SAPIENZA
UNIVERSITÀ DI ROMA

On the Sensitivity and Uncertainty of Convolution Neural Networks to Adversarial Perturbations

Facoltà di Ingegneria dell'Informazione, Informatica e Statistica
Laurea Magistrale in Computer Science

Senad Beadini
ID number 1754617

Advisor
Prof. Iacopo Masi

Co-Advisor
Prof. Gabriele Tolomei

Academic Year 2021/2022

Thesis defended in 19/10/2022

On the Sensitivity and Uncertainty of Convolution Neural Networks to Adversarial Perturbations

Master thesis. Sapienza University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Author's email: beadini.senad@gmail.com

Dedicated to my parents and all the people who, with just a few words, encouraged me to never give up.

Abstract

Convolutional neural networks (CNNs) are the de facto standard for facing highly complex problems in various areas of science, engineering, and physics. Specifically, extraordinary leaps were made in cognitive tasks like computer vision [19] [17], biometrics [54], and medical imaging [30] [50]. The capacity to generalize and capture nonlinear patterns from data makes CNNs extremely effective for predicting phenomena, analyzing large amounts of data, and making decisions. Despite their amazing performance on the aforementioned tasks, it is well known that neural networks suffer from adversarial examples. These data contain small perturbations, imperceptible to the human eye, that easily fool even state-of-the-art models. In addition, it has been verified that there exist adversarial examples based on geometric transformations (such as rotation, translation, color, etc.) that cause a drop in accuracy even to models trained with those same transformations.

To investigate this phenomenon, we examine the robustness of a well-known class of CNNs with residual connections in the parameter space of common transformations. Considering one of the strongest adversarial attacks known in the literature to fool CNNs, namely Projected Gradient Descent (PGD) [40], we analyze the trade-off between robustness to PGD perturbations and five common perturbations: rotation, perspective, Gaussian noise, Rademacher noise, motion blur.

The results show that is not always possible to make a model robust via data augmentation to both PGD perturbations and common ones. In addition, our experiments reveal that data augmentation and adversarial training are effective ways to make neural networks well-calibrated in the presence of common perturbations.

To shed some light on the sensitivity of CNNs to adversarial perturbations, we hypothesize that adversarial examples are out-of-distribution (OOD) data. Thus, we reinterpret adversarial examples in the context of energy-based models.

We found an empirical validation of the above hypothesis by showing that 1) adversarial examples have high energy and 2) the strength of the attack is correlated with energy.

Leveraging these observations, we reinterpret a CNN classifier as an energy-based model and show that it is possible to use energy as a scoring function for any neural network to detect adversarial examples at inference time. The energy-based detection approach, 1) does not need training 2) does not modify the neural network on which it is applied 3) detects adversarial examples with only 1 degree of freedom, and 4) requires only a threshold estimation.

The detector built on top of the energy function achieves a 98% detection rate and a 1% false positive rate for adversarial perturbations created using PGD on the high dimensional dataset Imagenette.

In contrast, we furthermore show that is possible to "break" our detector by crafting a low-energy PGD attack (LE-PGD). LE-PGD produces adversarial data that still fool completely the network yet have energy similar to natural data, making the energy-based detection algorithm valueless. The existence of low-energy adversarial data introduces questions about the relationship between these types of perturbations, generative models, and energy-based models.

Contents

Chapter 1

Introduction

Men, since ancient times, have dreamed of creating machines that can think and learn. Artificial intelligence (AI), the engineering discipline that develops algorithms to replicate human capabilities and the capabilities of intelligent beings in general, has made exponential progress in recent years. Much of this progress, however, is due to an approach to AI called machine learning (ML).
Tom Mitchell in [41] defines that "a computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E".
Talking in simpler terms, the experience E can be considered the set of data that our program is going to interact with. It is clear that the performance of an ML program depends heavily on the representation of the data (experience E) they are given.
Many AI tasks, can be solved by designing suitable features to extract from the data and providing them to an ML algorithm. On the other hand, however, for many difficult tasks, we do not know a priori what data features are needed to efficiently solve that task and what other features can be ignored; sometimes even the task definition itself is difficult to formalize. A solution to this problem is to use machine learning to discover the data representation itself.
Deep learning, a sub-field of machine learning —Fig. 1.1, solves the main problem in representation learning by introducing data representations that are expressed in terms of other simpler representations [20].

1.1 Deep Learning

The boost that machine learning has had in the past decade is due to several reasons. First, deep neural networks (DNNs) have been supported by massive labeled data [12] and increased computing power due to hardware accelerators. Second, research contributions in neural network theory have enabled the development of efficient computational techniques [53], [32], [34]. The most concrete achievement, which in some ways gave birth to deep learning, was the convolution neural network (CNN) [36], [38]. Even more surprising are the applications of DNNs and their performances in fields such as computer vision [19] [17] [34], biometrics [54], medical imaging [30] [50], speech recognition [42], applied physics [29] [8], and computational biology [43].

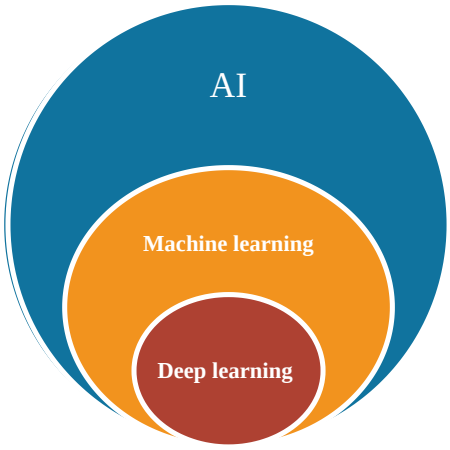


Figure 1.1. A Venn diagram showing the relations between AI, machine learning, and deep learning.

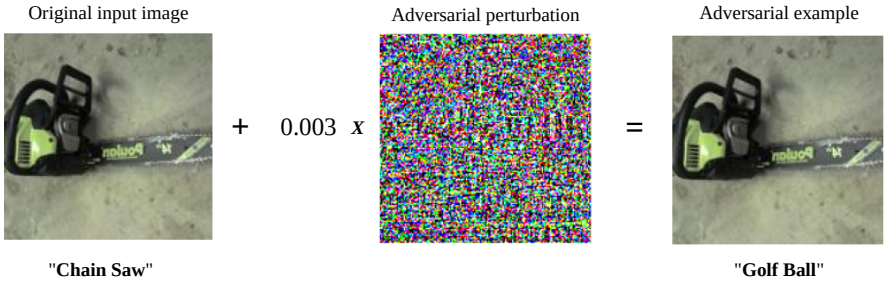


Figure 1.2. An adversarial example. A chain saw is made to be a golf ball.

1.2 Adversarial Deep Learning

Despite their great achievements, DNNs suffer from an intriguing phenomenon. They are vulnerable to adversarial examples [55]. These are data that DNNs misclassify, though they are similar to data that the model correctly classifies. In particular, these types of data contain perturbations, undetectable to human perception, that destroy the performance of even a state-of-the-art model—Fig. 1.2. This topic opens a first security and reliability problem for DNNs, in the sense that it makes all previous advances in the diverse disciplines futile. For example, the signal recognition system of a self-driving car could be fooled by so-called attacks in the physical world [14], [16]. So a stop sign can be turned into a 50 mph limit signal using special graffiti —Fig. 1.3.
So in the same way, a biometric face recognition system becomes useless if you wear glasses with a special texture [49] —Fig. 1.4.
Yet, even a speech recognition system is vulnerable to adversarial perturbations [7]—Fig. 1.5. Therefore, all these examples make it clear that the study of adversarial perturbations and vulnerabilities of DNNs is a crucial topic.

Figure 1.3. On the left, a stop sign with real graffiti. On the right, a stop sign with adversarial graffiti [16].

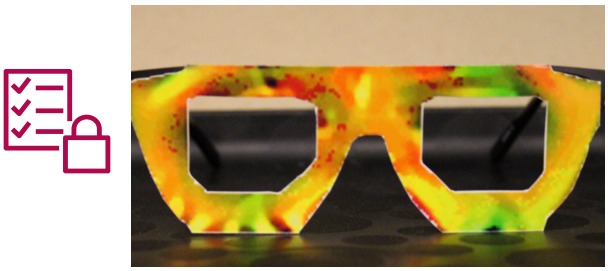


Figure 1.4. Adversarial glasses. Anyone wearing these glasses can change identities and fool a face recognition system [49].

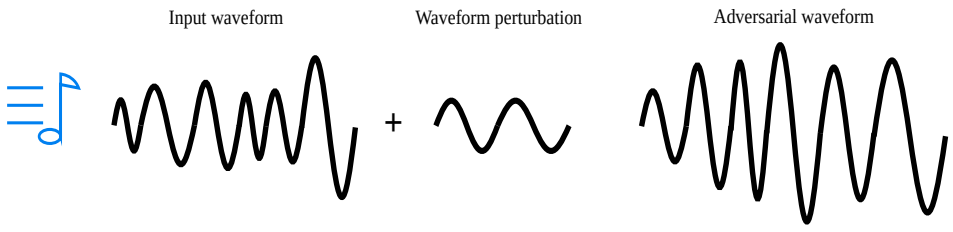


Figure 1.5. A schematic view of an audio adversarial attack [7].

1.3 Contribution

In this master's thesis, we address the topic of the robustness of CNNs. First, we investigate the robustness of the ResNet10 model, a classifier belonging to a well-known architecture with residual connections, in the parameter space of common transformations. Taking into consideration one of the most widely used attacks in literature, namely PGD, we analyzed and looked for relationships between robustness to PGD perturbations and robustness to five common perturbations (rotation, perspective, Gaussian noise, Rademacher noise, motion blur). For each perturbation we produce two kinds of graphs showing the relationships between the two robustness; the finding is that is not always possible to make a model robust via data augmentation to both PGD perturbations and common ones.
In the same direction, we analyze the uncertainty of the model under perturbed data, confirming the observations of previous work that adversarial training yields a more calibrated model. Our experiments show that data augmentation is also a way to make a model more calibrated in the presence of common perturbations.
By better investigating the results obtained, and under the assumption that adversarial examples are out-of-distribution data (OOD), we reinterpret adversarial perturbations in the context of energy-based models (EBMs)—[39],[60].
Under the empirical observation that adversarial data have high energy, we show that one can use the energy (or Lp norms) as a scoring function for any DNN to detect adversarial examples at inference time. Our analysis describes an interesting relationship between the energy value (or L_2 norm of logits) and the aggressiveness of the PGD attack. In particular, the larger the number of steps of the PGD attack the higher the energy of those adversarial data. The energy-based adversarial attack detection algorithm gets excellent results on the high dimensional dataset Imagenette [26]. Specifically, it achieves a 98% detection rate and a 1% false positive rate for adversarial perturbations created using PGD.
Changing perspective, we put ourselves in the shoes of an attacker and try to break our detection algorithm. To do that, we suggest a new variant of PGD, which we call low-energy PGD (LE-PGD), which uses a new loss to be optimized. LE-PGD yields adversarial examples that have a similar energy to natural data, making the energy-based detection algorithm useless.

Chapter 2

Related Work

Due to the fact that deep learning models are vulnerable to adversarial examples [55] a huge amount of research has been done to investigate adversarial robustness. Given the vastness of the work already done, the topics will be outlined by themes.

2.1 L_p-norm Adversarial Examples

Since the first moment, research on DNNs robustness has been based on L_p-norm adversarial attacks [21], [35], [40], [6]. These types of adversarial examples are generated by adding to an image $\boldsymbol{x}$ a perturbation $\boldsymbol{\delta}$ (in the pixel space) such that $\|\boldsymbol{\delta}\|_p$ (and thus the similarity of the two images) is limited by a parameter ϵ.
Goodfellow et al. [21] proposed the fast gradient sign method (FGSM), which generates adversarial examples with a single gradient step.
Madry et al. [40] demonstrated that Projected Gradient Descent (PGD), a multistep generalization of FGSM, can produce strong adversarial images; moreover, they leveraged PGD to significantly increase DNNs robustness to L_2 and L_∞ attacks. Carlini and Wagner [6] proposed an optimization-based algorithm capable of producing strong adversarial examples using the norm L_0, L_2 and L_∞.

2.2 Transformation-based Adversarial Examples

While the L_p-norm attacks provoke modifications to the raw pixel values, other methods generate adversarial examples using a different approach. Athalye et al. [2] introduced a technique to develop adversarial examples via a chosen distribution of transformations. Azulay et al. [3] have already analyzed why neural networks are very sensitive to small geometric transformations and the lack of invariance in modern CNNs.
Kanback et al. [28] investigated DNNs' robustness under geometric transformation by making use of a new algorithm (i.e. ManiFool); the idea of the ManiFool algorithm is to iteratively move from an image sample toward the decision boundary of the classifier while staying on the transformation manifold. Furthermore, they showed that adversarial training with ManiFool increases DNN robustness against random and worst-case transformations [28].
Engstrom et al. [15] explored rotation/translation (spatial) robustness on a large

number of datasets and training strategies; in their analysis, they observed that a first-order method, although very effective for pixel-wise attack, is not useful for generating adversarial examples w.r.t. parameters of spatial transformations. Instead, they showed that grid search and worst-of-k approaches are effective in producing good adversarial examples.
Sehwag et al. [47] extended the work of [15] by focusing not only on spatial transformations but also on other transformations (blurring, coloring, weather, etc.); they also confirm once again the ineffectiveness of using the first-order method in producing adversarial data with respect to transformation parameters. However, they focus their attention on the worst-of-k with k=1 justifying a not big difference between k=10 and k=1.

