data}} \). This issue is also observed in GANs training, and we similarly address it using \( R_1 \) regularization [51]. Apart from this failure mode, we didn’t observe other stability issues. In contrast, in existing work on learning EBMs, there does not seem to be a consensus about how to stabilize training (Tab. 2).

Table 2. Techniques used for stabilizing EBMs training.

| Method | Stabilizing techniques |
|--------|------------------------|
| Ours   | \( R_1 \) regularization [51] |
| VAEBM [81] | Weight normalization [67]; Swish activation [61]; gradient clipping; weight decay |
| CF-EBM [87] | Multistage coarse-to-fine expanding and sampling; smooth activation functions |
| ImprovedCD [15] | Gradient norm clipping on model parameters; use a KL term in the training objective |
| JEM [20] | Adjust learning rate and SGLD steps during training; add Gaussian noise to input images |
| IGEBM [16] | Gradient clipping on SGLD and model parameters; spectral normalization |

5. Experiments

This section provides an empirical evaluation of the proposed AT approach for generative modeling. We first evaluate the approach’s effectiveness in terms of image generation performance and sampling efficiency, and then provide an analysis on the method’s training stability, and finally demonstrate its applications to worst-case out-of-distribution detection, denoising, inpainting, and image translation. Experiment setup including model architectures, training hyperparameters, sample generation settings, and evaluation protocols can be found in the supplementary materials.

5.1. Image Generation Performance

Tab. 4 shows that on CIFAR-10 [41] our approach improves over state-of-the-art explicit EBMs in terms of Inception Score, and at the same time has a slightly worse FID. (Generated samples can be found in the supplementary materials.) Compared to VAEBM [81], our method does not require an auxiliary model to train, and has better test time sampling efficiency (Tab. 6). Diffusion Recovery [18] trains a sequence of conditional EBMs, with each one defining the conditional distribution of a noisy sample given the same sample at a higher noise level. Similar to score-based approaches, these conditional EBMs do not directly model the
data distribution of the observed data, so it is unclear how these models can be applied to tasks which require explicit knowledge of the data distribution (e.g., out-of-distribution detection).

Tab. 4 shows that on CelebA-HQ 256 [35] our method outperforms (or is on par with) state-of-the-art generative models except GANs. Our method similarly falls below GANs on AFHQ-CAT 256 [9]. Fig. 2 shows that our approach is capable of generating realistic images, although the generated samples contain artifacts and the overall quality is not as good as those obtained with state-of-the-art generative models.

In addition, the interpolation results and nearest neighbor analysis in the supplementary materials show that the model has captured the manifold structure of the observed data, as opposed to simply memorized the data samples.

| Method                                | IS↑ | FID↓ |
|---------------------------------------|-----|------|
| Ours                                  | 9.10 | 13.21 |
| Adversarially Robust Classifier [68]  | 7.5  | -    |

**Table 3. IS and FID scores on CIFAR-10.**

**5.2. Training and Test Time Sampling Efficiency**

Tab. 5 shows that our method has competitive training and test time sampling efficiency. (See the supplementary materials for the number of training iterations.) VAEBM typically requires much fewer update steps than our method, but its per-step efficiency is much worse (Tab. 6), suggesting that its VAE component has considerable computational complexity. We also observe that the quality of our generated samples is not sensitive to the number of sampling steps as long as the overall perturbation (#step × step-size) remains the same (Tab. 7). This allows us to use a much larger step size than the one used during training to speedup test time sampling in real applications.

**Table 5. The number of update steps in the PGD attack (our method) and Langevin dynamics (other methods).** Each entry shows the training time #steps (left value) and the test time #steps (right value). “PCD” refers to using a persistent sampling chain.

