Towards Understanding the Generative Capability of Adversarially Robust Classifiers

Yao Zhu\textsuperscript{1}, Jiacheng Ma\textsuperscript{2}, Jiacheng Sun\textsuperscript{2†}, Zewei Chen\textsuperscript{2}, Rongxin Jiang\textsuperscript{1}, Zhenguo Li\textsuperscript{2}

\textsuperscript{1}Zhejiang University, \textsuperscript{2}Huawei Noah’s Ark Lab

Abstract

Recently, some works found an interesting phenomenon that adversarially robust classifiers can generate good images comparable to generative models. We investigate this phenomenon from an energy perspective and provide a novel explanation. We reformulate adversarial example generation, adversarial training, and image generation in terms of an energy function. We find that adversarial training contributes to obtaining an energy function that is flat and has low energy around the real data, which is the key for generative capability. Based on our new understanding, we further propose a better adversarial training method, Joint Energy Adversarial Training (JEAT), which can generate high-quality images and achieve new state-of-the-art robustness under a wide range of attacks. The Inception Score of the images (CIFAR-10) generated by JEAT is 8.80, much better than original robust classifiers (7.50).

1. Introduction

Adversarial training can improve the robustness of a classifier against adversarial perturbations imperceptible to humans. Unlike a normal classifier, an adversarially robust classifier can generate good images by gradient descending the cross-entropy loss from random noises. Recently, some works discovered this phenomenon, and the quality of generated images is even comparable to GANs \cite{22, 7}. The generative capability of an adversarially robust model from a classification task is interesting and surprising. However, it is unclear why an adversarially trained classifier can generate natural images. As image generation is a crucial topic, understanding the generative capability of adversarially robust classifiers could be inspiring and can give hints to many other generative methods.

In this paper, we aim to understand the generative capability of an adversarially trained classifier and further improve the quality of generated images from the energy perspective. For Energy-Based Model (EBM) \cite{10}, it first generates low energy samples from random noises, then increases the energy of generated samples by updating model parameters. In this way, EBM can obtain a good energy function that is smooth and has low energy near the real data, as illustrates in Fig. [a]. Thus, EBM can generate good images by sampling with Langevin Dynamics with the good energy function [5].

We show that adversarially trained classifiers can also obtain such a good energy function, which is flat and has low energy near the real data. This implies the generative capability of the adversarially robust classifier. For a classifier, we can define energy functions on the output logits. We reformulate the original adversarial training and image generation in terms of energy functions. In fact, the adversarial examples are high-energy samples near the real data, and adversarial training is trying to decrease the energy of these examples by updating model parameters. This procedure can also help us to learn a flat energy function with low energy near the real data, as illustrated in Fig. [b]. Moreover, based on our understanding, we find another interesting phenomenon that a normal classifier is able to generate images if we add random noise to images during training. Though injecting noises to training data can generate high-energy samples, the perturbation direction may not be efficient compared with the adversarial attack. Thus, larger noise is needed in training for generating high-energy samples.

The cross-entropy loss can be expressed as the difference between $E_\theta(x, y)$ and $E_\theta(x)$. We show that $E_\theta(x, y)$, which is flat and has lower energy around the real data, plays a key role in the conditional generation task. However, making the energy $E_\theta(x, y)$ flat and low near the real data is just a by-product of original adversarial training. Thus we propose Joint Energy Adversarial Training (JEAT) to directly optimize the $E_\theta(x, y)$. We generate adversarial...
examples by directly increasing the $E_\theta(x, y)$ as Eq. \textcolor{red}{18} and update model parameters by maximizing the likelihood of joint distribution $p_\theta(x, y)$ as Eq. \textcolor{red}{19}. We show that JEAT further improves the quality of the generated images.

The main contributions of our paper are summarized below:

- We propose a novel explanation of the image generation capability of a robust classifier from an energy perspective.
- We find the generative capability of normal classifiers with injecting noise in the training process and explain it from the energy point of view.
- We propose a training algorithm JEAT from an energy perspective that improves image generative capability.

2. Preliminaries

2.1. Adversarial Training

Adversarial training, first proposed in \textcolor{red}{8}, can effectively defend against adversarial attacks by solving a bi-level min-max optimization problem \textcolor{red}{19}. It can be formulated as:

\[
\begin{align*}
\min_\delta & \mathbb{E}_{(x, y) \sim D} [\mathcal{L}(x + \delta, y; \theta)], \\
\delta^* & = \arg \max_\delta \|\delta\|_p, \mathcal{L}(x + \delta, y; \theta),
\end{align*}
\]

where $y$ is the ground-truth label of the input $x$, $\delta^*$ denotes the adversarial perturbation added to $x$, $\mathcal{L}$ denotes the loss function, $\|\cdot\|_p$ denotes the $\ell_p$-norm that constrains the perturbed sample in an $\ell_p$-ball with radius $\epsilon$ centered at $x$.

The $\ell_\infty$ adversarial perturbation is usually approximately solved by the fast gradient sign method (FGSM) \textcolor{red}{8} and projected gradient descent (PGD) \textcolor{red}{19}. For FGSM attacks $\delta_x$ takes the form:

\[
\delta_x = \epsilon \text{sign}(\nabla_x \mathcal{L}(x, y; \theta)),
\]

where $\text{sign}$ is the sign function. PGD attack is a kind of iterative variant of FGSM which generates an adversarial sample starting from a random position in the neighborhood of the clean image. PGD can be formulated as:

\[
\begin{align*}
x_n &= x_{n-1} + \eta \cdot \text{sign}(\nabla_{x_{n-1}} \mathcal{L}(x_{n-1}, y; \theta)), \\
x_n &= \text{clip}(x_n, x_0 - \epsilon, x_0 + \epsilon),
\end{align*}
\]

where $x_0$ is a clean image and $\eta$ is the perturbation step.