2.3 Defenses and Adversarial Training

The best technique to obtain robust models is adversarial training [40]. This approach has been improved with many variants like in [58], [46].
Data augmentation can significantly improve neural networks' generalizability; recently this technique has been exploited to increase adversarial robustness as well. [44].
Kamath et al. [27] have recently discovered the trade-off between spatial and raw pixel-wise robustness in neural networks. In particular, they proved that the spatial robustness of both equivariant models [11] and standard CNNs improves by training augmentation with progressively larger spatial transformations, at the same time, their adversarial robustness worsens progressively; models are adversarially trained with progressively larger raw pixel-wise perturbations, their spatial robustness drops progressively. They provide a curriculum-based adversarial training approach to balance the trade-offs.

Chapter 3

Background

In this chapter, we provide a formal definition of adversarial training and set up our framework. We will describe raw pixel-wise adversarial examples and then generalize the mathematical setting to more advanced attacks.
Consider a set of labeled images $X = \{(\boldsymbol{x_i}, y_i) \mid \boldsymbol{x_i} \in \mathcal{M} \subseteq \mathbb{R}^d \text{ and } y_i \in \{1, .., K\}\}$, assuming that each $(\boldsymbol{x_i}, y)$ is generated from an underlying distribution D; we can define a classifier $F_{\boldsymbol{\theta}}$ parameterized by $\boldsymbol{\theta}$ and assume that $F_{\boldsymbol{\theta}}$ is differentiable w.r.t. $\boldsymbol{\theta}$. The classifier $F_{\boldsymbol{\theta}}$ is said accurate if its predictions match labels y.

3.1 $\mathbf{L}_p$ Adversarial Learning

We define the problem of learning a classifier $F_{\boldsymbol{\theta}}$ parameterized by $\boldsymbol{\theta}$ from the dataset X as :

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{x},y)\sim D} \left[\mathcal{L}\Big(F_{\boldsymbol{\theta}}(\boldsymbol{x}),\ y \Big) \right] \tag{3.1}$$

where $\mathcal{L}$ is a loss function (i.e. cross-entropy). Given $(\boldsymbol{x}, y) \in X$ we say $\boldsymbol{x}^*$ is an adversarial example if $F_{\boldsymbol{\theta}}(\boldsymbol{x}^*) \neq y$ and $\boldsymbol{x}^*$ is "visually" similar $\boldsymbol{x}$. Therefore, a robust model is one that is capable of recognizing two visually similar images $\boldsymbol{x}$ and $\boldsymbol{x}^*$ returning the same label for both.
Eq. 3.1 does not provide robust models to adversarial examples [21], [55]; To address the adversarial robustness problem, Madry et al. in [40] changed Eq. 3.1 producing a new learning problem taking into account robustness.

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{x},y)\sim D} \left[\max_{\boldsymbol{\delta}\in S} \mathcal{L}\Big(F_{\boldsymbol{\theta}}(\boldsymbol{x}+\boldsymbol{\delta}),\ y \Big) \right] \tag{3.2}$$

in which $S = \{\boldsymbol{\delta} : ||\boldsymbol{\delta}||_p \leq \epsilon\}$. Thus, in other words, $\boldsymbol{\delta}$ is a raw pixel perturbation such that $\boldsymbol{x}^* = \boldsymbol{x} + \boldsymbol{\delta}$ with property $||\boldsymbol{x}^* - \boldsymbol{x}||_p \leq \epsilon$. Note that the similarity of $\boldsymbol{x}^*$ (the adversarial) and $\boldsymbol{x}$ (the original image) depends on a L_p norm.
Informally, Eq. 3.2 is saying that we want a classifier $F_{\boldsymbol{\theta}}$ such that it performs well on the "clean" data and, at the same time, the model should be robust (the inner maximization part) to $\boldsymbol{\delta}$ perturbations. Eq. 3.2 is a double optimization problem. These kinds of problems are difficult to solve and often intractable. Although this characteristic is present, Madry et al. [40] showed that, after all, the inner

maximization part can be approximated and the whole learning problem can be tractable.
The following algorithms approximate the maximization part of Eq. 3.2 and produce adversarial examples to address the problem 3.2.

3.1.1 Fast Gradient Sign Method

Fast gradient sign method (FGSM) [21] is an algorithm for L_∞-norm adversarial attacks and it is described in the following formula below

$$\boldsymbol{x}^* = \boldsymbol{x} + \epsilon \ \text{sign}\Big(\nabla_{\boldsymbol{x}}\{\mathcal{L}(F_{\boldsymbol{\theta}}(\boldsymbol{x}),\ y)\} \Big) \tag{3.3}$$

where sign is a function that returns the sign of its argument. Note that the gradient is computed w.r.t. $\boldsymbol{x}$.

3.1.2 Projected Gradient Descent

Projected gradient descent (PGD) can be seen as an iterative generalization of the FGSM algorithm [40]. The iterative procedure is shown below :

$$\boldsymbol{x}^* = \text{clip}_\epsilon \ \Big(\boldsymbol{x}^* + \alpha \ \text{sign}\Big(\nabla_{\boldsymbol{x}^*}\mathcal{L}(F_\theta(\boldsymbol{x}^*),\ y)\} \Big)\Big) \tag{3.4}$$

where clip denotes the function that projects its argument to the surface of $\boldsymbol{x}$ s neighbor ϵ-ball while α is the step size. This algorithm is able to produce strong adversarial data that have a high probability to fool a model; for this reason, it usually is used as a benchmark for the robustness of neural networks.

3.1.3 Carlini-Wagner

Carlini and Wagner [6] proposed an optimization-based algorithm that can produce a strong adversarial example. C&W attack minimizes the L_p distance between the adversarial image and the original image until the image label changes. The general formulation of C&W can be described as follows:

$$\min ||\boldsymbol{\delta}||_p + c\ f(\boldsymbol{x} + \boldsymbol{\delta}) \ \ s.t. \ \ \boldsymbol{x} + \boldsymbol{\delta} \in [0,1]^d \tag{3.5}$$

where $f(\boldsymbol{x} + \boldsymbol{\delta})$ is a well-designed function [6] and c is a scalar.

Summarizing, we can learn our $F_{\boldsymbol{\theta}}$ classifier by training on the adversarial examples produced by the methods described above; this technique takes the name of adversarial training. Adversarial training has been improved many times, nowadays there are numerous techniques and variations [48], [58], [48]. Other types of adversarial example generation algorithms are not based on L_p norm directly in the pixel space [25], [15], [59], [5], [47]. These are more natural adversarial attacks that do not exploit just raw pixel perturbations but instead take into account the whole image representation (for instance, applying spatial transformations).

3.2 Transformation-based Adversarial Learning

Let us consider a set of possible transformation S; we call a transformation operation $T \in S$ *augmentation* defined as $T_{\boldsymbol{\gamma}}(\boldsymbol{x}) : \mathcal{M} \to \mathcal{M}$ where $\boldsymbol{\gamma}$ are the parameters of the transformation. We assume that the label of $\boldsymbol{x}$ does not change after the application of $T_{\boldsymbol{\gamma}}$. Adversarial robustness can be generalized w.r.t. T as follows [47] :

$$\min_{\boldsymbol{\gamma}} \mathbb{E}_{(\boldsymbol{x},y)\sim D} \left[\max_{\boldsymbol{\gamma}} \mathcal{L}\Big(F_{\boldsymbol{\theta}}(\, T_{\boldsymbol{\gamma}}(\boldsymbol{x})), y \,\Big) \right] \tag{3.6}$$

Sometimes it could be useful to add the constraint $||\boldsymbol{\gamma}||_p \leq ||\boldsymbol{\gamma}^*||_p$ to limit the possible number of transformations (like limit the degree of rotation to less than 45-degree).
We have to note that in practice we hardly know D, we only have a set of data points (X in our case). Thus, the problem in Eq. 3.6 (but the same applies to Eq. 3.1, 3.2) becomes :

$$\min_{\boldsymbol{\theta}} \ \frac{1}{|X|} \sum_{(\boldsymbol{x},y)\in X} \left[\max_{||\boldsymbol{\gamma}||_p \leq ||\boldsymbol{\gamma}^*||_p} \mathcal{L}\Big(F_{\boldsymbol{\theta}}(\, T_{\boldsymbol{\gamma}}(\boldsymbol{x})\,), y \Big) \right] \tag{3.7}$$

where $\mathcal{L}$ is, as previously, the cross-entropy function.
Notice that what we have just described can be related to a more general property of deep learning models: invariances. In fact, a function F is said to be invariant to a transformation $T_{\boldsymbol{\gamma}}$ if $F(T_{\boldsymbol{\gamma}}(\boldsymbol{x})) = F(\boldsymbol{x})$. In this context, Eq. 3.7 can be seen as a learning invariance problem. In other words, we are looking for a model capable of being invariant to challenging (adversarial) transformations. It is important to note that although in 3.7 appears the L_p norm, the latter is applied to the parameters of the transformations, while in L_p adversarial robustness the norm is used as a metric for image similarity. From now on we will call the two methods without any sort of ambiguity adversarial transformation robustness and (standard) adversarial robustness. The question now arises as to how adversarial training can be applied to Eq. 3.7.

First order method. It might come to mind to use the approach as in the previous scenario subject to perturbations based on the raw pixels and L_p bounded norm, i.e., to use loss gradients with respect to the parameters $\boldsymbol{\gamma}$ of T transformations. That is, making optimization on the parameters of the transformations instead of the raw pixels as in the case of L_p-based attacks to find the most adversarial transformation T. However, Engstrom et al. in [15] showed that this approach, for spatial transformations, is not capable of finding maxima because of the high non-convexity of the loss with respect to the parameters of the spatial transformations. Sehwag et al. in [47] saw that this fact happens for other transformations and corruptions as well.

Worst-of-k. Engstrom et al. in [15] showed how an effective and simple approach for problem 3.7 is grid search. That is, discretize the $\boldsymbol{\gamma}$ parameters of a transformation $T_{\boldsymbol{\gamma}}$ and choose the ones which make the model wrong. It is important to note that the grid-search approach in this scenario is possible since the dimensionality of the $\boldsymbol{\gamma}$ parameters is low. Furthermore, Engstrom et al. in [15] analyzed the worst-of-k approach, which consists of choosing k types of attack parameters and choosing

those that are worst for the model $F_{\boldsymbol{\theta}}$. In the same direction, Sehwag et al in [47] showed that it is reasonable to consider the worst-of-k with k=1, hence performing a random augmentation, to obtain a high standard and robust accuracy compared to worst-of-k with k=10.

3.2.1 Geometric Transformations

The human vision system is able to be robust to strong changes in images. Nevertheless, neural networks are the state-of-the-art model in many computer vision tasks they can be fooled easily by several transformations, corruptions, and even unnoticeable perturbations [24], [47]. In our analysis, we are going to consider 4 main operations.

Rotation. Rotation is a spatial transformation and it has already been studied in [15] and [27]. We are going to consider this transformation as a baseline for the next ones.

Perspective. The human eye when it sees, distant objects are smaller than closer ones, this phenomenon is called perspective. The camera works in the same way as the human vision system and introduces perspective effects of objects placed sideways and far away [56]. The study of perspective is also motivated by the observation in [47], in which perspective transformations are shown as highly adversarial.

Gaussian Noise. The relationship between Gaussian noise and adversarial robustness has been analyzed many times in the scientific community [18], [10]. In [18] it is shown that there is a strong connection between adversarial robustness and Gaussian noise, precisely that a non-zero error rate in Gaussian noise implies the existence of small adversarial perturbations of noisy images. Furthermore, again in [18], it has been shown that training on Gaussian noise moderately improves adversarial robustness.

Motion Blur. A common corruption of real images is motion blur. This usually happens when there is relative movement between the camera and the objects or scene being framed.

Although numerous other transformations and corruptions exist, we will focus on these four classical transformations to study their possible relationships with L_p-based adversarial robustness, expanding the analysis performed by Kamath et al. in [27]. In particular, it is interesting to ask whether trade-offs similar to those found in [27] exist between other geometric transformations rather than spatial ones.