**Table 6. Number of steps and time to generate 50 CIFAR-10 samples.** Data of NCSN and VAEBM are from Xiao et al. [81].

**5.3. Training Stability Analysis**

To gain some insight into the training stability of our approach we test whether the PGD attack can be used
with the EBMs training objective Eq. (2). Specifically, in Algorithm 2, we perform step 5’s update on \( \theta \) using gradient \( \nabla_\theta (\frac{1}{m} \sum_{i=1}^{m} f_\theta(x_i) - \frac{1}{m} \sum_{i=1}^{m} f_\theta(x^*_i)) \). We observe that even under a small learning rate of \( 1e^{-6} \), \( \frac{1}{m} \sum_{i=1}^{m} f_\theta(x_i) - \frac{1}{m} \sum_{i=1}^{m} f_\theta(x^*_i) \) quickly increases and eventually overflows. This suggests that the stability of the AT approach can be largely attributed to the log-likelihood objective Eq. (10). We argue that the stability is due to the gradient cancelling effect of this objective: when \( f_\theta \) has a large positive output on a sample \( x \sim p_{\text{data}} \), \( -\sigma(f_\theta(x)) \) approaches 0 and therefore the corresponding scaled gradient in Eq. (13) vanishes, and similarly \( \sigma(f_\theta(x^*)) \nabla_\theta f_\theta(x^*) \) vanishes when \( f_\theta \) has a large negative output on a sample \( x^* \sim p_{\gamma} \). In contrast, the EBMs training objective Eq. (2) does not have any constraints on \( f_\theta \)’s outputs and is therefore prone to divergence.

### 5.4. Applications

#### Worst-case out-of-distribution detection

Tab. 8 shows that our model achieves comparable OOD detection performance to the state-of-the-art method of RATIO. OE, RATIO, and JEM all perform OOD detection by utilizing a classifier that has low confidence predictions on the out-distribution data (clean or worst-case). In RATIO, the worst-case out-distribution data is computed by performing PGD attacks on 80 million tiny images [77], whereas in JEM it is computed via Langevin dynamics initialized from uniform random noise. RATIO and our method’s strong out-distribution adversarial robustness demonstrates the benefit of using a real and diverse out-distribution dataset to train the model. Our method does not make use of class labels and thus can be considered as a binary variant of RATIO. On CelebA-HQ 256 and AFHQ-CAT our model similarly achieves strong out-distribution adversarial robustness (see the supplementary materials). These results suggest that our generative model can be applied to detect both naturally occurring OOD data and adversarially created malicious content.

| OOD dataset | Classifier-based approach | Ours |
|-------------|---------------------------|------|
| OE [29]     | 99.4 / 0.6                | 93.0 / 81.6 |
| JEM [20]    | 89.3 / 7.3                | 88.3 / 70.3 |
| RATIO [3]   | 91.6 / 73.0               | 91.3 / 73.5 |

#### Denosing, inpainting, and image-to-image translation

Fig. 3 shows that the AFHQ-CAT model can be used to transform CelebA-HQ images into cat images, and vice-versa. Note that these two models are trained independently without knowledge of the source domain dataset, suggesting that our approach may generalize better to unseen data than approaches (e.g., pix2pix [34], CycleGAN [89], and StarGAN [9]) that use a fixed source domain dataset to train the model. However, our translation results may be further improved by finetuning the trained model on the source domain dataset, or including the source domain data in the \( p_{\theta} \) dataset during training. Synthesizing content with a dynamic process of gradient descent is also more flexible than using a fixed generation: it allows the user to choose how much transformation to apply to the content, or create cinematic effect from intermediate results. Demonstration of applications to denosing and inpainting, and more translation results are included in the supplementary materials.

![Figure 3. Translation between human faces and cat faces](image)

### 6. Limitations

The main limitation of the studied approach is that it only learns the support of the observed data, which makes it inappropriate for density estimation tasks. In many practical applications of generative models, like the ones considered in the paper, we did not find this limitation to be an issue.
Finding an appropriate $p_0$ dataset can be challenging. The ImageNet is a reasonable choice for image modeling tasks, but can a problem if the task is to model the ImageNet itself. A potential solution is to use the Open Images [44] as $p_0$ in this case. In other modalities or domains, a real and diverse out-distribution dataset may not be readily available.