2.2. Image Generation of Robust Model

Generating images with an adversarially trained classifier is first raised by Santurkar, Madry, etc. \textcolor{red}{21} where they demonstrate that a robust classifier alone suffices to tackle various image synthesis tasks such as generation. For a given label $y$, image generation minimizes the loss $\mathcal{L}$ of label $y$ by:

\[
x' = x - \frac{\eta}{T} \cdot \nabla_x \mathcal{L}(x, y; \theta) + \sqrt{T} \epsilon.
\]

Starting from sample $x_0 \sim \mathcal{N}(\mu_y, \Sigma_y)$ where $\mu_y = \mathbb{E}_{x \sim P_{x,y}}(x)$ and $\Sigma_y = \mathbb{E}_{x \sim P_{x,y}}((x - \mu_y)^T(x - \mu_y))$, we can obtain a better image by minimizing the loss $\mathcal{L}$. Some simple improvements such as choosing a more diverse distribution to start with could further improve the quality of generated images \textcolor{red}{21}. However, it is not the focus of this paper.

2.3. Energy Based Model

Energy Based Models \textcolor{red}{17} show that any probability density $p(x)$ for $x$ can be expressed as:

\[
p_\theta(x) = \frac{\exp(-E_\theta(x))}{Z_\theta},
\]

where $E_\theta(x)$ represents the energy of $x$ and is modeled by neural network, $Z_\theta = \int \exp(-E_\theta(x)) dx$ is the normalizing factor parameterized by $\theta$ also known as the partition function. The optimization of the energy-based model is through maximum likelihood learning by minimizing $\mathcal{L}_{ML}(\theta) = \mathbb{E}_{x \sim P_{D}}[-\log p_\theta(x)]$. The gradient is \textcolor{red}{5}:

\[
\nabla_\theta \mathcal{L}_{ML}(\theta) = \mathbb{E}_{x \sim P_{D}}[\nabla_\theta E_\theta(x^+)] - \mathbb{E}_{x \sim P_{\theta}}[\nabla_\theta E_\theta(x^-)].
\]
Because sampling from $p_\theta$ is not feasible for an EBM, Langevin Dynamics is commonly used to approximately find samples from $p_\theta$ \footnote{Langevin Dynamics could generate samples through Langevin Dynamics.}. Thus, EBM training usually contains two stages: approximately generating samples from $p_\theta$ by Langevin Dynamics along the direction of energy descent and optimizing model parameters to increase the energy of these samples and decrease the energy of real samples by SGD. In this way, as illustrated in Fig. \ref{fig:energy_contour}(a), EBM could obtain a smooth energy function around real data and generate samples through Langevin Dynamics.

From energy perspective, we can also define $p_\theta(x, y)$ as follows:

$$p_\theta(x, y) = \frac{\exp(-E_\theta(x, y))}{\hat{Z}_\theta}, \tag{7}$$

where $\hat{Z}_\theta = \int \sum_y \exp(-E_\theta(x, y))dx$. Thus we also get $p_\theta(y|x)$ by $E_\theta(x, y)$ and $E_\theta(x, y)$:

$$p_\theta(y|x) = \frac{p_\theta(x, y)}{p_\theta(x)} = \frac{\exp(-E_\theta(x, y)) \cdot \hat{Z}_\theta}{\exp(-E_\theta(x)) \cdot \hat{Z}_\theta}. \tag{8}$$

### 2.4. Sampling with Langevin Dynamics

Using $\nabla_x \log(p(x))$, Langevin Dynamics could generate samples from density distribution $p(x)$. The process starts from an initial point $\tilde{x}_0$ from a prior distribution $\pi$ and recursively updates $\tilde{x}$ by:

$$\tilde{x}_t = \tilde{x}_{t-1} + \frac{\eta}{2} \nabla_{\tilde{x}_{t-1}} \log(p(\tilde{x}_{t-1})) + \sqrt{\eta} \epsilon, \tag{9}$$

where $\epsilon \sim \mathcal{N}(0, I)$ and $\eta$ is a fixed step size. When $\eta \to 0$ and $T \to \infty$, $\tilde{x}_T$ is exactly an sample from $p(x)$ under some condition \footnote{We denote $f(; \theta)$ as a classification neural network parameterized by $\theta$. Let $x$ be a sample. Then $f(x; \theta)[k]$ represents the $k^{th}$ output of the last layer and we define $p_\theta(y|x)$ as: $p_\theta(y|x) = \frac{\exp(f(x; \theta)[y])}{\sum_{k=1}^{n} \exp(f(x; \theta)[k])}$, which resembles Boltzmann distribution, and $n$ represents total possible classes. From Eq. \ref{eq:energy} and \ref{eq:energy_total}, we define two energy functions as follows: $E_\theta(x, y) = -\log(\exp(f(x; \theta)[y]))$, $E_\theta(x) = -\log(\sum_{k=1}^{n} \exp(f(x; \theta)[k]))$. From Eq. \ref{eq:energy_total}, we have $\hat{Z}_\theta = Z_\theta$. Furthermore, if the classification loss function is cross-entropy loss, it could also be expressed as: $\mathcal{L}(x, y; \theta) = E_\theta(x, y) - E_\theta(x)$. In original adversarial training as in \cite{goodfellow2014explaining}, by Fast Gradient Sign Method, adversarial example could be found by $x_{adv} = x + \eta \cdot \text{sign}(\nabla_x (E_\theta(x, y) - E_\theta(x)))$, which is the direction of gradient ascent of loss defined in Eq. \ref{eq:adversarial}. The direction of adversarial perturbation relates to not only $E_\theta(x, y)$ but also $E_\theta(x)$. And in original adversarial training, the optimization process aims to decrease the loss in Eq. \ref{eq:adversarial} by updating model parameters. As we mentioned in Sec. \ref{sec:adversarial}, a good energy function $E_\theta(x, y)$ which is smooth and has low energy around real data is the key factor for generating good images for a given label $y$. We define the change of energy $E_\theta(x, y)$ after adversarial attack and after optimization as: $\Delta_x E_\theta(x, y) = E_\theta(x_{adv}, y) - E_\theta(x, y)$, $\Delta_\theta E_\theta(x, y) = E_\theta_{update}(x_{adv}, y) - E_\theta(x_{adv}, y)$,}.