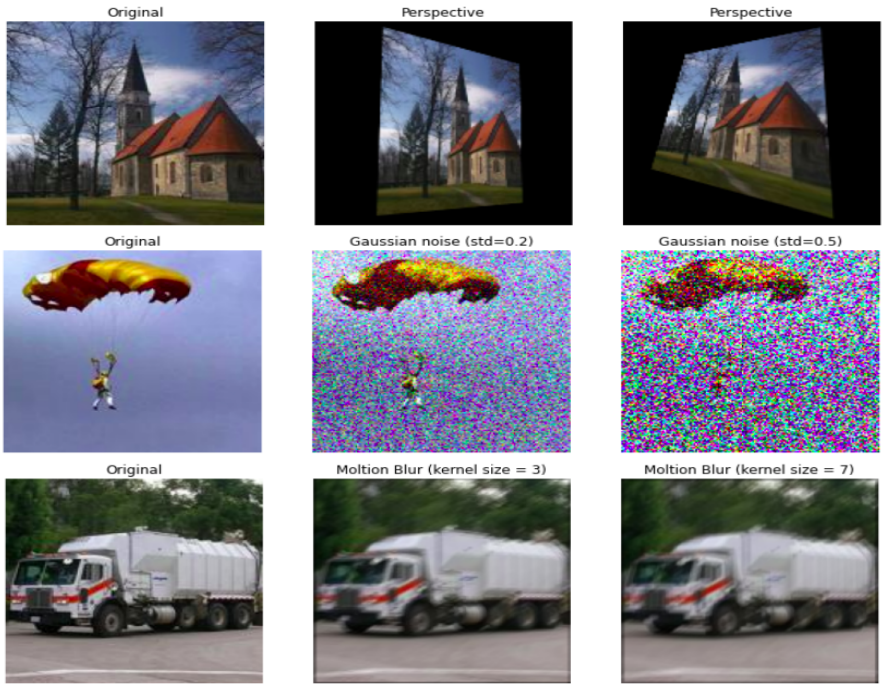


Figure 3.1. Perspective, Gaussian noise, and motion blur operations with different intensity.

Take-home message

Eq. 3.7 describes the problem of robustness of adversarial transformations. Addressing the inner maximization part of the problem 3.7 :

- Engstrom et al. in [15] note that first-order methods (such as FGSM) are not effective at producing strong spatial adversarial transformations.
- Effective methods for spatial transformations are random, grid search and worst-of-k [15].
- Sehwag et al. in [47] observed that first-order methods are also not effective for other transformations such as noise and geometric transformations.
- The worst-of-k method with k=1, i.e. random augmentation, seems to provide similar performance to methods with larger k. So a small loss of invariance is accepted to have lower computational complexity [47].

Kamath et al. in [27] find that there is a trade-off for CNNs between adversarial robustness and invariance to rotations and translations. Specifically, the more robust a model is to larger perturbations the less robust it is to rotations and translations. Thus the problems expressed by Eq. 3.6 and Eq. 3.1 seem to be orthogonal at least for translation and rotation.

Chapter 4

Methodology

In this chapter, we will present our analysis of the sensitivity in terms of invariance and adversarial robustness of CNNs under different settings. In all of our investigations, we focus the attention on L_∞ since this is a standard metric used in adversarial machine learning. Our main hypothesis is the one followed in [47] and [27] that, given a small parameter space (that of the transformation T) and the generalization ability of the deep neural networks, adversarial training with weak adversarial data might be sufficient to achieve robustness. This means that we consider robustness to transformations with respect to the average case as opposed to the worst case as in the scenario of L_p adversarial raw pixel-wise perturbations. In section 4.2, we introduce energy-based models and provide our detection algorithm.

4.1 Adversarial Training and Data Augmentation

Data augmentation is a common technique to improve the generalization of CNNs. [51]. Adversarial Training can be considered a special application of data augmentation. Indeed, as described earlier, the model is trained so that during training it sees adversarial data (Eq. 3.1); among other things, one can decide whether to train the model only on adversarial data or to conduct mixed training with adversarial and clean data.

4.1.1 Training with Gaussian Noise

The Gaussian, or "normal", distribution is an important probability distribution since it turns out to be an accurate model for noise in many communication systems. There are numerous properties of the Gaussian distribution. One of these is that the two values μ (mean) and σ (standard deviation) completely characterize the distribution.

Definition 1 : The probability density function of Gaussian distribution is :

$$\mathcal{G}(x) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{\frac{-(x-\mu)^2}{(2\sigma^2)}} \tag{4.1}$$

Definition 2 : A d dimensional vector random variable $\boldsymbol{X} = [x_1, .., x_d]$ is said to have a multivariate normal (or Gaussian) distribution with mean $\boldsymbol{\mu} \in \mathbb{R}^d$ and covariance matrix $\boldsymbol{\Sigma} \in S^d_{++}$ if its probability density function is given by :

$$\mathcal{G}(\boldsymbol{X}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{|\boldsymbol{\Sigma}|^{1/2}\sqrt{(2\pi)^d}} \exp\Big(-\frac{1}{2}(\boldsymbol{X} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{X} - \boldsymbol{\mu})\Big) \tag{4.2}$$

We denote with $\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ such vector.
A nice property is that one can show that a d-dimensional Gaussian vector with mean $\boldsymbol{\mu}$ and a diagonal covariance matrix $\boldsymbol{\Sigma} = diag(\sigma_1^2, .., \sigma_d^2)$ is equal to a set of d independent Gaussian random variables with mean μ_i and variance σ_i^2.
In contrast to previous work, we focus our attention on possible trade-offs and relations for various σ and ϵ between Gaussian noise sensitivity and adversarial robustness respectively. Training a classifier $F_{\boldsymbol{\theta}}$ on data affected by Gaussian noise can be formalized as follows :

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{x},y)\sim D} \ \mathbb{E}_{\boldsymbol{\delta}\sim\mathcal{N}(0,\, I\sigma^2)} \ \Big[\mathcal{L}\Big(F_{\boldsymbol{\theta}}(\boldsymbol{x} + \boldsymbol{\delta}),\ y\Big)\Big] \tag{4.3}$$

In high dimension, Gaussian vectors with variance σ^2 are highly concentrated on the hyper-sphere of radius $\sigma\sqrt{d}$ (d dimension of $\boldsymbol{x}$) [18]. Thus, the above Eq. 4.3 can be approximately written as :

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{x},y)\sim D} \ \mathbb{E}_{||\boldsymbol{\delta}||_2 = \sigma\sqrt{d}} \ \Big[\mathcal{L}\Big(F_{\boldsymbol{\theta}}(\boldsymbol{x} + \boldsymbol{\delta}),\ y\Big)\Big] \tag{4.4}$$

By comparing Eq. 3.2 with Eq. 4.4, as noted in [18] and [33], we can see that there is a close relationship between adversarial robustness and Gaussian noise. Specifically, while adversarial training is based on worst-case perturbation, training on Gaussian noise data approximates the average case at distance $\sigma\sqrt{d}$. In other words, adversarial training seeks the worst-case perturbation among all possible perturbations inside a ball of ϵ radius whereas training on Gaussian noise perturbed data approximates an average case over scattered perturbations on a hyper-sphere.

4.1.2 Training with Random Transformations

Given a transformation T_γ, one can perform training using T as data augmentation. It is important to decide how many of the data samples the transformation will be applied to. To keep the trade-off balanced between clean and adversarial accuracy a common choice is to apply the T transformation to about half the data.
Algorithm 2 describes high-level training with data augmentation, which we recall corresponds to the worst-of-k case with k equal to 1.

Algorithm 1: Transformation Adversarial Training

Input : F_θ, training set X, transformation T, $0 \leq p \leq 1$, $s \in \mathbb{N}$, $\gamma_1^*, \gamma_2^* \in \mathbb{R}$
- Initialize the weights of F_θ
repeat
  0) Read minibatch $B = \{\boldsymbol{x_1}, .., \boldsymbol{x_b}\}$ from X and $B^* = \{\}$
  1) for *each* $\boldsymbol{x}$ *in* B **do**
    choose randomly $\gamma \in [\gamma_1^*, \gamma_2^*]$
    $\boldsymbol{x^*} = T_\gamma(\boldsymbol{x})$ with probability p othw. ($\boldsymbol{x^*} = \boldsymbol{x}$ with probability $1 - p$)
    add $\boldsymbol{x^*}$ to B^*
  end
  2) Do one optimization step on F_θ using B^* and loss $\mathcal{L}$
until F_θ *converges*;

4.1.3 Training with Rademacher Noise Data

It has been repeatedly shown that adversarial training with FGSM is not effective in making a model robust to PGD attacks [1], [57] and that it suffers from the phenomenon of *catastrophic overfitting*. However, to our best knowledge, it has not been studied whether an FGSM attack with a random direction (and not that given by the gradient sign) is effective. Formally, we call Rademacher the following probability mass function :

$$r(x) = \begin{cases} \frac{1}{2} & \text{if x = -1} \\ \frac{1}{2} & \text{if x = 1} \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

We note that using this distribution we can create a perturbation that is identical in concept to the one that FGSM performs but whose values (+1 or -1) do not follow that of the sign of the gradient with respect to $\boldsymbol{x}$ but are chosen randomly.

4.1.4 Adversarial Training with PGD

We follow the implementation of Madry et al. in [40]. Indeed, nowadays there are numerous approaches for adversarial training [4]. The approach that still provides excellent results is adversarial training with PGD proposed in [40].
Algorithm 2 describes the high-level implementation of the procedure.

4.2 Energy-based Models

Energy-based models (EBMs) [37] are based on the idea that any probability density $p_\theta(\mathbf{x})$ for $\boldsymbol{x} \in \mathbb{R}^d$ can be expressed via a Boltzmann distribution as :

$$p_\theta(\boldsymbol{x}) = \frac{\exp\left(-E_\theta(\boldsymbol{x})\right)}{Z(\boldsymbol{\theta})} \tag{4.6}$$

Algorithm 2: L_∞ PGD Adversarial Training

Input : model F_θ, training set X, $s \in \mathbb{N}$, cross-entropy loss $\mathcal{L}$
- Initialize the weights of F_θ
repeat
  0) Read mini-batch $B = \{\boldsymbol{x_1}, .., \boldsymbol{x_b}\}$ from X and $B^* = \{\}$
  1) for *each* $\boldsymbol{x}$ *in* B **do**
    $\boldsymbol{x}^* = clip_\epsilon\ (\boldsymbol{x}^* + \alpha\ sign(\ \nabla_{\boldsymbol{x}^*}\{\mathcal{L}(F_\theta(\boldsymbol{x}^*),\ y)\}\ )\ )$ \\repeat s times
    add $\boldsymbol{x}^*$ to B^*
  end
  3) Do one optimization step on F_θ using B^* and loss $\mathcal{L}$
until F_θ *converges*;

where $E_\theta(\boldsymbol{x})$ is called energy function, modeled as a neural network, that maps each input $\boldsymbol{x}$ to a scalar. $Z(\boldsymbol{\theta})$ is the normalizing constant, known as the partition function, such that $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ is a valid probability function; the challenge for training EBMs is approximating the constant $Z(\boldsymbol{\theta})$. Training an EBM is performed via maximum likelihood estimation by minimizing the negative log-likelihood of the data :

$$\mathcal{L}_{ML}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x}\sim P_D}\left[-\log p_{\boldsymbol{\theta}}(\boldsymbol{x})\right] \tag{4.7}$$

Nevertheless, the latter is not effortless and several sampling methods have been designed to approximate it efficiently. Specifically, it is known that the derivative of Eq. 4.7 w.r.t $\boldsymbol{\theta}$ is:

$$\nabla_{\boldsymbol{\theta}}\mathcal{L}_{ML}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x}^+\sim P_D}\left[\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\boldsymbol{x}^+)\right] - \mathbb{E}_{\boldsymbol{x}^-\sim p_{\boldsymbol{\theta}}}\left[\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\boldsymbol{x}^-)\right] \tag{4.8}$$

Intuitively, the above gradient reduces the energy of the positive data samples $\boldsymbol{x}^+$, while boosting the energy of the negative samples $\boldsymbol{x}^-$ from $p_{\boldsymbol{\theta}}$.
Unfortunately, for EBMs we can't sample from $p_{\boldsymbol{\theta}}$, therefore several MCMC methods have been proposed to sample from that density function [52]. For instance, in [22] and [13] Stochastic Gradient Langevin Dynamics (SGLD) has been used to train EBMs using gradient information. SGLD sample with the following algorithm :

$$x_0 \sim p_0(\boldsymbol{x}) \quad , \quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \frac{\alpha}{2}\frac{\partial\ E_{\boldsymbol{\theta}}(\boldsymbol{x}_t)}{\partial \boldsymbol{\theta}} + \alpha\ \epsilon \quad , \quad \epsilon \sim \mathcal{N}(0,\ 1) \tag{4.9}$$

where $p_0(\boldsymbol{x})$ is typically a Uniform distribution over the input domain. Thus, we can summarize that the training of EBMs consists of two steps: generating approximate data samples from $p_{\boldsymbol{\theta}}$ using MCMC methods and optimizing the model parameters to increase the energy of these samples and decreasing the energy of the real samples via gradient descent [60].