7. Conclusion

We have studied an adversarial training-based approach for learning EBMs. Our analysis shows that binary AT learns a special kind of energy function that models the support of the observed data, and the training procedure can be viewed as an approximate maximum likelihood learning algorithm. We proposed improved techniques for generative modeling with AT, and demonstrated that the proposed method provides competitive generation performance to explicit EBMs, has competitive sampling efficiency, is stable to train, and is well-suited for image translation tasks. The proposed approach’s strong out-distribution adversarial robustness suggests its potential application to detecting abnormal inputs or adversarially created fake content. As future work, we consider improving the studied approach’s generation performance by mitigating overfitting with data augmentation [36, 78, 86, 88], and improving the training efficiency using PGD attacks with a persistent buffer.

References

[1] Vahdat Abdelzad, Krzysztof Czarnecki, Rick Salay, Taylor Denoudren, Sachin Vernekar, and Buu Phan. Detecting out-of-distribution inputs in deep neural networks using an early-layer output. arXiv preprint arXiv:1910.10307, 2019.

[2] Michael Arbel, Liang Zhou, and Arthur Gretton. Generalized energy based models. In International Conference on Learning Representations, 2021.

[3] Maximilian Augustin, Alexander Meinke, and Matthias Hein. Adversarial robustness on in-and out-distribution improves explainability. In European Conference on Computer Vision, pages 228–245. Springer, 2020.

[4] Julian Bitterwolf, Alexander Meinke, and Matthias Hein. Certifiably adversarially robust detection of out-of-distribution data. Advances in Neural Information Processing Systems, 33, 2020.

[5] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In International Conference on Learning Representations, 2019.

[6] Ciwan Ceylan and Michael U Gutmann. Conditional noise-contrastive estimation of unnormalised models. In International Conference on Machine Learning, pages 726–734. PMLR, 2018.

[7] Wenhu Chen, Yilin Shen, Hongxia Jin, and William Wang. A variational dirichlet framework for out-of-distribution detection. arXiv preprint arXiv:1811.07308, 2018.

[8] Hyunsun Choi, Eric Jang, and Alexander A Alemi. Waic, but why? generative ensembles for robust anomaly detection. arXiv preprint arXiv:1810.01392, 2018.

[9] Yunjey Choi, Youngjuh Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 8188–8197, 2020.

[10] F. Croce and M. Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In ICML, 2020.

[11] Ekin D Cubuk, Barret Zoph, Dandeline Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 113–123, 2019.

[12] Erik Daxberger and José Miguel Hernández-Lobato. Bayesian variational autoencoders for unsupervised out-of-distribution detection. arXiv preprint arXiv:1912.05651, 2019.

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pages 248–255. Ieee, 2009.

[14] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. arXiv preprint arXiv:1605.08803, 2016.

[15] Yilun Du, Shuang Li, Joshua B. Tenenbaum, and Igor Mordatch. Improved contrastive divergence training of energy based models. In ICML, 2021.

[16] Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. In Advances in Neural Information Processing Systems, volume 32, 2019.

[17] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Brandon Tran, and Aleksander Madry. Adversarial robustness as a prior for learned representations. arXiv preprint arXiv:1906.00945, 2019.

[18] Ruiqi Gao, Yang Song, Ben Poole, Ying Nian Wu, and Diederik P Kingma. Learning energy-based models by diffusion recovery likelihood. In International Conference on Learning Representations, 2021.

[19] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Advances in neural information processing systems, pages 2672–2680, 2014.

[20] Will Grathwohl, Kuan-Chieh Wang, Joern-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. In International Conference on Learning Representations, 2020.

[21] Will Sussman Grathwohl, Jacob Jin Kelly, Milad Hashemi, Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. No mcmc for me: Amortized sampling for fast and stable training of energy-based models. In International Conference on Learning Representations, 2021.