$$\tilde{x}_t = \tilde{x}_{t-1} - \frac{\eta}{2} \nabla_{\tilde{x}_{t-1}} E_\theta(\tilde{x}_{t-1}, y) + \sqrt{\eta} \epsilon. \tag{10}$$
where \( E_\theta(x_{\text{adv}}, y) \) is the energy of \( x_{\text{adv}} \) given label \( y \) before updating parameters, \( E_\theta(x_{\text{ori}}, y) \) is the energy of \( x_{\text{ori}} \) given label \( y \) before updating parameters, and \( E_{\theta_{\text{updated}}}(x_{\text{adv}}, y) \) is the energy of \( x_{\text{adv}} \) given label \( y \) after updating parameters. The definition of \( \Delta_x E_\theta(x) \) and \( \Delta_y E_\theta(x) \) are similar in [15] with removing \( y \).

As illustrated in Fig. 3(a), adversarial attacks generate high-energy adversarial examples, and then the energy of adversarial examples is decreased by updating model parameters. Compared to EBM, adversarial training flattens the energy region around real data in a way different from EBM training. Adversarial training finds adversarial examples with high energy \( E_\theta(x, y) \) near the data and then decreases the energy of those samples by updating model parameters. Noted that, in EBM, sampling from \( p_\theta(x, y) \) by Langevin Dynamics follows the direction of energy descent, which is the opposite direction of adversarial examples. As illustrated in Fig. 1(b), adversarial training can obtain a flat energy function around real data.

We illustrate energy \( E_\theta(x, y) \) contours in Fig. 2 for a normal classifier and an adversarially trained classifier. From Fig. 2(a), the energy function around the center is sharp for a normal classifier, and the energy of the real data is high. Because the low-energy region deviates from the center in a normal classifier, the generated images along the direction of energy descent are likely to fall into a region far away from the real data, which would not have good quality. For a robust classifier, the energy function near the center is smoother and lower, as shown in Fig. 2(b).

Image generation by a robust classifier is introduced in Sec. 2.2 if we transform the loss function to energy expression as Eq. (13), the generating procedure of images becomes:

\[
x' = x - \frac{\eta}{2} \cdot \nabla_x (E_\theta(x, y) - E_\theta(x)) + \sqrt{\eta} \epsilon,
\]

where \( \epsilon \sim N(0, I) \) and \( \eta \) is step size. The term \( E_\theta(x, y) - E_\theta(x) \) in Eq. (16) implies that the generation iterations are related to both the energy of \( E_\theta(x, y) \) and the energy of \( E_\theta(x) \). By exploring low energy region of \( E_\theta(x, y) \), which corresponds to high probability \( p_\theta(x, y) \) region given a constant normalizing factor from Eq. (7), we could obtain a good sample from high probability \( p_\theta(x, y) \) region.

Minimizing \( E_\theta(x, y) \) contributes to maximizing \( p_\theta(x, y) \) to generate images corresponding to the label \( y \) as shown in Eq. (10), but the energy of \( E_\theta(x) \) is irrelevant to label \( y \).

As illustrated in Fig. 3(b), \( E_\theta(x) \) has not been well optimized in the classification task, which may introduce label-independent noise. Thus, \( E_\theta(x) \) may be a factor restricting the generative capability of the robust classifier, and we could drop \( E_\theta(x) \) while only using \( E_\theta(x, y) \) as:

\[
x' = x - \frac{\eta}{2} \cdot \nabla_x (E_\theta(x, y)) + \sqrt{\eta} \epsilon.
\]

Eq. (17) is closely relate to sampling from Langevin Dynamics as Eq. (10). Approaching low energy region of \( E_\theta(x, y) \) is equivalent to approaching the high probability region of \( p_\theta(x, y) \). As shown in Fig. 4(a) and (b), using Eq. (17) gives better generated images than using Eq. (16).

Figure 3. We illustrate the changes of energy in original adversarial training [19] on CIFAR-10 in 50 epochs (model has converged). The center points of the tags represent the mean value and the lengths represent the variance. (a) Adversarial examples increase the energy \( E_\theta(x, y) \) as \( \Delta_x E_\theta(x, y) > 0 \) during the training. The energy \( E_\theta(x, y) \) of adversarial examples decrease after updating parameters as \( \Delta_y E_\theta(x, y) < 0 \) during the training. (b) The value of \( \Delta_y E_\theta(x) \) fluctuates around zero, sometimes positive and sometimes negative. Thus \( E_\theta(x) \) has not been well optimized in the classification task.

Figure 4. Given the label ship, the images generated by different methods. (a) shows ten images generated by original robust classifier using Eq. (16). (b) shows ten images generated by original robust classifier using Eq. (17). (c) shows ten images generated by robust classifier trained with adversarial examples got by Eq. (18) using Eq. (17).

3.2. Endowing Generative Capability to Normal Classifier

As analyzed before, we can generate high-energy samples from real data and reduce the energy of these samples during the optimization process to obtain a classifier with generative capability. The process of optimizing the energy \( E_\theta(x, y) \) to be flat and low around real data contributes to generating good images. Adversarial training follows the
Figure 5. The images generated by the models injected different strengths of noise in the training process. The clean accuracy of these models is almost the same. We use $\delta$ to denote the noise added to training data. (a) $\delta \sim \mathcal{N}(0, 24/255)$. Obviously, this model produces recognizable images. (b) $\delta \sim \mathcal{N}(0, 16/255)$, the shape of ship is visible. (c) $\delta \sim \mathcal{N}(0, 8/255)$, it is hard to recognize the shape of objects.

procedure and provides an effective way to find high-energy samples with small perturbations. Meanwhile, a simple random noise could also find high-energy samples but less effectively.