4.2.1 Energy Function

In machine learning, a classification problem with K classes is carried out using a model $F_{\boldsymbol{\theta}}$ that maps the input $\boldsymbol{x}$ into a vector of size K known as the logits vector. These logits are used to compute the so-called Softmax function :

$$p_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}) \;=\; \frac{\exp\Big(F_{\boldsymbol{\theta}}(\boldsymbol{x})\;[y]\Big)}{\sum_{k=1}^{K}\exp\Big(F_{\boldsymbol{\theta}}(\boldsymbol{x})\;[k]\Big)} \tag{4.10}$$

where $F_{\boldsymbol{\theta}}(\boldsymbol{x})\;[k]$ indicates the k-th index of $F_{\boldsymbol{\theta}}(\boldsymbol{x})$.
Recently, Grathwohl et al. made a connection between CNN classifiers and EBMs [22]. They show that an energy-based model is hidden inside a standard classifier and we can re-interpret the logits of a model $F_{\boldsymbol{\theta}}$ to build an EBM to compute $p(\boldsymbol{x}, y)$ and $p(\boldsymbol{x})$:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, y) \;=\; \frac{\exp\Big(F_{\boldsymbol{\theta}}(\boldsymbol{x})\;[y]\Big)}{Z(\boldsymbol{\theta})} \tag{4.11}$$

where $Z(\boldsymbol{\theta})$ is the unknown normalizing constant and $E_{\boldsymbol{\theta}}(\boldsymbol{x}, y) = -F_{\boldsymbol{\theta}}(\boldsymbol{x})[y]$ is the energy function. Following [22], we thus compute $p_{\boldsymbol{\theta}}(y\;|\boldsymbol{x})$ as :

$$p_{\boldsymbol{\theta}}(y\;|\boldsymbol{x}) \;=\; \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, y)}{p_{\boldsymbol{\theta}}(\boldsymbol{x})} \;=\; \frac{\exp\Big(E_{\boldsymbol{\theta}}(\boldsymbol{x}, y)\;\Big)\;Z(\boldsymbol{\theta})}{\sum_{k=1}^{K}\exp\Big(F_{\boldsymbol{\theta}}(\boldsymbol{x})\;[k]\Big)\;Z(\boldsymbol{\theta})} \tag{4.12}$$

Notice now that the logarithm sum of the logits of any classifier can be re-used to define the energy function at a data point $\boldsymbol{x}$ as :

$$E_{\boldsymbol{\theta}}(\boldsymbol{x}) \;=\; -\log\sum_{k=1}^{K}\exp\Big(F_{\boldsymbol{\theta}}(\boldsymbol{x})[k]\Big) \tag{4.13}$$

Adversarial training and EBMs have been studied in [60] in which the authors analyze the generative capabilities of adversarially trained models from an energy perspective. Specifically, they note that adversarial examples find high-energy examples and observed that the adversarial training optimization aims to lower the energy by optimizing model parameters. Although very intriguing the results produced in [60], do not show how the data of an adversarial model and a normal model are actually distributed from the point of view of energy, and also do not touch on the topic of attacks with different parameters.

4.2.2 Detection of Adversarial Perturbations via Energy Function

Based on the observations that adversarial attacks have a large L_2 norm of logits (simultaneously also larger energy)—see Fig. 4.1— we propose the construction of an adversarial example detection algorithm that exploits this property. The basis of our algorithm is the ability to detect adversarial perturbations through simple norm (or energy function) checking. This idea has already been implemented in [39] to detect out-of-distribution data using the energy function. Yet, to the best of our knowledge, it was not yet applied to adversarial example detection. In our scenario, we use the energy function to detect adversarial examples. This detection algorithm, described in Algorithm 3, detects adversarial perturbations at inference time without any additional computation overhead like adversarial training, gradient computations, etc.

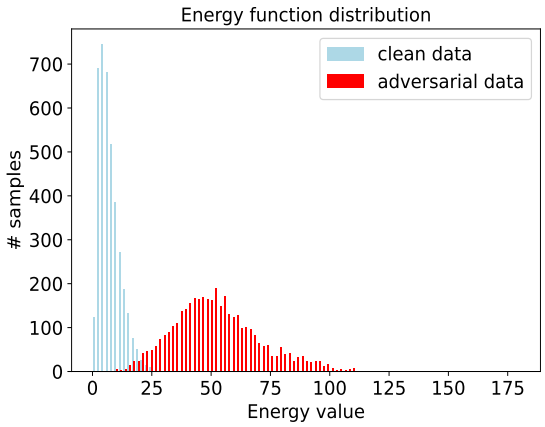


Figure 4.1. Energy function histogram of a non-robust model. Adversarial data have larger energy than clean data.

Algorithm 3: Adversarial Perturbation Detection Algorithm

Input : model F_θ, Validation Set V, Test Set S
Output : $\boldsymbol{c} \in \{0,1\}^{\|S\|}$
0) Estimate a threshold t_v on V based on some metrics we want to optimize
1) for *each* $\boldsymbol{x}$ *in* S **do**
  if $\mid E_{\boldsymbol{\theta}}(\boldsymbol{x}) \mid \leq t_v$ **then**
    return 0 \\it is not an adversarial example
  else
    return 1 \\it is an adversarial example
  end
end

4.2.3 Attacking The Energy-based Detection Algorithm

While the energy-based detection algorithm is practical for adversarial data that have high energy such as those produced by PGD; the question remains whether it is possible to generate adversarial perturbations that fool the method described above. This, in practice, means asking whether it is possible to produce adversarial examples that have an energy $E_{\boldsymbol{\theta}}(\boldsymbol{x})$ similar to that of the natural data, thus low energy—Fig. 4.1.
Our proposal is to force an attack to generate adversarial data with low energy by constructing an objective that removes the energy gap between clean and adversarial data. Specifically, we define a new modified version of the PGD attack that takes into account the value of energy. Given a classifier $F_{\boldsymbol{\theta}}$ the optimization we define is :

$$\arg\max_{\boldsymbol{\delta}} \Big[\mathrm{CE}(F_{\boldsymbol{\theta}}(\boldsymbol{x}+\boldsymbol{\delta}), y) - \lambda \left| E_{\boldsymbol{\theta}}(\boldsymbol{x}) \right| \Big] \tag{4.14}$$

where CE is the cross-entropy function and λ is a scalar that balances the effect of the energy term. We call this variant of PGD as low-energy PGD (LE-PGD). The Algorithm 4 describes the LE-PGD procedure. Using LE-PGD attack we force the

creation of adversarial examples that have a similar energy to natural data.

Algorithm 4: LE-PGD Algorithm

Input : model $F_{\boldsymbol{\theta}}$, data point $(\boldsymbol{x}, y)$, ϵ, steps, α, λ
Output : adversarial image $\boldsymbol{x}^*$
0) Define $\mathcal{L} = \text{CE} - \lambda |E_{\boldsymbol{\theta}}(\boldsymbol{x})|$
1) $\delta \sim$ Uniform(-ϵ, ϵ)
2) $\boldsymbol{x}^* = \boldsymbol{x} + \boldsymbol{\delta}$
3) for *i=1 to steps* **do**
  $\boldsymbol{x}^* = \boldsymbol{x}^* + \alpha \, \text{sign}\Big(\nabla_{\boldsymbol{x}^*} \mathcal{L}\Big(F(\boldsymbol{x}^*), y\Big)\Big)$
  $\boldsymbol{\delta} = \min(\max(\boldsymbol{x}^* - \boldsymbol{x}, -\epsilon), \epsilon)$ //clamp the perturbation
  $\boldsymbol{x}^* = \boldsymbol{x} + \boldsymbol{\delta}$
end

Take-home message

In this chapter, we have described the procedures for dealing with the problem of neural network robustness described in Eq. 3.2 and Eq. 3.7.

- We have seen how adversarial training is a very special form of data augmentation.
- Under the assumption that adversarial examples are OOD data we reinterpreted adversarial examples in the context of EBM. Finding experimental validation that adversarial data are high-energy points.
- By exploiting energy we show that it is possible to construct a detector of adversarial attacks at inference time with basically no effort and no modification of the main CNN classifier.
- By investigating the detector, we provide a method, namely LE-PGD, to produce adversarial examples with low energy, similar to that of natural data.

Chapter 5

Beyond Accuracy for Robustness Evaluation

The most used metric for classification performance in machine learning is accuracy. That is, simply count the correct number of predictions out of the total test data. Robustness, however, is a very complex property of neural networks; it is not clear how to really define robustness (both adversarial robustness and transformations) in a way that is independent of models, data given, and types of attacks. Therefore, an objective measurement of the robustness of neural networks is lacking.
To see how simple accuracy is not an effective metric for robustness just think of two classifiers F and G where in the last layer they exploit a Softmax operation, so it means they return as output a probability distribution over the classes; given now a data point $(\boldsymbol{x}, y)$ and an adversarial example $\boldsymbol{x}^*$ (it does not matter to define now how $\boldsymbol{x}^*$ was obtained from $\boldsymbol{x}$) if we measure the performance of F and G only using accuracy it means noting whether simply compute $\arg\max(F(\boldsymbol{x}^*)) = y$ or $\arg\max(G(\boldsymbol{x}^*)) = y$. We note that in computing accuracy we lose the information of the vector $F(\boldsymbol{x}^*)$ and $G(\boldsymbol{x}^*)$ containing probabilities for each class. Thus, according to the accuracy metric F and G that predict wrongly $\boldsymbol{x}^*$ are considered equally non-robust regardless of the values $F(\boldsymbol{x}^*)$ and $G(\boldsymbol{x}^*)$, Fig 5.1.
We now begin to look at another metric that should be considered with neural networks when discussing robustness and invariance.

5.1 Calibration

In real-world applications, it is important that neural networks are not only accurate but also effectively indicate with true probability their predictions. In other words, the probability output by the network regarding the occurrence of an event should reflect the true frequency of that event: for example, if a diagnosis system says that 1000 patients have cancer with a probability of 0.05, about 50 of them should actually have cancer. A model that behaves this way is said to be well-calibrated. Intuitively one might think that the Softmax layer embedded in a generic neural network, which returns a vector of probabilities spread over the classes, does this, and thus the values that a neural network returns actually describe the real probability distribution. In reality, this is not the case and there are numerous papers that have

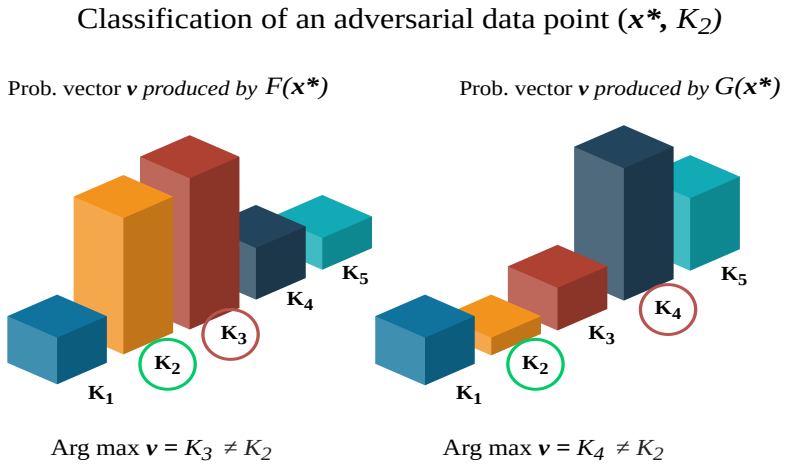


Figure 5.1. Accuracy metric in classification. What matters is only whether $\arg\max \boldsymbol{v}$ is equal to the ground-truth. The model F performs better than G when classifying $\boldsymbol{x}^*$; nevertheless for the accuracy metric they are considered equivalent.

observed that neural networks are not well-calibrated [23].
This means that the models we use are often overconfident (or underconfident).

- **Overconfidence**: is the characteristic whereby in classification a model on a data point $\boldsymbol{x}$ is highly confident of its prediction (so it returns a high probability for a certain class K_i) though the ground-truth label of $\boldsymbol{x}$ being different from K_i.
- **Underconfidence**: is the opposite situation of overconfidence whereby the model tends to underestimate its correct predictions.

Formally, a model is perfectly calibrated if, for any probability value p, a prediction of a class with confidence p is correct $100 * p$ percent of the time.
Mathematically a perfect calibration is expressed as :

$$\mathcal{P}\Big(\hat{y} = y \mid \hat{p} = p\Big) = p \quad \forall p \in [0, 1] \tag{5.1}$$

Where $\hat{y}$ is the predicted label of the data point $\boldsymbol{x}$ and $\hat{p}$ is the confidence of that label. Eq. 5.1 cannot be calculated because we have a finite number of data samples, so we need techniques to estimate the calibration.

5.1.1 Reliability Diagrams

Reliability diagrams are a visual representation of calibration [23]. To compute reliability diagrams, firstly the model's predictions are divided into bins based on the confidence score of the winning class. Next, for each bin, it is computed the average confidence and average accuracy. Finally, for each bin, we plot the difference between the accuracy and the confidence, previously computed, Fig. 5.2.

Ideally, if the model is well calibrated the accuracy and confidence are equal; in this scenario, the model' confidence score can be interpreted as a real probability score and the reliability diagram should represent the identity function. If the model is not calibrated, there is going to be a gap between accuracy and confidence, this fact leads in the reliability diagrams some red bars in the diagonal; the larger the gap, the bigger the bar; in particular, if the red bar goes below the diagonal, it means the confidence is larger than the accuracy and the model is too confident in its predictions, if the red bar goes above the diagonal, the average accuracy is larger than the average confidence and the model is underconfident, Fig. 5.2.
Formally, given n predictions, we can group these predictions into M bins. Let B_m be the set of sample indices whose prediction confidence falls in the $m-th$ bin. We define accuracy and average confidence as follows :

$$\mathrm{Acc}(B_m) = \frac{1}{B_m} \sum_{i \in B_m} \mathbf{1}(\hat{y}_i = y_i) \tag{5.2}$$

$$\mathrm{Conf}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \hat{p}_i \tag{5.3}$$

whereas previously $\hat{p}_i$ is the confidence score of a sample i. We say that the model is well calibrated if $\mathrm{Acc}(B_m) = \mathrm{Conf}(B_m)\ \forall m \in \{1, .., M\}$.
Reliability diagrams are usually plotted together with a confidence histogram showing the confidence distribution of various data samples. In this way, we can see how many data samples fall within a certain bin.
Sometimes it is more convenient to have scalar metrics that immediately tell us the calibration of a model than reliability diagrams.