[22] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of
wasserstein gans. In Advances in Neural Information Processing Systems, volume 30, 2017. 7
[23] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Proceedings of the thirteenth international conference on artificial intelligence and statistics, pages 297–304. JMLR Workshop and Conference Proceedings, 2010. 2
[24] Tian Han, Erik Nijkamp, Xiaolin Fang, Song-Chun Zhu, and Ying Nian Wu. Divergence triangle for joint training of generator model, energy-based model, and inferential model. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 8670–8679, 2019. 2
[25] Tian Han, Erik Nijkamp, Linqi Zhou, Bo Pang, Song-Chun Zhu, and Ying Nian Wu. Joint training of variational autoencoder and latent energy-based model. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7978–7987, 2020. 2
[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016. 15
[27] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 41–50, 2019. 13
[28] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. arXiv preprint arXiv:1610.02136, 2016. 13
[29] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. arXiv preprint arXiv:1812.04606, 2018. 8, 13, 16
[30] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In Advances in neural information processing systems, pages 6666–6673, 2017. 15
[31] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Advances in Neural Information Processing Systems, volume 33, 2020. 2, 7
[32] Yujia Huang, Sihui Dai, Tan Nguyen, Richard G Baraniuk, and Jonathon Ho. Out-of-distribution detection using neural rendering generative models. arXiv preprint arXiv:1907.04572, 2019. 13
[33] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. Journal of Machine Learning Research, 6(4), 2005. 2
[34] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1125–1134, 2017. 8
[35] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196, 2017. 5, 6, 7, 15
[36] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In Advances in Neural Information Processing Systems, volume 33, pages 12104–12114, 2020. 7, 9, 15
[37] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4401–4410, 2019. 16, 18
[38] Diederik P Kingma. Fast gradient-based inference with continuous latent variable models in auxiliary form. arXiv preprint arXiv:1306.0733, 2013. 13
[39] Dark P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In Advances in neural information processing systems, pages 10215–10224, 2018. 7, 13
[40] Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Why normalizing flows fail to detect out-of-distribution data. Advances in neural information processing systems, 33, 2020. 13
[41] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5, 6, 15
[42] Rithesh Kumar, Sherjil Ozair, Anirudh Goyal, Aaron Courville, and Yoshua Bengio. Maximum entropy generators for energy-based models. arXiv preprint arXiv:1901.08508, 2019. 2
[43] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. arXiv preprint arXiv:1611.01236, 2016. 3
[44] Afina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, MatteoMalloci, Alexander Kolesnikov, et al. The open images dataset v4. International Journal of Computer Vision, 128(7):1956–1981, 2020. 9
[45] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. Predicting structured data, 1(0), 2006. 1, 2
[46] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In Advances in Neural Information Processing Systems, pages 7167–7177, 2018. 13
[47] Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. arXiv preprint arXiv:1706.02690, 2017. 13
[48] Aleksandr Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083, 2017. 3
[49] Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. In Advances in Neural Information Processing Systems, pages 7047–7058, 2018. 13
[50] Alexander Meinke and Matthias Hein. Towards neural networks that provably know when they don’t know. arXiv preprint arXiv:1909.12180, 2019. 13
[51] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In International conference on machine learning, pages 3481–3490. PMLR, 2018. 6
and energy-based models. In *International Conference on Learning Representations*, 2021.

[82] Jianwen Xie, Yang Lu, Ruiqi Gao, Song-Chun Zhu, and Ying Nian Wu. Cooperative training of descriptor and generator networks. *IEEE transactions on pattern analysis and machine intelligence*, 42(1):27–45, 2018.

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

[84] Jianwen Xie, Zilong Zheng, and Ping Li. Learning energy-based model with variational auto-encoder as amortized sampler. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*, volume 2, 2021.

[85] Xuwang Yin, Soheil Kolouri, and Gustavo K Rohde. Gat: Generative adversarial training for adversarial example detection and robust classification. In *International Conference on Learning Representations*, 2020.

[86] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient gan training. *arXiv preprint arXiv:2006.10738*, 2020.

[87] Yang Zhao, Jianwen Xie, and Ping Li. Learning energy-based generative models via coarse-to-fine expanding and sampling. In *International Conference on Learning Representations*, 2021.