To verify our claim, we train normal classifiers by injecting different strengths of random noise into training data. As shown in Fig. 5, classifiers trained with different strengths of noise generate images of different qualities through the procedure in Sec. 3.3. Though the quality is not good enough, cars, birds and ships can be recognized with high confidence in Fig. 5(a). From the comparison of images generated by adding different strengths of noise, we find that stronger random noise is needed to train a normal classifier in order to generate recognizable images. This is consistent with our explanation above that random noise is a “blind” adversarial direction to find high-energy samples. Because the direction is uncertain and very likely not the direction to efficiently increase energy, stronger perturbations may be needed. This experiment also shows that the process which generates high-energy samples from real data and reduces the energy of these samples during the optimization contributes to obtaining a classifier with generative capability.

3.3. Generating Adversarial Examples by Energy

As mentioned above, $E_\theta(x, y)$ determines the generative capability of a robust classifier. In original adversarial training, it does not increase the energy $E_\theta(x, y)$ of original data directly. In fact, only using $E_\theta(x, y)$ can also make $x$ to be an adversarial sample $x_{adv}$, as shown in Tab. 1.

Using $E_\theta(x, y)$ only in Eq. (14), we have:

$$x_{adv} = x + \eta \cdot \text{sign}(\nabla_x(E_\theta(x, y))).$$  \hspace{1cm} (18)

Adversarial examples generated by energy attack in Eq. (18) have almost the same attack effect with the original normal attack by Eq. (14). The difference between $x$ and $x_{adv}$ is unrecognizable, but their energy are quite different. We name the method which trained with adversarial examples found by Eq. (18) and generates images by Eq. (17) after training as Preliminary Joint Energy Adversarial Training (PreJEAT).

As we illustrate in Fig. 2 in a PreJEAT trained classifier, the energy around the natural image (center) is flatter and lower than a normal classifier and original robust classifier. In a robust classifier, the low energy region deviates from the center. But in the PreJEAT trained classifier, the natural image has the lowest energy. Thus, directly generating adversarial examples by $E_\theta(x, y)$ contributes to obtaining a flat and low energy function near the real data. By adversarial training with Eq. (18) we can obtain a robust classifier with better generative capability, as shown in Fig. 4(c).

3.4. Joint Energy Adversarial Training

We verified that a PreJEAT trained model has good generative capability in the previous section. However, there is still a discrepancy between training loss objective as Eq. (13) and adversarial training as Eq. (18) in PreJEAT. And we also find that the images generated by PreJEAT are not smooth enough in Fig. 4(c). We hope that the optimization process also optimizes $E_\theta(x, y)$ more directly to reduce the energy around real data. Hence we propose a new algorithm, Joint Energy Adversarial Training (JEAT), in Algorithm 1 to improve the generative capability of a classifier. In JEAT, we replace cross-entropy loss $-\log p_\theta(y|x)$ in PreJEAT with:

$$-\log p_\theta(x, y) = -\log p_\theta(y|x) - \log p_\theta(x),$$  \hspace{1cm} (19)

where $p_\theta(x)$ is defined in Eq. (5). Optimizing $p_\theta(x, y)$ helps to get better $E_\theta(x, y)$ as shown in Eq. (7). The gradient of $\log p_\theta(x)$ is

$$\nabla_\theta \log(p_\theta(x)) = -\nabla_\theta E_\theta(x) - E_{p_\theta(x)}[\nabla_\theta E_\theta(x)].$$  \hspace{1cm} (20)

We use Stochastic Gradient Langevin Dynamics (SGLD) to approximate $E_{p_\theta(x)}$.

JEAT uses Eq. (18) to find adversarial example and Eq. (17) to generate image like PreJEAT. With our proposed JEAT, adversarial examples, adversarial training, and image generation are connected to the energy $E_\theta(x, y)$ in a clear way. We also plot the energy contour of $E_\theta(x, y)$ in Fig. 2(c). Compared to the other three models, the energy function of a JEAT trained classifier near the center is the

| Attack         | CIFAR-10 | CIFAR-100 |
|----------------|----------|-----------|
| No Attack      | 94.39%   | 78.54%    |
| Normal Attack  | 0.12%    | 0.08%     |
| Energy Attack  | 0.11%    | 0.10%     |
4. Experiments

4.1. Experimental settings

To verify the validity of our energy-based explanation and the effectiveness of JEA T, we conducted experiments on CIFAR-10, CIFAR-100 and compare them with other methods. For network architecture, we use the WideResNet-28-10 for all algorithms. We repeat our experiments five times and report the mean and standard deviation.

4.2. Image Generation

We use the procedure described in Sec. 3.1 to generate images by using different classifiers. The images generated by the adversarially trained classifier are shown in Fig. 6(a). As analyzed in Sec. 3.1, $E_{\theta}(x)$ can be the factor that restricts the generative capability of a robust model. We show the images generated by PreJEAT which uses $E_{\theta}(x, y)$ instead of $E_{\theta}(x, y) - E_{\theta}(x)$ both in adversarial training and generating images in Fig. 6(b). This approach greatly improves the quality of generated images. The energy plots of classifiers are presented in Sec. 3. By using PreJEAT, the energy is smoother and has a larger low energy region around real data, and it generates better quality images, especially in fine details and overall shape.

We also show the images generated by JEM [10] which is an energy-based model utilizing a classifier in Fig. 6(c). Compared to Fig. 6(b), JEM generates a more diverse background and is smooth in pixel. However, the features of objects stand out and are more perceptible to human in the images generated by PreJEAT.