5.1.2 Expected Calibration Error (ECE)

The Expected Calibration Error (ECE) is a simple metric that measures the calibration error. It is defined as follows :

$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} \Big| \mathrm{Acc}(B_m) - \mathrm{Conf}(B_m) \Big| \tag{5.4}$$

5.1.3 Maximum Calibration Error (MCE)

The Maximum Calibration Error (MCE) is a metric that measures the largest calibration gap across all bins, formally defined as :

$$\mathrm{MCE} = \max_{m \in \{1,..,M\}} \Big| \mathrm{Acc}(B_m) - \mathrm{Conf}(B_m) \Big| \tag{5.5}$$

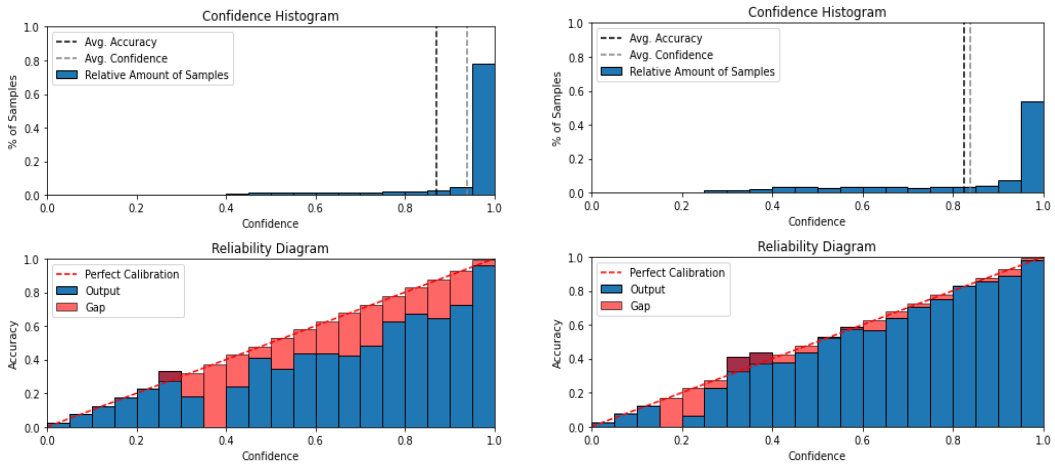


Figure 5.2. Reliability diagrams for two different models. On the top, the confidence histogram shows the number of data samples in terms of their predicted confidence. On the bottom, the two reliability diagrams with 20 bins. The diagrams show that the left model is more overconfident than the one on the right; the red bars are much larger in the left graph.

Take-home message

The most used metric for classification performance in machine learning is accuracy. In real-world applications, it is important that neural networks are not only accurate but also effectively indicate with true probability their predictions. In other words, the probability output by the network regarding the occurrence of an event should reflect the true frequency of that event.

- A model that behaves this way is said to be well-calibrated.
- Guo et al. in [23] demonstrated that modern CNNs are not well-calibrated.
- Some methods to measure calibration are reliability diagrams, ECE and MCE.

Chapter 6

Experiments and Results

In this chapter, we show our tests and experiments. We start analyzing our experimental setting and then describing our observations and considerations.

6.1 Experimental Setting

Dataset	Training size	Test size	Sample dimension
CIFAR-10	50k	10k	32×32
Imagenette	8190	3925	160×160

Table 6.1. Datasets details.

6.1.1 Datasets and Models

For our experiments, we used CIFAR-10 and Imgenette [26]. Due to the low dimensionality of the images (32×32), CIFAR-10 allows fast training and development. On the other hand, the low dimensionality of the data does not allow in general, to perform an in-depth analysis of transformations leading to strong distortions; in addition, having high-dimensional data allows for a more complex problem since adversarial examples can be found more easily [9]. For these reasons, we took advantage of the Imagenette dataset [26], a subset of ImageNet consisting of 10 classes containing in total about 8k images. On Imagenette, we applied center-cropping by reducing the image dimension to 150×150 to decrease the computational overhead. We used ResNet10, a model belonging to a well-known DNN architecture known to have excellent performances in several computer vision tasks. Regarding the Imagenette dataset, in order to once again reduce the complexity of training and adversarial training, we changed the stride parameter of the first layer of the architecture; this leads to smaller feature maps. We changed from stride = 1 for CIFAR-10 to stride = 3 for Imagenette. All the code is publicly available online [1].

[1] https://github.com/senad96/TransformRobustness

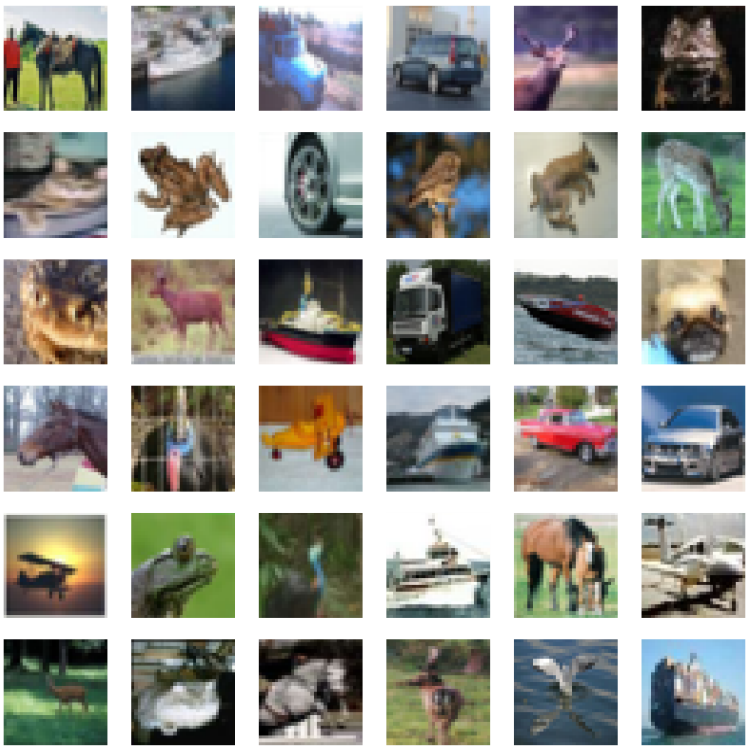


Figure 6.1. Some images of CIFAR-10 dataset.

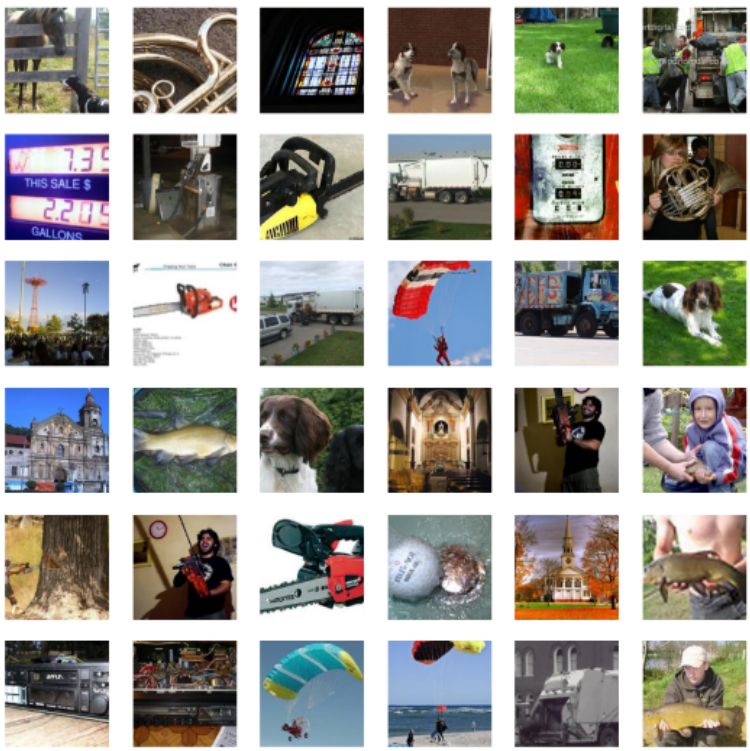


Figure 6.2. Some images of Imagenette dataset.

6.1.2 Training, Validation and Testing details

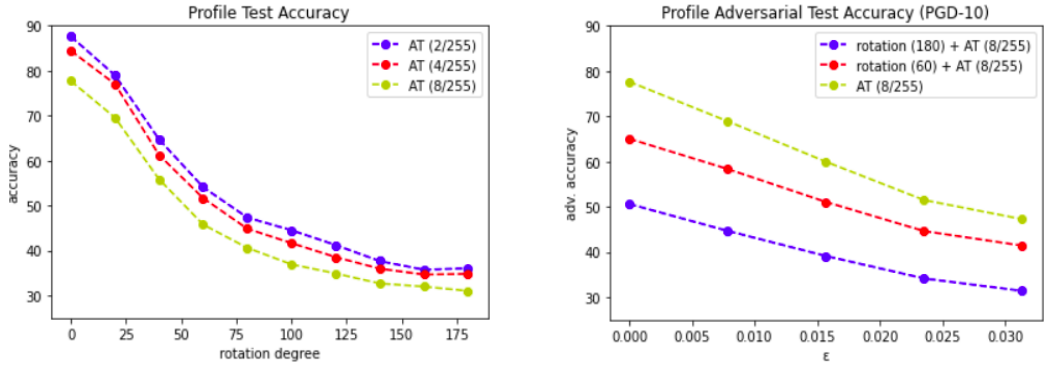
We use as optimization algorithm Adam with a learning rate equal to 0.001. We do not use any regularizer in the training procedure, whereas we use batch normalization of the ResNet architecture. The batch size during training was always 64 for all models. Adversarial Training is performed using PGD, as described in Algorithm 2, with 5 steps (PGD-5); while robustness evaluation, at test time, is executed with PGD 10 steps (PGD-10) with $stepsize = 1/255$. For this process, we leveraged the open-source library *torchattacks* [31].
On CIFAR-10 the models were trained for 100 epochs on the clean data with data augmentation while with 60 epochs for adversarial training.
On Imagenette, the baseline model on clean data was trained for 60 epochs and all other variants either with data augmentation or adversarial training for 60 epochs as well.
The learning rate was reduced by a factor of 10 for all models on Imagenette at the 30th epoch, while for CIFAR-10 it was reduced by a factor of 10 at the 30th and 80th epochs.
For validation, we randomly selected from the training set 10k images for CIFAR-10 and 500 images for Imagenette. For all models, the early stopping technique was applied, that is, choosing the model that had the best performance on the validation set. Models are always evaluated on the full test set.

6.2 Analysis of the Trade-off between Adversarial Robustness and Other Transformations

To investigate the trade-off, we represented the performance of each model using a test accuracy profile for both adversarial transformation robustness and the standard adversarial robustness. For each T_γ transformation, we have produced two graphs showing the relationship between the two types of robustness.
The first graph of a generic T_γ transformation shows how the accuracy of a model varies by applying the T_γ transformation to the entire test set, then different curves are plotted for different training procedures; each curve describes a model trained with only one specific data augmentation (described in the legends) along with the horizontal flipping operation. In other words, the graph shows how the accuracy of a model trained with a specific transformation varies as the γ parameter changes. We call this graph as *Profile test accuracy*.
The second graph, on the other hand, shows how the robustness of a specifically trained model varies as the ϵ parameter of the L_∞ PGD attack changes. The entire test set was used here as well. We call this graph as *Profile adversarial test accuracy*.
We emphasize that we will consider only the robustness with respect to the L_∞ PGD attack; as described earlier in the previous chapter, the use of PGD is fairly standard given the fact that the algorithm is powerful enough to produce strong adversarial examples and easily fool *common* DNNs. We now begin to show our experiments for each transformation.

6.2.1 Rotation

In [27] it has already been proved that there is a trade-off for rotation (and also for translation), nevertheless we start our analysis by reproducing the results of [27] by verifying the results with our implementation with ResNet10. We recall that by rotation we mean a 2D rotation of the image in the plane with only one degree of freedom: the angle. Fig. 6.3 shows the two profiles of robustness. Specifically, the two graphs show 3 different trained models as described in the legend. Adversarial training was performed on 100% of the data for each minibatch with PGD-5 and $stepsize = 2/255$. Rotation data augmentation, when needed, was applied to 50% of the data for each minibatch. For instance, the $AT(8/255)$ symbol indicates that the ResNet10 model was adversarially trained with PGD-5 using $\epsilon = 8/255$. While the abbreviation $rotation(60) + AT(8/255)$ describes ResNet10 adversarial trained with PGD-5 and $\epsilon = 8/255$ applying rotation data augmentation on 50% of the minibatch with degree γ randomly chosen from [-60,60].
What we can notice is that if one tries to increase adversarial robustness by performing harder PGD adversarial training with a larger ϵ the robustness with respect to rotation decreases—see Fig. 6.3 **(a)**. On the contrary, if we increase the rotation robustness with training augmentation the model is less robust to PGD attacks—Fig. 6.3 **(b)**. This result indicates a trade-off between rotation robustness and adversarial robustness. Our results are consistent and comparable with those discovered in [27].