[88] Zhengli Zhao, Zizhao Zhang, Ting Chen, Sameer Singh, and Han Zhang. Image augmentations for gan training. *arXiv preprint arXiv:2006.02595*, 2020.

[89] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.
A. Related Work on Worst-case Out-of-Distribution Detection

Out-of-distribution (OOD) detection is an important application of EBM. Many generative models, including Glow [39], PixelCNN [57], PixelCNN++ [66], VAEs [38,63], and RealNVP [14], although directly model the density function of the observe data, do not perform well at this task [29,40,53,72]. Apart from methods that employ generative models [8,12,32,40,54,62,71,73], there is also a plethora of methods [1,7,28,46,47,69] that make use of statistics computed from the predictions or intermediate activations of a standard classifier trained on labeled in-distribution data. A widely considered baseline method is Outlier Exposure (OE) [29], a supervised approach that uses a real and diverse out-distribution dataset, specifically the 80 million tiny images dataset [77], to train the OOD detector. More recently, OE and some other OOD detection methods have been found to be vulnerable to out-distribution adversarial examples [3,4,50,70].

B. Proof of Proposition (2)

Proposition 2. The optimal solution of \( \max_D \min_{p_T} U(D, p_T) \) is \( U(D^*, p_T^*) = -\log(4) \), where \( D^* \) outputs \( \frac{1}{2} \) on \( \text{Supp}(p_{\text{data}}) \) and \( \leq \frac{1}{2} \) outside \( \text{Supp}(p_{\text{data}}) \), and \( p_T^* \) has its mass distributed in locations where \( D \) outputs \( \frac{1}{2} \).

Proof. Let

\[
p_T^* = \arg \min_{p_T} E_{x \sim p_T} [\log(1 - D(x))],
\]

then

\[
\max_D \min_{p_T} U(D, p_T) = \max_D U(D, p_T^*).
\]

C.1. Comparative Analysis with GANs

Compared to the maximin game, the minimax game \( \min_{p_T} \max_D U(D, p_T) \) has a reversed rule: player \( p_T \)
makes the first move by choosing a $p_T$; player $D$ then chooses a $D$ to maximize the payoff, which results in a payoff of $\max_D U(D, p_T)$; player $p_T$ knows player $D$’s strategy and will choose a $p_T$ such that the worst case payoff $\max_D U(D, p_T)$ is minimized, which results in an overall payoff of $\min_{p_T} \max_D U(D, p_T)$.

The solution to this minimax game is analyzed in Goodfellow et al. [19]: the best strategy of player $p_T$ is to choose a $p_T^*$ which minimizes the Jensen-Shannon divergence (JSD) between $p_T$ and $p_{\text{data}}$: $p_T^* = \arg \min_{p_T} \text{JSD}(p_T \parallel p_{\text{data}}) = p_{\text{data}}$, and the best strategy of player $D$ is to choose $D^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_T^*(x)} = \frac{1}{2}$. Under these strategies, the payoff function $U$ measures the JSD between $p_T$ and $p_{\text{data}}$: $U(D^*, p_T^*) = -\log(4) + 2 \cdot \text{JSD}(p_T^* \parallel p_{\text{data}}) = -\log(4)$, which coincides with the $U$ solution in the maximin game (Proposition 2). Noted that $D^*$ does not need to be defined outside of $\text{Supp}(p_T) \cup \text{Supp}(p_{\text{data}})$ [19].

Algorithm 3 (adapted from GANs training algorithm) outlines the pseudo code for solving the minimax problem. Fig. 4 shows that when the algorithm converges, $p_T^*$ samples overlap $p_{\text{data}}$ samples. The bottom panel shows that when $p_0$ data is concentrated in the bottom left corner, the $D$ solution loses the gradient information for translating $p_0$ samples to the support of $p_{\text{data}}$.

The main differences between the maximin game solution and minimax game solution can be summarized as:

- **$p_T^*$ difference** In the minimax game, $p_T^* = p_{\text{data}}$, whereas in the maximin game, mass of $p_T^*$ can be at any location(s) where $D$ outputs $\frac{1}{2}$.