As is shown in Fig. 6(d), our JEA T algorithm generates the best images compared to the other three. The reason behind is that JEA T is adversarially trained with energy $E_{\theta}(x, y)$ as in Eq. (18), and directly optimize $E_{\theta}(x, y)$ in the training contributes to getting better $E_{\theta}(x, y)$. The classifier trained by JEA T generates natural images with abundant features. Moreover, JEA T has the best performance based on the metrics of the Inception Score and Frechet Inception Distance Distance metrics.

4.3. Score Matching Metric

“Score” is defined here as $s_{\theta}(x, y) = \nabla_x \log p_{\theta}(x, y)$ [26]. Score matching tries to match the vector field of the gradient of $\log p_{\theta}(x, y)$ with respect to $x$ to the real vector field given in data distribution $p_{data}$ by minimizing Fisher Divergence:

$$\frac{1}{2} \mathbb{E}_{p_{data}(x|y)}[||\nabla_x \log p_{\theta}(x, y) - \nabla_x \log p_{data}(x, y)||^2_2].$$

(21)

With the score defined in the distribution of $(x, y)$ as $s_{\theta}(x, y)$, an equivalent optimization problem to (21) is min-
imizing

\[
E_{p_{\text{data}}(x|y)} \left[ \frac{1}{2} \cdot \left\| s_{\theta}(x, y) \right\|_2^2 + tr(\nabla_x s_{\theta}(x, y)) \right],
\]

(22)

where \( \nabla_x s_{\theta}(x, y) \) denotes the Jacobian of \( s_{\theta}(x, y) \) \[14\]. From energy perspective, score \( s_{\theta}(x, y) = -\nabla_x E_{\theta}(x, y) \). Hence we denote objective in Eq. (22) as Joint Fisher Score (FS):

\[
E_{p_{\text{data}}(x|y)} \left[ \frac{1}{2} \cdot \left\| \nabla_x E_{\theta}(x, y) \right\|_2^2 - tr(\nabla_x^2 E_{\theta}(x, y)) \right].
\]

(23)

It is a simple and effective metric that a lower value indicates \( \nabla_x \log p_{\theta}(x, y) \) is closer to \( \nabla_x \log p_{\text{data}}(x, y) \). So smaller value of Fisher Score corresponds to better generative capability (not considering failure cases here). Fisher Score could be used as a metric to assess the generative capability of models. Similarly, we denote Marginal Fisher Score (MFS) as:

\[
E_{p_{\text{data}}(x)} \left[ \frac{1}{2} \cdot \left\| \nabla_x E_{\theta}(x) \right\|_2^2 - tr(\nabla_x^2 E_{\theta}(x)) \right].
\]

(24)

As shown in Tab. 2 both the Joint Fisher Score and the Marginal Fisher Score for our JEAT are the smallest in all the four models, which implies that the score of our model matches well with the ground truth distribution. This also means that JEAT can generate better images using Langevin Dynamics. The score matching metric is also consistent with the results of IS and FID, which are commonly used.

Table 2. Metrics for different models. “JFS” and “MFS” are the scores to evaluate the Joint and marginal Fisher Scores. “Normal” denotes normal training. “AT” denotes adversarial training.

| Method     | Normal | AT [21] | JEM [10] | JEAT |
|------------|--------|---------|----------|------|
| JFS \downarrow | 6.97   | 2.83    | 0.330    | 0.023|
| MFS \downarrow | 23.08  | 3.12    | 1.44     | 0.15 |
| IS \uparrow  | -      | 7.50    | 8.66     | 8.80 |
| FID \uparrow | -      | 60.90   | 38.40    | 38.24|
4.4. Robustness of JEAT

In this section, we show that the JEAT models not only can generate good images but also has comparable robustness with other hybrid models. We compare our model among hybrid models, including Glow [15], IGEBM [4], and JEM [10]. As shown in Tab. 3 we compare standard classification accuracy, and robust accuracy under $\ell_\infty$ PGD-20 attack with $\epsilon = 8/255$. The experimental results show that JEAT can both improve the generative capability and adversarial robustness of JEM.

JEM [10] is a well-written paper, which proposes that classifiers can also be trained as generative models and such models have the robustness comparable to adversarial training. However, when we use standard method [3] for evaluation, JEM’s robustness is only 6.11% (under $\ell_\infty$ PGD-20, $\epsilon=8/255$). We follow the same standard evaluation [3] method to test our JEAT’s robustness and show it can improve from 6.11% to 30.55% (under $\ell_\infty$ PGD-20, $\epsilon=8/255$).

Table 3. The performance comparison among four hybrid models on CIFAR-10. We use $\ell_\infty$ PGD-20 attack with $\epsilon = 8/255$.

| Model  | Standard Accuracy | Robustness |
|--------|------------------|------------|
| Glow [15] | 67.6% | - |
| IGEBM [4] | 45.06% | 32.19% |
| JEM [10] | 92.90% | 6.11% |
| **JEAT** | **85.16%** | **30.55%** |

5. Related Work

5.1. Adversarial Training

Adversarial attacks can give the wrong prediction while adding negligible perturbations for humans to input data [29]. Adversarial training first proposed in [8] can effectively defend against such attacks by training on both clean data and adversarial examples. [19] formulates adversarial training as a bi-level min-max optimization problem and trains models exclusively on adversarial images rather than both clean and adversarial images. Although it effectively improves the adversarial robustness, expensive computational cost and performance degradation on clean images are the two fatal shortcomings of it. Many works try to reduce the computation cost to the natural training of a model. [25] updates the perturbation and weights at the same time, which reduces the bi-level problem to single-level optimization. [33] samples the perturbation randomly in every iteration. Many works try to mitigate the performance degradation on clean images. [25] gives a hint on how to balance the trade-off between vanilla and robust accuracy. [9] shows that additional unlabeled data can help to increase accuracy in adversarial training. Interestingly, robust optimization can be recast as a tool for enforcing priors on the features learned by deep neural networks [7]. Moreover, a robust classifier can tackle some challenging tasks in image synthesis [23].