(a) **(b)**

Figure 6.3. On CIFAR-10 **(a)** Profile test accuracy. Curves denote models adversarial trained with different ϵ. **(b)** Profile adversarial test accuracy. Different curves denote different trained models.

6.2.2 Perspective

We continue our analysis, with the perspective transformation. Here again, we show the two profiles of accuracy. The perspective transform, with the standard Pytorch implementation, has a parameter called *scale factor (or distortion factor)*, ranging from 0 to 1, that controls the strength of the perspective operation. Fig. 6.4 shows the three profiles of robustness. Adversarial training was performed on 100% of the data for each minibatch with PGD-5 and $stepsize = 2/255$ (1/255 for $\epsilon = 2/255$).

Perspective data augmentation, when needed, was applied to 50% of the data for each minibatch. The abbreviation $perspective(0.5) + AT(8/255)$ describes ResNet10 adversarially trained with PGD-5 and $\epsilon = 8/255$ using perspective data augmentation on 50% of minibatch data with a distortion factor γ randomly chosen from $[0, 0.5]$. As for rotation, we discover that even for perspective transformation there seems to be a trade-off, even though less strong than rotation. It can be seen from the graph Fig. 6.4 **(a)** that by performing adversarial training with larger ϵ the robustness to perspective decreases. Vice versa, by trying to increase the robustness to perspective using stronger training augmentation the adversarial robustness decreases — see Fig. 6.4 **(b,c)**. We also see that indeed a perspective transformation is adversarial for different models.

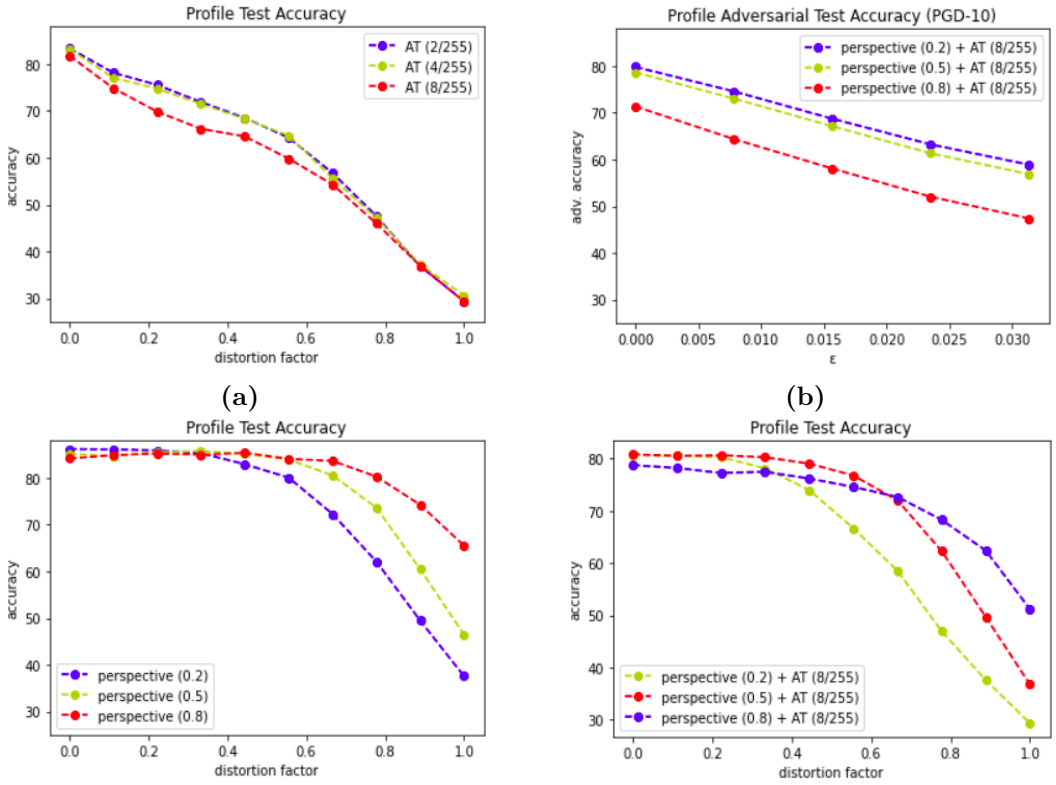


Figure 6.4. On Imagenette, the perspective trade-off. **(a)** Profile test accuracy. Curves denote models adversarial trained with different ϵ. **(b)** Profile adversarial test accuracy. Different curves denote different trained models. **(c)** Profile test accuracy. Curves denote models trained with different data augmentation **(d)** Different curves denote adversarially trained models with different data augmentation.

6.2.3 Motion Blur

For the motion blur transformation, we applied the same considerations as for the perspective. We exploited the implementation of the Kornia library [45]. The motion blur, in this implementation, has several hyper-parameters; for our analysis, we

decided to consider the filter size.
Adversarial training was performed on 100% of the data for each minibatch with PGD-5 and *stepsize* = 2/255 (1/255 for $\epsilon = 2/255$). Motion blur data augmentation, when needed, was applied to 50% of the data for each minibatch. Thus, the abbreviation $MB(3) + AT(8/255)$ describes ResNet10 adversarially trained with PGD-5 using motion blur augmentation on 50% of minibatch data with a filter size $\gamma = 3$. In Fig. 6.5 we present our results.
We find that, in contrast to perspective and rotation, on its own adversarial training increases accuracy on motion blur data significantly, in particular, the larger the ϵ during adversarial training the more robust the model is to the motion blur transformation; the gain is considerable, for motion blurred test data with a *filter size* = 13, adversarial training with $\epsilon = 8/255$ provides $\sim +10\%$ accuracy compared to adversarial training on $\epsilon = 2/255$ and $\sim +20\%$ accuracy respect the baseline model (no training augmentations and no adversarial training)—see Fig. 6.5 **(a)**. We see in Fig. 6.5 **(b)** that if we perform adversarial training on motion blur data slightly decreases adversarial robustness. Moreover, the $MB(13)$ model is the most robust to the motion blur transformation for a very large range of filter sizes, but the $MB(13) + AT(8/255)$ model is less robust by about 5% in adversarial accuracy at $\epsilon = 2/255$ than $AT(8/255)$ and $MB(3) + AT(8/255)$. Our results suggest that only part of the trade-off is true, namely that while with adversarial training alone the robustness of motion blur increases, adversarial training using more aggressive motion blur data leads to a loss of adversarial robustness.

6.2.4 Gaussian Noise

We analyzed the case with Gaussian augmentation with different types of σ. Since Gaussian noise robustness and adversarial robustness have a close relationship [18], we train with the same setting, so 100% of data are augmented with Gaussian noise. Adversarial training, as usual, was performed with PGD-5 and *stepsize* = 2/255 (1/255 for $\epsilon = 2/255$). Summarizing, the abbreviation $GN(0.1) + AT(8/255)$ describes ResNet10 adversarially trained with PGD-5 and $\epsilon = 8/255$ using Gaussian noise data augmentation applied on 100% of the minibatch data with $\sigma = 0.1$. The *Baseline* stands for clean training without augmentations.
Many interesting things can be seen from Fig 6.6. First, we produced 3 *Profile test accuracy* and 2 *Profile adversarial test accuracy* to investigate in detail all the different configurations and relations.
From Fig. 6.6 **(a)** it is clearly visible that by increasing the ϵ in simple adversarial training the robustness to Gaussian noise tends to increase (the strong green curve indicating adversarial training with $\epsilon = 8/255$ is above the other curves indicating adversarially trained models with lower ϵ). The difference is substantial, the solely trained adversarial model with $\epsilon = 8/255$ has about $\sim +15\%$ accuracy for Gaussian noise data with $\sigma = 0.3$ compared to an adversarial trained model with $\epsilon = 4/255$. In the same direction, the plot Fig. 6.6 **(c)** shows that training with Gaussian noise data increases adversarial robustness. The larger the σ parameter of augmentation during training, the greater the adversarial robustness seems to be (e.g. training with Gaussian noise $\sigma = 0.2$ offers $\sim +25\%$ adversarial accuracy at a $\epsilon = 2/255$ compared to the baseline model). In Fig. 6.6 **(d)** is shown that combining Gaussian

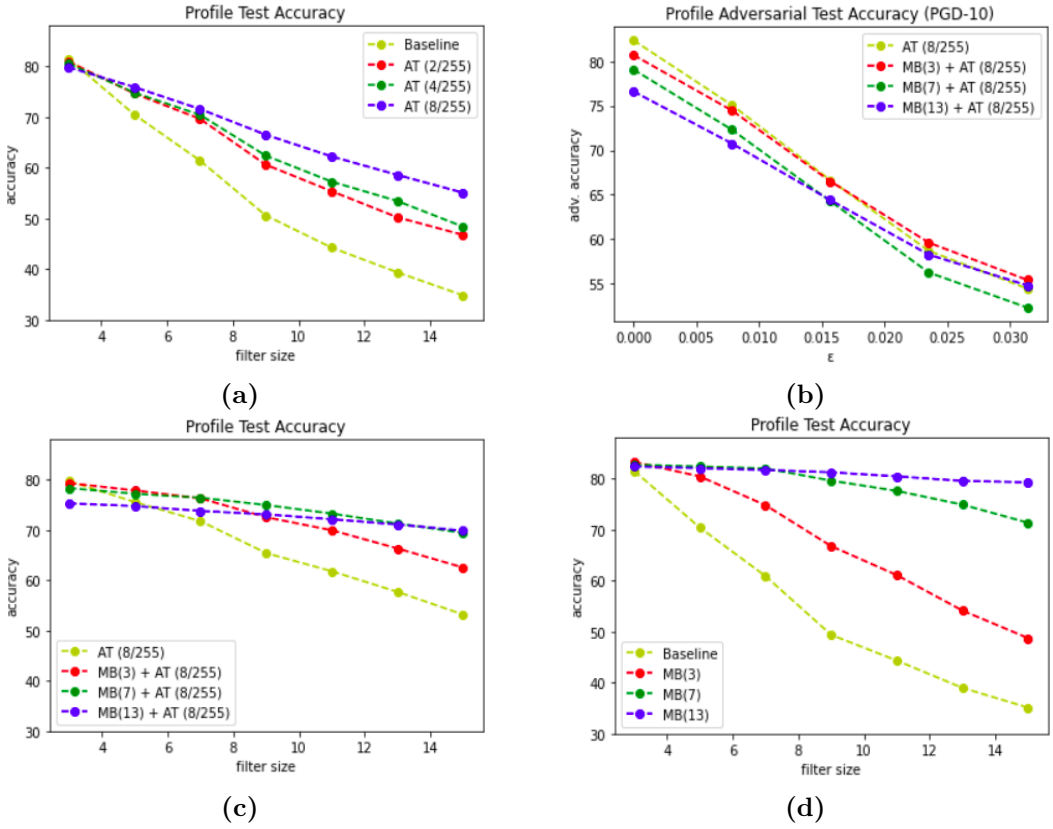


Figure 6.5. On Imagenette, the motion blur trade-off. **(a)** Profile test accuracy. Curves denote models adversarially trained with different ϵ. **(b)** Profile adversarial test accuracy. Different curves denote different trained models. **(c)** Profile test accuracy. Curves denote models adversarially trained with different ϵ and different training augmentation.

noise data augmentation and adversarial training has an improvement in adversarial robustness only for a small σ Fig. 6.6 **(b)**. However, combining both modes the robustness of Gaussian noise is significantly higher than in models trained with only Gaussian augmentation. For example, the $GN(0.2) + AT(8/255)$ curve in Fig. 6.6 **(e)** achieves better accuracy over the whole range of σ than models only trained with weaker or equal training augmentation Fig. 6.6 **(b)**.
These results show no trade-off between Gaussian noise robustness and adversarial robustness. In addition, our experiments show that adversarial training with Gaussian noise perturbed data increases Gaussian noise robustness for a wider variety of σ — see black curve Fig. 6.6 **(e)**, thus providing broader Gaussian noise robustness than data augmentation alone.

6.2.5 Rademacher Noise

We also produced the two accuracy profiles for the Rademacher noise. The analysis performed is similar to that of other transformations such as perspective. The abbreviation $Rad(8/255)$ stands for ResNet10 trained with data perturbed with Rademacher noise with a $\epsilon = 8/255$ applied to 50% of the data in a minibatch.

Specifically, in our analysis, we evaluated whether there is a relationship between Rademacher noise and Gaussian noise as well as with adversarial robustness. In Fig. 6.7, we can see the two profiles. What can be seen is that Rademacher noise does not help in any kind of robustness to either Gaussian noise or adversarial robustness—Figs. 6.7 **(a,b)**. This result, against expectations, shows that the perturbation produced by the FGSM method is different from Rademacher noise and that random perturbations inspired by FGSM are not effective.