- **$D^*$ difference** While both $D^*$’s output $\frac{1}{2}$ in $\text{Supp}(p_{\text{data}})$, their outputs outside $\text{Supp}(p_{\text{data}})$ are different: in the maximin game, $D^*$ outputs $\leq \frac{1}{2}$ outside $\text{Supp}(p_{\text{data}})$, whereas in the maximin game, $D^*$ is undefined outside $\text{Supp}(p_{\text{data}})$.

Due to the above differences these two formulations give rise to different applications. The minimax formulation, which is the formulation used by GANs, is ideal for learning a generator model that recovers $p_{\text{data}}$. The discriminator, because of its undefined behavior outside the support of $p_{\text{data}}$, may not be very useful for downstream tasks (in particular, it cannot be used for discriminating between in-distribution and out-distribution data). In the maximin formulation, when the model is trained with a real and diverse $p_0$ dataset, we can obtain a $D$ solution in which local maxima are suppressed and therefore can be used for out-of-distribution detection. The $D$ solution at the same time retains informative gradient for translating out-distribution samples to the support of $p_{\text{data}}$, and thus can be used in generation, image-to-image translation, and various image restoration tasks.

**Algorithm 3** Solving the minimax problem

1: Draw samples $\{x_i\}_{i=1}^m$ from $p_{\text{data}}$, and samples $\{x_i^*\}_{i=1}^m$ from $p_0$.
2: repeat
3: Update $D$ by maximizing $\frac{1}{m} \sum_{i=1}^m \log D(x_i) + \frac{1}{m} \sum_{i=1}^m \log(1 - D(x_i^*))$ (until converge).
4: For each $x \in \{x_i^*\}_{i=1}^m$, update its value by $x \leftarrow x + \lambda \nabla D(x)$ (single step).
5: until $\{x_i^*\}_{i=1}^m = \{x_i\}_{i=1}^m$

---

**Figure 4.** Plots of contours and (normalized) gradient vector fields of the $D$ functions learned with Algorithm 3. Left images: $p_0$ data (red points) and initial states of the $D$ functions. Right images: $p_T^*$ data (red points) and the final states of the $D$ functions when the algorithm converges.

**D. Ablation on Progressive Training**

We perform an ablation study of the progressive training algorithm on CIFAR-10. Fig. 5 shows that with the same number of training iterations, progressive training achieves a higher AUROC score with less amount of time. The AUROC score measures the $D$ model’s ability to discriminate between $p_{\text{data}}$ samples and perturbed $p_0$ samples. Hence the score decreases each time we increase $K$ which causes larger perturbations on $p_0$ samples. Fig. 5 shows that at the end of training when $K = 9$, progressive training’s AUROC scores are higher than that of fixed-steps training.
Figure 5. AUROC curves of progressive training and training with a fixed number of steps.

E. Experimental Setups

Model architecture On CIFAR-10 we use the standard ResNet50 [26] architecture with ReLU activation for the $D$ model. On CelebA-HQ 256 and AFHQ-CAT 256 we use a customized architecture adapted from Choi et al. [9] (Tab. 9).

Datasets We evaluate our method on CIFAR-10 [41] (50K training samples), CelebA-HQ 256 [35] (30K training samples), and the AFHQ-CAT [9] dataset (5153 training samples) which is a recently introduced benchmark dataset for image-to-image translation.

Evaluation metrics We use Inception Score (IS) [65] and FID score [30] to assess the quality of generated samples. We follow Karras et al. [36] and compute the FID score between 50K generated samples and all training samples (IS is also calculated on the generated 50K samples). We use the original code from Salimans et al. [65] and Heusel et al. [30] to calculate the scores. For OOD detection, we use area under the ROC curve (AUROC) as the evaluation metric.

Training We use Algorithm 2 to train the generative models on CIFAR-10, CelebA-HQ 256, and AFHQ-CAT 256.