5.2. Energy-based Model

Recently, the energy-based model has attracted significant attention. Effective estimating and sampling the partition function is the primary difficulty in training energy-based models [17,13,30]. Some research works have made a contribution to improve the training of energy-based models, such as sample the partition function through amortized generation [20,2], utilize a separate generator network for negative image sample generations [16,28] and score matching where the gradients of an energy function are trained to match the gradients of real data [27,24]. Kevin Swersky et al. propose that treat classifier as an energy-based model can enable classifiers to generate samples rivaling the quality of recent GAN approaches [10]. Kyungmin et al. present a hybrid model which is built upon adversarial training and energy based training to deal with both out-of-distribution and adversarial examples [18].

Motivated by these discoveries, we investigate the generative capability of an adversarially trained model from an energy perspective and provide a novel explanation. We have further proposed an algorithm that can improve generation capacity.

6. Conclusion

We present a novel energy perspective on the generative capability of an adversarially trained classifier and propose our JEAT methods to obtain a classifier with stronger robustness that generates images with better quality. We find that a normal classifier can also generate images by injecting random noise in the training process, and we interpret this as blind adversarial training, for that random noise may find high energy examples accidentally. In summary, adversarial examples, blind or not, aim to find high-energy examples, and the classifier’s optimization aims to lower the energy by optimizing model parameters. This process, as we validate, is beneficial for the robustness of the model and the quality of generated images. In addition, larger model capacity [9,19], smoother activation function [9,34] and unlabeled data [9,11] are shown to improve adversarial robustness as well. We think these methods are promising ways to boost the performance of JEAT further and leave them for future work.
References

[1] Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C Duch, and Percy S Liang. Unlabeled data improves adversarial robustness. In Advances in Neural Information Processing Systems, pages 11192–11203, 2019.

[2] Bo Dai, Zhen Liu, Hanjun Dai, Niao He, Arthur Gretton, Le Song, and Dale Schuurmans. Exponential family estimation via adversarial dynamics embedding, 2020.

[3] Y. Dong, Q. Fu, X. Yang, T. Pang, H. Su, Z. Xiao, and J. Zhu. Benchmarking adversarial robustness on image classification. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 318–328, Los Alamitos, CA, USA, jun 2020. IEEE Computer Society.

[4] Yilun Du and Igor Mordatch. Implicit generation and generalization in energy-based models. CoRR, abs/1903.08689, 2019.

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

[6] Yilun Du and Igor Mordatch. Implicit generation and generalization in energy-based models, 2020.

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

[8] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. ICLR, 2015.

[9] Sven Gowal, Chongli Qin, Jonathan Uesato, Timothy Mann, and Pushmeet Kohli. Uncovering the limits of adversarial training against norm-bounded adversarial examples. arXiv preprint arXiv:2010.03593, 2020.

[10] Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one, 2020.

[11] Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, and Richard Zemel. Learning the stein discrepancy for training and evaluating energy-based models without sampling, 2020.

[12] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks, 2017.

[13] Matthew Hoffman, Pavel Sountsov, Joshua V. Dillon, Ian Langmore, Dustin Tran, and Srinivas Vasudevan. Neutralizing bad geometry in hamiltonian monte carlo using neural transport, 2019.

[14] Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. Journal of Machine Learning Research, 6(24):695–709, 2005.

[15] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In Advances in neural information processing systems, pages 10215–10224, 2018.

[16] Rithesh Kumar, Sherijl Ozair, Anirudh Goyal, Aaron Courville, and Yoshua Bengio. Maximum entropy generators for energy-based models, 2019.

[17] Yann LeCun, Sumit Chopra, Raia Hadsell, Marc’Aurelio Ranzato, and Fu Jie Huang. A tutorial on energy-based learning, 2006.

[18] Kyungmin Lee, Hunmin Yang, and Se-Yoon Oh. Adversarial training on joint energy based model for robust classification and out-of-distribution detection. In 2020 20th International Conference on Control, Automation and Systems (ICCAS), pages 17–21, 2020.

[19] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083, 2017.

[20] Erik Nijkamp, Mitch Hill, Song-Chun Zhu, and Ying Nian Wu. Learning non-convergent non-persistent short-run mcmc toward energy-based model, 2019.

[21] Shibani Santurkar, Andrew Ilyas, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Image synthesis with a single (robust) classifier. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 32, pages 1262–1273. Curran Associates, Inc., 2019.

[22] Shibani Santurkar, Dimitris Tsipras, Brandon Tran, Andrew Ilyas, Logan Engstrom, and Aleksander Madry. Image synthesis with a single (robust) classifier, 2019.

[23] Shibani Santurkar, Dimitris Tsipras, Brandon Tran, Andrew Ilyas, Logan Engstrom, and Aleksander Madry. Image synthesis with a single (robust) classifier, 2019.

[24] Saeed Saremi, Arash Mehrjou, Bernhard Schölkopf, and Aapo Hyvärinen. Deep energy estimator networks, 2018.

[25] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In Advances in Neural Information Processing Systems, pages 3353–3364, 2019.

[26] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 32, pages 11918–11930. Curran Associates, Inc., 2019.

[27] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution, 2020.

[28] Yunfu Song and Zhijian Ou. Generative modeling by inclusive neural random fields with applications in image generation and anomaly detection, 2020.

[29] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199, 2013.

[30] Yulin Wang, Xuran Pan, Shiji Song, Hong Zhang, Gao Huang, and Cheng Wu. Implicit semantic data augmentation for deep networks. In Advances in Neural Information Processing Systems, pages 12614–12623, 2019.

[31] Zhou Wang, Member, Alan Conrad Bovik, Hamid Rahim Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity, 2004.
[32] Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient langevin dynamics. In Proceedings of the 28th International Conference on International Conference on Machine Learning, ICML’11, page 681–688, Madison, WI, USA, 2011. Omnipress.