6.3 Analyzing the Uncertainty of CNN using Calibration Diagrams

As described earlier, calibration is an essential property of predictive models, especially in safety-critical environments. To better analyze the calibration of our models, we decided to use expected calibration error (ECE), which is more concise than reliability diagrams. Nevertheless, we can see a reliability diagram in Fig. 6.8 that shows adversarial training provides models that are more calibrated than standard training.
Additionally, to examine the calibration of a model as training techniques and test set change we produced a *Profile ECE graph*. Given a T_γ transformation, *Profile ECE graph* shows how the expected calibration error (ECE) of a model varies as the parameter γ of T varies on the test set.
In Fig. 6.9 we show for each transformation the *Profile ECE graph*. There are numerous observations to be made. First, we can say a certain degree of confidence that adversarial training makes a model more calibrated; in fact for each transformation adversarial training (red curve) showed a lower ECE for almost all values of the parameters of the transformations considered. It is important to note that data augmentation also makes a model more calibrated to the corrupted data than the baseline model. Surprisingly, adversarial training sometimes turns out to produce more calibrated models even than data augmentation.
Last but not least, the graph Fig. 6.9 **(e)** shows that a model trained with Gaussian noise is more calibrated to the perturbed data with PGD than the baseline model. This is another confirmation that Gaussian noise robustness and adversarial robustness are two closely related phenomena.

6.4 Interpreting Adversarial Perturbations using Energy-Based Models

Here we analyze the difference between adversarial and natural data in terms of energy in an EBM. Understanding how the energy values are distributed is important since it can give us insights into how classifiers digest different types of data and what kind of logits they produce. In particular, what we do is analyze how the L_2 norm of logits and the energy $E_{\boldsymbol{\theta}}(\boldsymbol{x})$ of adversarial and clean data vary. To visualize the results we produce a histogram for both energy $E_{\boldsymbol{\theta}}(\boldsymbol{x})$ and L_2 norm. In Fig. 6.10 we can see how there is a significant difference between adversarial and natural data for a non-robust model. In particular, the adversarial data keep both a higher

logits L_2 norm and higher energy $E_{\boldsymbol{\theta}}(\boldsymbol{x})$ (in absolute value) than the natural data. Even more interesting, however, is to note what characteristics the adversarial data have with different norms. What emerges —see Figs. 6.12, 6.13— is that adversarial data points with low logits L_2 norm tend to have bright colors and monochrome backgrounds, while adversarial data points with high logits L_2 norm tend to have dark colors and more complex shapes.
The detection algorithm, defined in Algorithm 3, exploits this property and is able to detect adversarial attacks at inference time via the energy function $E_{\boldsymbol{\theta}}(\boldsymbol{x})$.
An important observation to make is that if a model is robust, that is, adversarial training has been applied to it, the energy property disappears —see Fig. 6.11. So it is no longer possible to apply any detection to adversarial examples with the energy-based approach. We leave the investigation of this phenomenon as feature work. In the next section, we analyze the performances of the detection algorithm using different L_p norm and energy function $E_{\boldsymbol{\theta}}(\boldsymbol{x})$.

6.5 Adversarial Detection Algorithm Analysis

We evaluate Algorithm 3 in the Imagenette dataset, highly studied by us in previous analyses. We emphasize that the use of Imagenette is a good benchmark since this dataset contains realistic and high-resolution images, contrary to standard datasets such as CIFAR-10 and MNIST.
The algorithm, firstly, estimates in the validation set a threshold that balances the TPR and the FNR. The approach we use is the G-mean. G-Mean is a metric that, when optimized, will seek a balance between sensitivity and specificity. Specifically, G-mean is defined as :

$$\text{G-Mean} = \sqrt{\text{Sensitivity} \times \text{Specificity}} \tag{6.1}$$

We tested our algorithm for different settings of the L_p norm and energy $E_{\boldsymbol{\theta}}(\boldsymbol{x})$. Sometimes it is useful and good practice to use so-called temperature scaling in the logits level when doing Softmax. This technique consists of simply dividing the logits with a constant t. Choosing a proper constant allows you to make a model more calibrated [23]. In order to choose a suitable constant that maximizes the performance of the detection algorithm we estimate the temperature scaling parameter t on the validation set maximizing the area under the receiver operating characteristic curve (AUROC). In Fig. 6.15 we can see the ROC curve of the detection algorithm on the validation set (Imagenette) using the energy function $-E_{\boldsymbol{\theta}}(\boldsymbol{x})$ with different temperature scaling parameter; It is visible that changing temperature scaling factor does not affect too much the performances, specifically the AUROC.
The metrics L_1, L_2, L_∞, and $E_{\boldsymbol{\theta}}(\boldsymbol{x})$ have comparable discriminate power to detect adversarial examples—see Fig. 6.15. The performance of the detector, on Imagenette, is remarkable. In fact, using the PGD-30 attack, the detector manages to achieve a 98% detection rate (computed as the number of adversarial examples correctly detected divided by the total number of adversarial examples) and a 1% false positive rate. Table 6.2 shows the performances of the energy-based detection algorithm on

Table 6.2. Detection rate (DR%), false positive rate (FPR%) and false negative rate (FNR%) for the energy-based detector against PGD attacks on Imagenette.

Attack	ϵ	DR	FPR	FNR
PGD-5	8/255	83%	19%	17%
PGD-10	8/255	93%	4%	7%
PGD-30	8/255	98%	1%	2%
PGD-50	8/255	98%	0.7%	2%
PGD-50	16/255	99%	0.02%	0.1%

the Imagenette dataset using different PGD strengths. Fig. 6.16 shows the confusion matrix of the detection algorithm.

6.6 Low-Energy PGD Attack Analysis

Surprised by the excellent results of the energy-based approach, we put ourselves in the shoes of an attacker trying to evade the defense. LE-PGD, described in the previous chapter, is a modified version of the PGD attack. Indeed, it introduces a new hyper-parameter λ that balances and controls the energy term in the new loss function. Our tests find that LE-PGD is at least as powerful as standard PGD, as indeed we expect since LE-PGD is a generalization of it. The adversarial examples produced by LE-PGD tend to have low energy —see Fig. 6.17.
Thus, LE-PGD, with a right value of λ, completely evades the defense of the energy-based detection —see Fig. 6.17. Note that λ can be estimated on the validation set. We strongly believe that a better investigation of LE-PGD, its connections with EBMs, and neural network robustness is needed.

Take-home message

In this chapter, we showed that for neural networks :

- adversarial robustness and robustness to common transformations can be tasks orthogonal to each other. Specifically, data augmentation does not allow maximum accuracy for both types of robustness.
- Data augmentation and adversarial training are practical approaches for providing well-calibrated models for common perturbations.
- By connecting EBM and adversarial examples, we illustrated that energy can be used to detect adversarial examples at inference time.
- A new variant of PGD attack, called LE-PGD, is able to produce adversarial data with low energy to bypass the energy-based adversarial example detector.

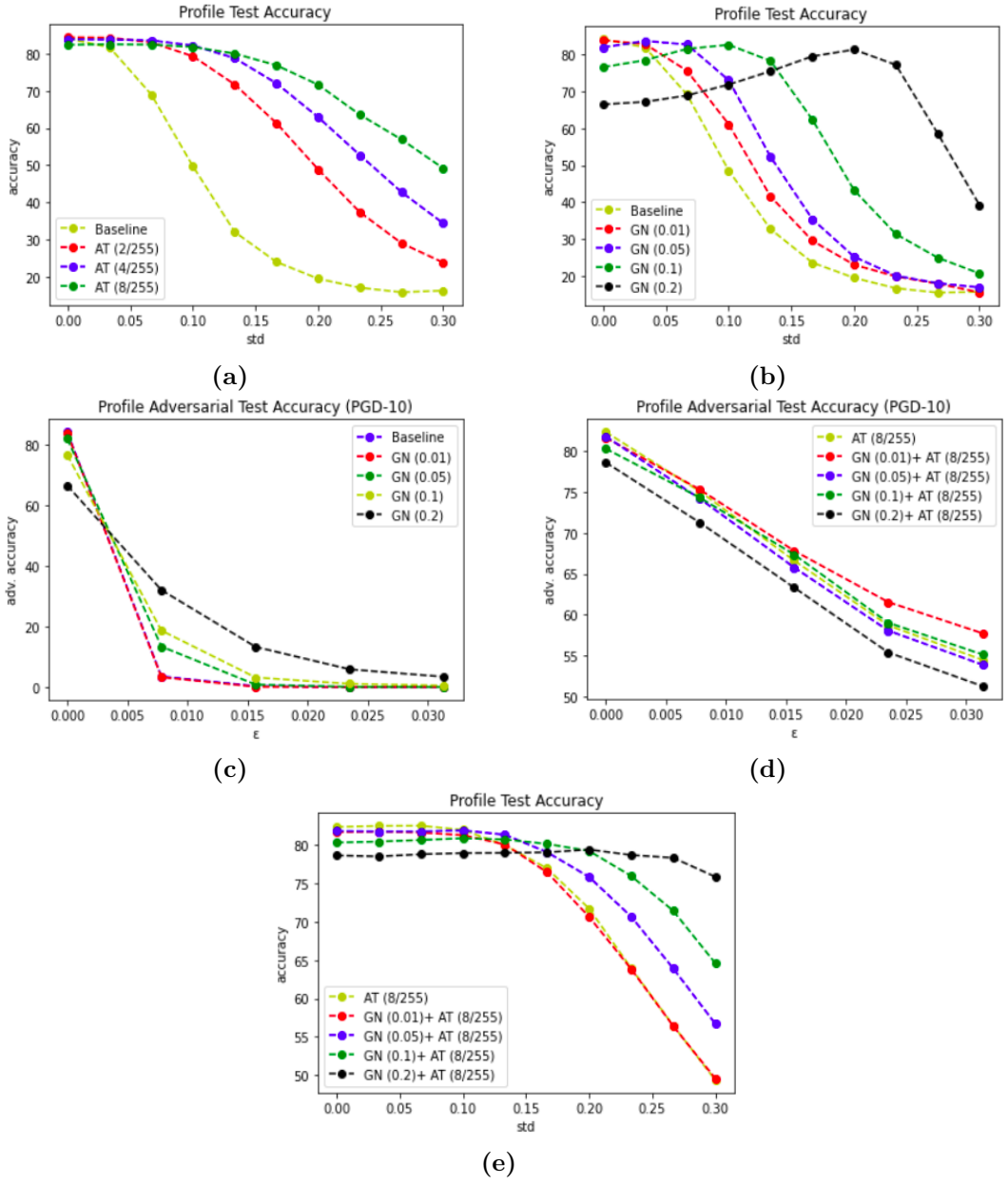
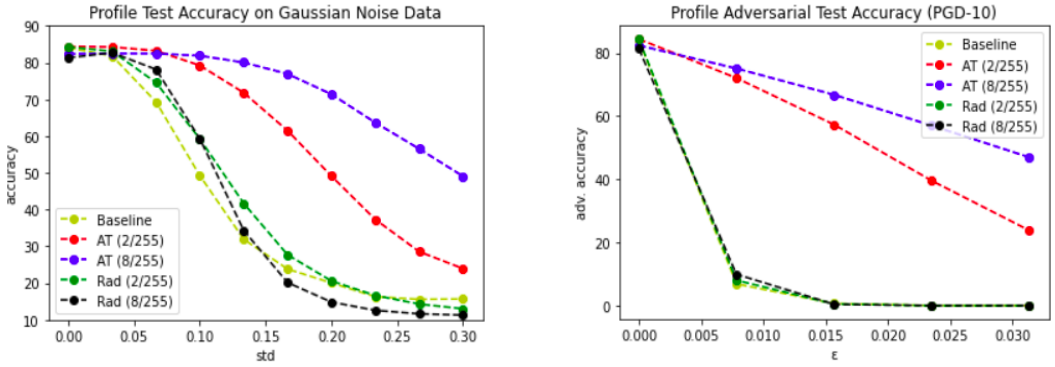
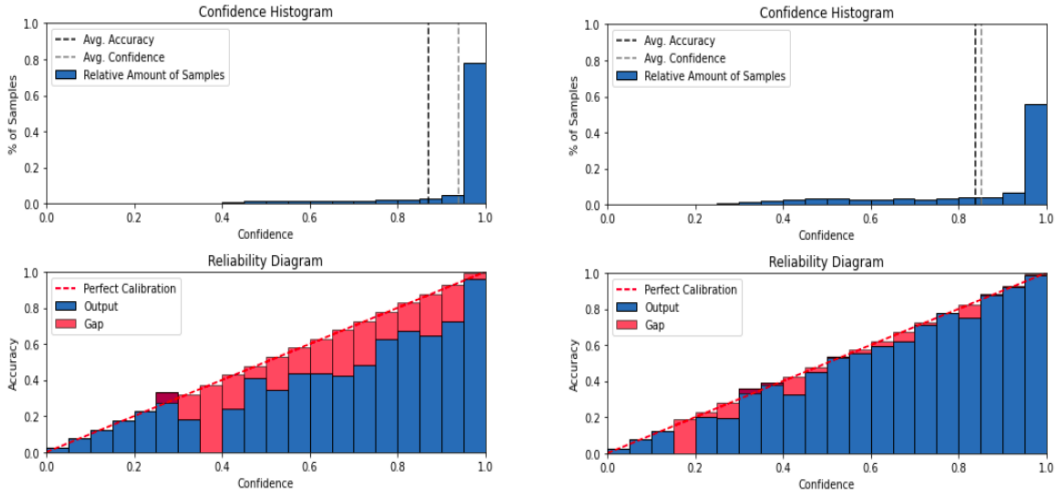


Figure 6.6. On Imagenette, Gaussian noise, and adversarial robustness analysis. **(a)** Profile test accuracy. Curves denote models adversarial trained with different ϵ. **(b)** Profile test accuracy. Different curves denote models trained with different Gaussian augmentation. **(c)** Profile adversarial test accuracy. Curves denote models trained with different Gaussian augmentations. **(d)** Profile adversarial test accuracy. Curves denote models adversarial trained with different ϵ and different Gaussian augmentation. **(e)** Profile test accuracy on Gaussian noise data. Curves denote models adversarial trained with different ϵ and different Gaussian augmentation.