The hyperparameter setting and training schedule for each task can be found in Tab. 10 and Tab. 11. We in addition perform 5-steps PGD attack, random resized cropping, and random horizontal flipping on $p_{data}$ samples to mitigate overfitting. The performance (FID score) of the model is monitored during training and the best-performing model to used to report the final FID score.

The CIFAR-10 worst-case OOD detection model is trained using in- and out-distribution adversarial training [3], where in-distribution AT uses a $l^2$-ball of radius 0.25 and PGD attacks of steps 10 and step-size 0.1, and out-distribution AT uses a $l^2$-ball of radius 0.5 and PGD attacks of steps 10 and step-size 0.1. Following Augustin et al. [3], we use a batch size of 128 and use the recommended AutoAugment policy from [11]. The model is trained for 400 epochs using a SGD optimizer with a constant learning rate of 0.1. A separate validation set split from the 80 million tiny images is used for selecting the model for final evaluation.

| Task         | Training schedule                                                                 |
|--------------|-----------------------------------------------------------------------------------|
| CIFAR-10     | $K = 0, 1, \ldots, 25$; train 5 epochs for each $K$, and then continue training with $K = 25$ |
| CelebA-HQ 256| $K = 0, 1, \ldots, 35$; train 5 epochs for each $K$, and then continue training with $K = 35$ |
| AFHQ-CAT 256 | $K = 0, 1, \ldots, 25$; train 50 epochs for each $K$, and then continue training with $K = 25$ |

Table 11. The training schedule for each task.

| Task         | $p_0$ dataset | PGD step size | PGD steps |
|--------------|---------------|---------------|-----------|
| CIFAR-10     | 80 million tiny images [77] | 0.2 | 32 |
| CelebA-HQ 256| ImageNet [13] | 8.0 | 20 |
| AFHQ-CAT 256 | ImageNet [13] | 8.0 | 14 |

Table 12. Sample generation settings.

Sample generation The generated samples for FID and IS evaluation are produced by performing PGD attacks on 50K samples randomly drawn from the $p_0$ dataset. The settings
for the $p_0$ dataset and the PGD attack can be found in Table 12.

F. Extended Experiment Results

Worst-case out-of-distribution detection on CelebA-HQ 256 and AFHQ-CAT 256. Tab. 13 shows that under a PGD adversary with $l^2$ radius 7.0 our model exhibits strong out-distribution robustness. Note that a perturbation of 7.0 is already large enough to cause an undefended model (e.g., OE [29]) to fail completely at the OOD detection task [3]. When we further increase the perturbation to 100, the AUC score decreases to near 0, suggesting that obfuscated gradients did not occur.

| Threat model | Out-distribution dataset | In-distribution dataset |
|--------------|--------------------------|-------------------------|
|              | CelebA-HQ 256            | AFHQ-CAT 256            |
| Uniform noise | 1.0 / 1.0                | 1.0 / 1.0               |
| SVHN          | 0.9955 / 0.9920          | 0.9944 / 0.9889         |
| $l^2$ ball of radius 7.0 |
| CIFAR-10     | 0.9966 / 0.9979          | 0.9930 / 0.9902         |
| ImageNet validation set |
| 0.9979 / 0.9963 | 0.9971 / 0.9945   |
| $l^2$ ball of radius 100.0 |
| CIFAR-10     | 0.9966 / 0.0787          | 0.9930 / 0.0042         |
| ImageNet validation set |
| 0.9979 / 0.1277 | 0.9971 / 0.0131   |
| $l^2$ ball of radius 100.0 |
| CIFAR-10     | 0.9964 / 0.0765          | N/A                     |
| AFHQ-CAT 256 | 0.9964 / 0.0765          | N/A                     |
| CelebA-HQ 256 | N/A                     | 0.9900 / 0.9810         |
| AFHQ-CAT 256 | N/A                     | 0.9900 / 0.0023         |

Table 13. CelebA-HQ 256 and AFHQ-CAT 256 OOD detection results. Each entry shows the AUC score on clean OOD samples (left value) and AUC score on adversarial OOD samples (right value). Adversarial OOD samples are computed by maximizing the model output in a $l^2$-ball of radius 7.0 or 100.0 (image data range is [0, 1]) around OOD samples via Auto-PGD [10] with 100 steps and 5 random restarts. We use 1024 in-distribution samples and 1024 out-distribution samples to compute the results.