[33] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. arXiv preprint arXiv:2001.03994, 2020.

[34] Cihang Xie, Mingxing Tan, Boqing Gong, Alan Yuille, and Quoc V Le. Smooth adversarial training. arXiv preprint arXiv:2006.14536, 2020.

[35] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy, 2019.
A. Generated Images on CIFAR-100

We show the images generated by JEAT on CIFAR-100 in Fig. 8. These images are rich in details and vivid.

Figure 8. Images generated by JEAT on CIFAR-100.

B. The Change of Energy

B.1. The Change of Energy $E_\theta(x, y)$ in Training

We illustrate the change of energy $E_\theta(x, y)$ in adversarial training on CIFAR-100 in Fig. 9. Adversarial attacks generate high-energy adversarial examples, and then the energy of adversarial examples is decreased by updating model parameters. As we show in our paper, adversarial training flattens the energy region around real data in this way.

B.2. The Change of Energy $E_\theta(x)$ in Training

We illustrate the change of energy $E_\theta(x)$ in adversarial training on CIFAR-100 in Fig. 10. The value of $\Delta_\theta E_\theta(x)$ fluctuates around zero, sometimes positive and sometimes negative. Thus $E_\theta(x)$ has not been well optimized in the classification task. Using $E_\theta(x)$ in the generation task by a robust classifier may be harmful to the images’ quality.

B.3. The Change of Energy in Generation

Image generation by an original robust classifier is introduced in the main paper, and the generating procedure of images can be formulated as:

$$x' = x - \frac{\eta}{2} \cdot \nabla_x (E_\theta(x, y) - E_\theta(x)) + \sqrt{\eta} \epsilon. \quad (25)$$

As analyzed in Sec. 3.2, $E_\theta(x, y)$ is the key term for conditional generation by using robust classifier. We illustrate $\nabla_x E_\theta(x, y)$ and $\nabla_x E_\theta(x)$ in generation process in Fig. 11(CIFAR-10) and Fig. 12(CIFAR-100). In the beginning, $\nabla_x E_\theta(x, y)$ dominates for that the absolute value of $\nabla_x E_\theta(x)$ is less than that of $\nabla_x E_\theta(x, y)$. As the number of iterations increases, the influence of $\nabla_x E_\theta(x)$ and $\nabla_x E_\theta(x, y)$ gradually become much more closer. Thus, $E_\theta(x, y)$ plays a key role for conditional generation by using robust classifier.

C. The Gradient on $\theta$ of $\log p_\theta(x)$

$p_\theta(x)$ in JEAT algorithm can be expressed as:

$$p_\theta(x) = \frac{\exp(-E_\theta(x))}{Z_\theta}, \quad (26)$$

The gradient on $\theta$ of $\log p_\theta(x)$ is defined as

$$\nabla_\theta \log p_\theta(x) = -\nabla_\theta E_\theta(x). \quad (27)$$
Table 4. Hyperparameters of different methods.

| HYPERPARAMETER | LEARNING RATE | BATCH SIZE | SGD SCHEDULER | EPOCHS | WEIGHT DECAY | MOMENTUM |
|----------------|---------------|------------|----------------|--------|--------------|----------|
| JEAT (ours)    | 1e-4          | 64         | ADAM [160, 180] | 200    | 0            | BETA=[0.9, 0.999] |
| JEM            | 1e-4          | 64         | ADAM [160, 180] | 200    | 0            | BETA=[0.9, 0.999] |
| Free           | 0.1           | 128        | MultiStep [100, 150] | 200    | 2e-4         | 0.9      |
| Fast           | 0-0.2         | 128        | Cyclic         | 15     | 5e-4         | 0.9      |
| TRADES         | 0.1           | 128        | MultiStep [75, 90, 100] | 120    | 2e-4         | 0.9      |

**D. Proof for** \( Z_\theta = \tilde{Z}_\theta \)

As defined in the main paper, the energy functions for classifier are:

\[
\begin{align*}
E_0(x, y) &= -\log(\exp(f(x; \theta)[y])), \\
E_0(x) &= -\log(\sum_k \exp(f(x; \theta)[k])).
\end{align*}
\]

(28)

Also \( p_0(x, y) = \frac{\exp(-E_0(x, y))}{Z_\theta} \), \( \tilde{Z}_\theta \) is the normalizing constant which is defined as:

\[
\tilde{Z}_\theta = \int_x \sum_y \exp(-E_0(x, y))dx
\]

(29)

As \( p_0(x) = \frac{\exp(-E_0(x))}{Z_\theta} \), \( Z_\theta \) is the normalizing constant which is defined as:

\[
Z_\theta = \int_x \exp(-E_0(x))dx
= \int_x \exp(\log(\sum_y \exp(\sum_{y} \exp(f(x; \theta)[y]))))dx
= \int_x (\sum_y \exp(f(x; \theta)[y]))dx
\]

(30)

By Eq. (29) and Eq. (30), it can be seen that \( Z_\theta = \tilde{Z}_\theta \).

**E. Algorithm for PreJEAT**

In the main paper, we show the training algorithm called Joint Energy Adversarial Training (JEAT), in which we replace cross-entropy loss \(-\log p_0(y|x)\) with \(-\log p_0(y|x) - \log p_0(x)\) and use adversarial examples found by

\[
x_{adv} = x + \epsilon \cdot \text{sign}(\nabla_x E_0(x, y)).
\]

(31)

We also propose Preliminary Joint Energy Adversarial Training (PreJEAT) which just trains model with adversarial examples found by Eq. (18) and use cross-entropy loss. We present the algorithm here in Algorithm 2.

**F. Training Details**

The hyper-parameters for our experiments are shown in Tab. 4. We run JEM [10]. Free adversarial training (Free m=8) [25], Fast adversarial training (Fast) [33] and TRADES (1/λ=6) [35] with their open-source code. Multi-Step scheduler decay by 10 every time.
Algorithm 2 Training and Generating of PreJEAT: Given network \( f \), \( E(x, y) = -\log(\exp(f(x)|y)) \) represent energy of \( (x, y) \), \( E(x) = -\log(\sum_y \exp(f(x)|y)) \) represent energy of \( x \), \( \ell_\infty \) adversarial perturbation radius \( \epsilon \), SGLD step-size \( \alpha \), SGLD steps \( K \), epochs \( T \), dataset of size \( M \), learning rate \( \eta \).

Training:
for \( t = 1, 2, \ldots, T \) do
   for \( j = 1, 2, \ldots, M \) do
      \( \triangleright \) Generating energy-based adversarial samples:
      \( \delta = U(-\epsilon, \epsilon) \)
      \( \delta = \delta + \epsilon \cdot \text{sign}(\nabla_x (E_\theta(x, y))) \)
      \( \delta = \max(\min(\delta, \epsilon), -\epsilon) \)
      \( x_{adv} = x_j + \delta \)
      \( \triangleright \) Updating model parameters:
      \( \nabla_\theta L_p(y|x_{adv}) = \nabla_\theta (E_\theta(x_{adv}, y) - E_\theta(x_{adv})) \)
      \( \theta = \theta - \eta \cdot \nabla_\theta L_p(y|x_{adv}) \)
   end for
end for

Generating:
\( x_0 \sim \text{random sample} \)
for \( t = 0, 1, 2, \ldots, K - 1 \) do
   \( x_{t+1} = x_t - \frac{\alpha}{2} \cdot \nabla_x E_\theta(x_t, y) + \sqrt{\alpha} \cdot N(0, I) \)
end for
Output: \( x_{gen} = x_K \)

G. Interesting Benefits of JEAT

G.1. Denoise

Due to the influence of factors such as the environment and transmission channels, the image is inevitably contaminated by noise in the process of acquisition, compression and transmission. In the presence of noise, subsequent image processing tasks (such as video processing, image analysis, and tracking) may be negatively affected. Therefore, image denoising plays an important role in modern image processing systems. In this section, we found that the classifier trained by JEAT has the effect of denoising to some extent.

We further show the energy contours of the normal classifier, robust classifier and JEAT classifier around a given image from test dataset (CIFAR-10) in Fig. 13. The points with the lowest energy of both the normal classifier and the robust classifier deviate from the point where the real image is located. The generation process will result in entering the low energy positions along the direction of energy descent from the real image. Therefore, even if it starts from the real image, the image generated by the normal classifier will be inferior. Because the low-energy region of the original robust classifier deviates slightly from the real data, the quality of generated images will be slightly better than that of the normal classifier. Nevertheless, in the energy of JEAT classifier, the real image has the minimum energy, and the energy function is flat and smooth around the real data. Starting from the real image, the image generated by the JEAT classifier is almost identical to the real image.

This inspires us that if the energy of the noise image is high, it is possible to generate an clean image along the direction of energy descent, thereby denoising. We show the different energy contours in the process of denoising by JEAT classifier in Fig. 14. We get noisy images by injecting Gaussian noise with a mean value of zero and a variance of 0.3 into the original image. Then we generate low-energy images from these noisy images using Langevin Dynamics. As shown in Fig. 15, JEAT classifier has good denoising effect, while normal classifier and original robust classifier have poor denoising effect.

G.2. Calibration

The outputs of a classifier are often interpreted as the predictive confidence that this class was identified. However Guo et al. [12] claim that deep neural networks are often not calibrated which means that the confidence always does...
Figure 15. We show that compared to the normal classifier and original robust classifier, JEAT classifier can indeed achieve the effect of denoising. (a) Original images. (b) Noise images obtained by superimposing Gaussian noise with a mean value of 0 and a variance of 0.3 on the original image. (c) The image obtained after denoising by the normal classifier. (d) The image obtained after denoising by the robust classifier. (e) The image obtained after denoising by the JEAT classifier.

Figure 16. We draw the reliability diagram of different classifiers on original test data of CIFAR-10 and adversarial data. We use FGSM attack ($\epsilon = 8/255$) to get adversarial data. (a) The reliability diagram of normal classifier on original clean data. (b) The reliability diagram of normal classifier on adversarial data. (c) The reliability diagram of robust classifier on original clean data. (d) The reliability diagram of robust classifier on adversarial data. (e) The reliability diagram of JEAT classifier on original clean data. (f) The reliability diagram of JEAT classifier on adversarial data.

not align with misclassification rate. Expected Calibration Error (ECE) is a metric to measure the calibration of a classifier. For a perfectly calibrated classifier, ECE value will be zero.

We using the reliability diagram to find out how well the classifier is calibrated in Fig. 16. The model’s predictions are divided into bins based on the confidence value of the target class, here, we choose 20 bins. The confidence histogram at the bottom shows how many test examples are in each bin. Two vertical lines represent accuracy and average confidence, and the closer these two lines are, the better the model calibration is. The reliability histogram at the top shows the average confidence of each bar and the accuracy
of the examples in the bar. For each bin we plot the difference between the accuracy and the confidence using the red bars in the diagram.

It can be clearly seen from the histogram that the confidence of most predictions of these classifiers is greater than 0.8. The normal classifier is always over-confidence and gives more false positives. This phenomenon becomes even worse when facing adversarial data and the ECE value is 49.63%. Robust classifier will not give much over-confident prediction when encountering adversarial perturbation. However, when faced with clean data, the robust classifier is under-confidence and gives more false negatives. The good news is that the JEAT classifier has a good calibration when classifying clean data or adversarial data. When a model is deployed in a real-world scenarios, good calibration is an important feature. And the confidence of a model with good calibration can be used to judge whether to output the result or recognize it again. The JEAT classifier which is better calibrated than both normal classifier and original robust classifier can be more useful in real-world scenarios.