Additional results are summarized below:

- **Image-translation** Fig. 7 and Fig. 8 show uncurated image translation results on CelebA-HQ 256 and AFHQ-CAT 256.

- **Denosing and inpainting** Fig. 9, Fig. 10, and Fig. 11 show uncurated denoising and inpainting results on CelebA-HQ 256, FFHQ [37] and AFHQ-CAT 256.

- **Uncurated generation samples** Fig. 6a, Fig. 12, and Fig. 13 show the uncurated generated samples on CIFAR-10, CelebA-HQ 256, and AFHQ-CAT 256.

- **Interpolation** Fig. 18 and Fig. 19 show the interpolation results on CelebA-HQ 256 and AFHQ-CAT 256. The interpolation works reasonable well even on AFHQ-CAT which has only about 5000 training images.

- **Nearest Neighbor Analysis** Fig. 6b and Fig. 6c, Fig. 15 and Fig. 14, Fig. 17 and Fig. 16 respectively show the nearest neighbors (in pixel space and Inception feature space) of the generated samples on CIFAR-10, CelebA-HQ 256 and AFHQ-CAT 256. Note that none of the nearest neighbors resemble the generated samples, suggesting that the models did not memorize the training data.

- **Long-run PGD attack** Fig. 20 and Fig. 21 show the intermediate results from 200 steps of PGD attacks. It can be seen that at step 200, the generated samples become saturated and do not resemble real faces. In addition, some of these samples look similar to each other, suggesting that some PGD attacks have converged to the same point. These results indicate that the model do not have a valid steady-state that reflects the distribution of the observed data.

Figure 6. Uncurated CIFAR-10 generated sample and nearest neighbors of some of the generated samples.
Figure 7. Uncurated image translation results on CelebA-HQ 256.

Figure 8. Uncurated image translation results on AFHQ-CAT 256.
Figure 9. Uncurated denoising and inpainting results on CelebA-HQ 256. The source images are from CelebA-HQ training set.

Figure 10. Uncurated denoising and inpainting results on FFHQ dataset. The source images are from FFHQ dataset [37].

Figure 11. Uncurated denoising and inpainting results on AFHQ-CAT 256. The source images are from AFHQ-CAT training set.
Figure 12. Uncurated generated samples on CelebAHQ-256.
Figure 13. Uncurated generated samples on AFHQ-CAT 256.
Figure 14. Pixel space nearest neighbors of generated images on CelebA-HQ 256. The left panel shows the generated samples and the right panel shows the corresponding top-5 nearest neighbors.

Figure 15. Inception space nearest neighbors of generated images on CelebA-HQ 256. The left panel shows the generated samples and the right panel shows the corresponding top-5 nearest neighbors.
Figure 16. Pixel space nearest neighbors of generated images on AFHQ-CAT 256. The left panel shows the generated samples and the right panel shows the corresponding top-5 nearest neighbors.

Figure 17. Inception space nearest neighbors of generated images on AFHQ-CAT 256. The left panel shows the generated samples and the right panel shows the corresponding top-5 nearest neighbors.
Figure 18. Interpolation results on CelebA-HQ 256. Intermediate images are generated by performing PGD attacks on linear interpolations between the source images used to generate the leftmost and rightmost samples.
Figure 19. Interpolation results on AFHQ-CAT 256. Intermediate images are generated by performing PGD attacks on linear interpolations between the source images used to generate the leftmost and rightmost samples.
Figure 20. Celeb-AHQ 256 samples generated by long-run PGD attacks. The attack steps for column 1-7 are [0, 6, 13, 19, 26, 33, 200].

Figure 21. AFHQ-CAT 256 samples generated by long-run PGD attacks. The attack steps for column 1-7 are [0, 6, 13, 19, 26, 33, 200].