(a) **(b)**

Figure 6.7. On Imagenette, Rademacher noise, Gaussian noise and adversarial robustness analysis. **(a)** Profile test accuracy on Gaussian noise data. Curves denote models adversarially trained with different ϵ and different augmentations. **(b)** Profile adversarial test accuracy. Curves denote models adversarially trained with different ϵ and different Rademacher augmentation.



(a) **(b)**

Figure 6.8. Two reliability diagrams. **(a)** The reliability diagram of a baseline model trained on Imagenette. **(b)** The reliability diagram of an adversarially trained model on Imagenette with $\epsilon = 4/255$. The model on the right is more calibrated.

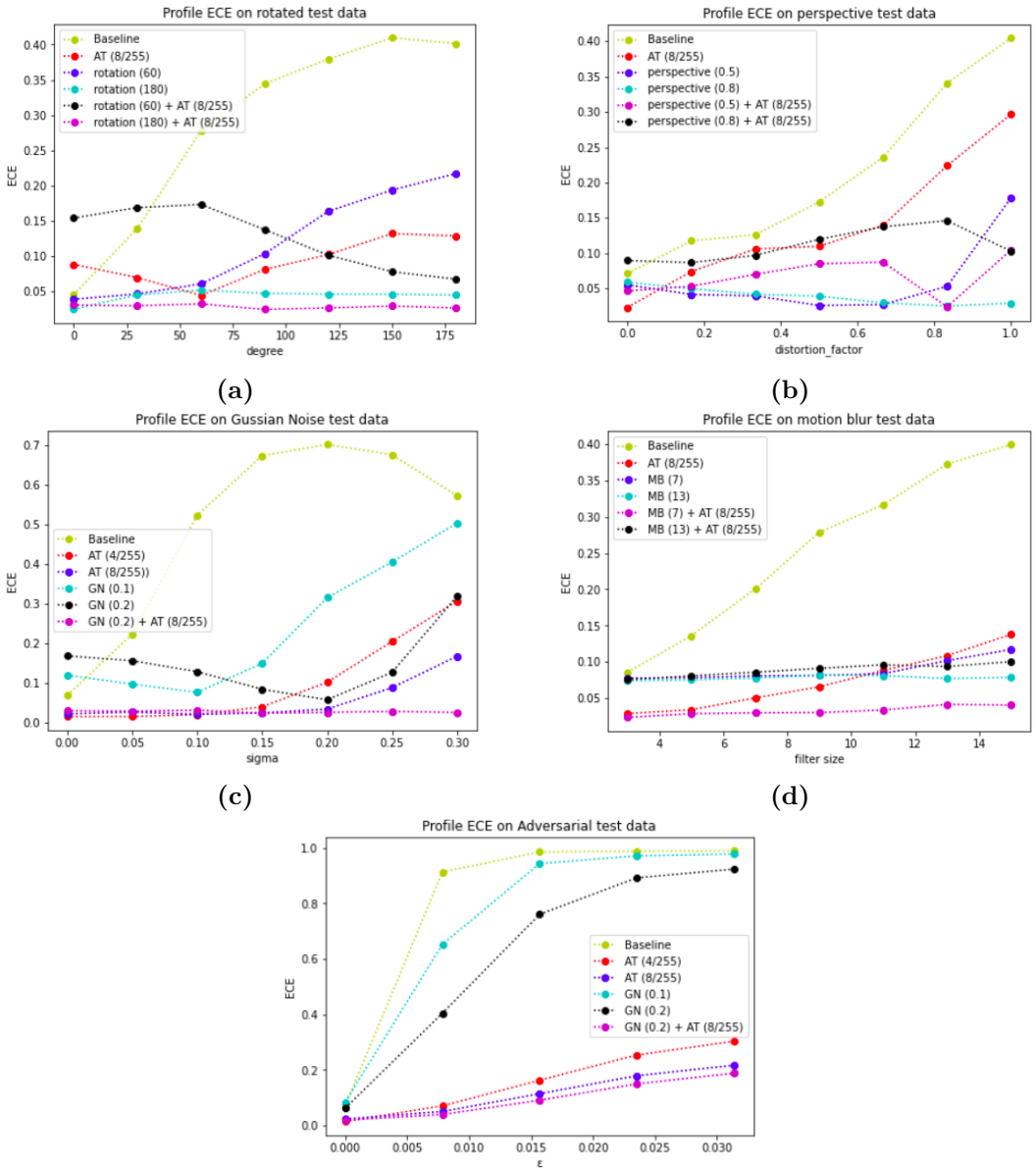
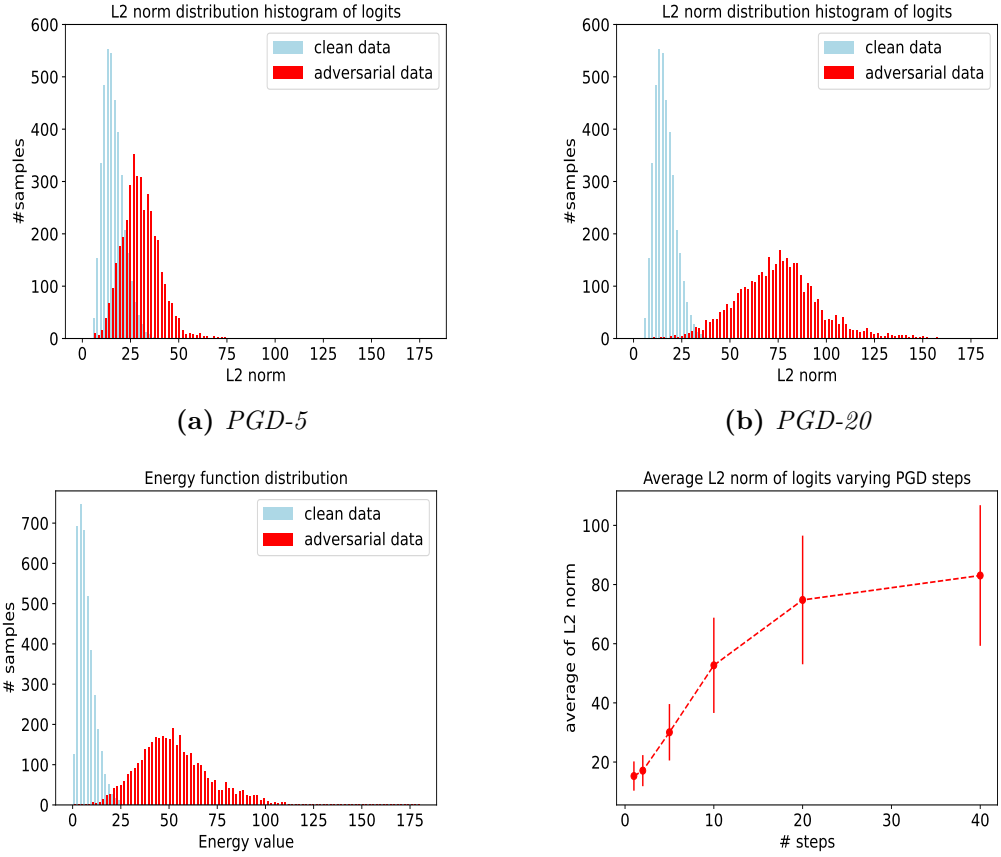


Figure 6.9. Calibration ECE profile on the perturbed test set. **(a)** On CIFAR-10 Profile ECE graph on rotated data. Curves denote models trained with different data augmentation or just adversarially trained. **(b)** On Imagenette, Profile ECE graph on perspective data. Curves denote models trained with different data augmentation or just adversarially trained. **(c)** On Imagenette, Profile ECE graph on Gaussian noise data. Curves denote models trained with different data augmentation or adversarially trained. **(d)** On Imagenette, Profile ECE graph on motion blurred data. Curves denote models trained with different data augmentation or adversarially trained. **(e)** On Imagenette, Profile ECE graph on PGD-10 perturbed data varying ϵ. Curves denote models trained with different data augmentation or adversarially trained.



(a) *PGD-5* **(b)** *PGD-20*

(c) *PGD-20* **(d)**

Figure 6.10. On Imagenette, L_2 norm of logits and energy varying the number of PGD steps at $\epsilon = 8/255$. **(a)** The histogram of the test data on how the L_2 norm of logits is distributed. The adversarial data were produced using PGD-5. **(b)** The histogram of the test data on how the L_2 norm of logits is distributed. The adversarial data were produced using PGD-20. **(c)** The histogram of the test data on how the energy function $E_{\boldsymbol{\theta}}(\boldsymbol{x})$ is distributed. The adversarial data were produced using PGD-20. **(d)** The relation among the L_2 norm of logits and the number of steps of PGD attack. It is recognizable that the average of the L_2 norm of the logits of the adversarial data tends to increase as the number of PGD steps increases.

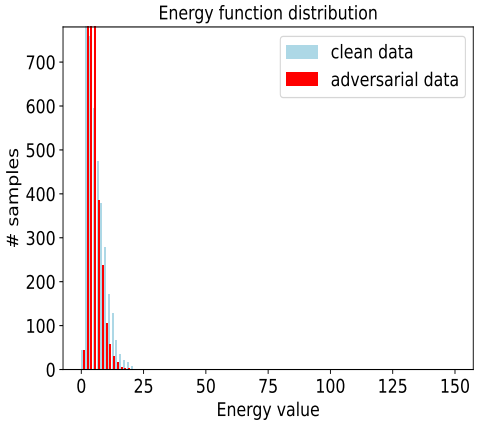


Figure 6.11. On Imagenette, the histogram of energy distribution for a **robust** model (adversarially trained). The difference in energy between adversarial and natural data is lost.

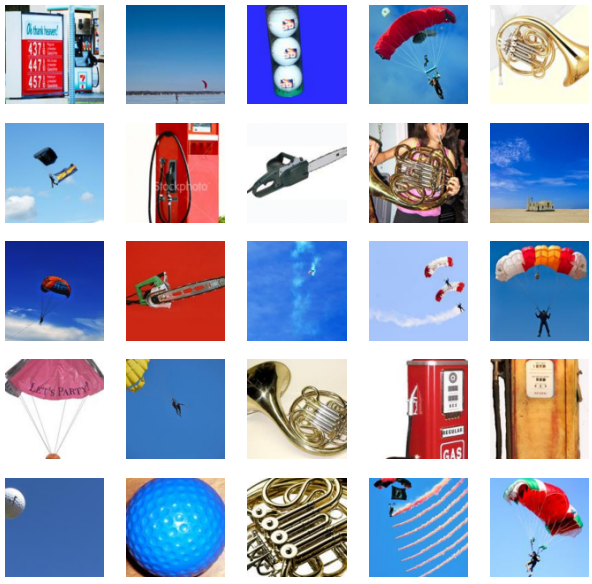


Figure 6.12. Some Imagenette images that have low logit L_2 norm.

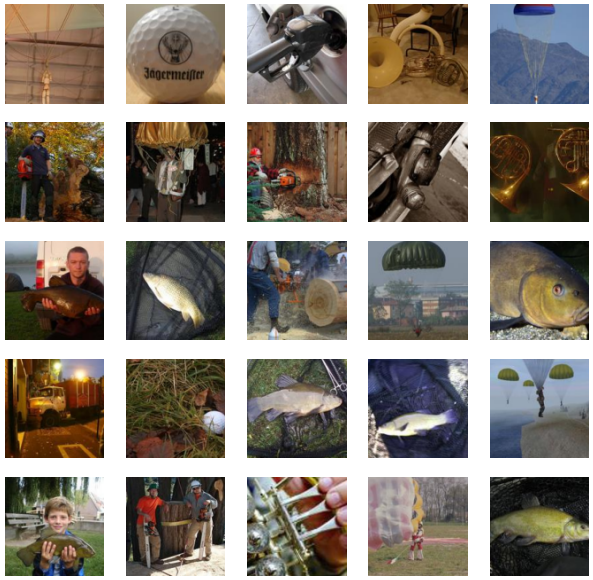


Figure 6.13. Some Imagenette images that have high logit L_2 norm.

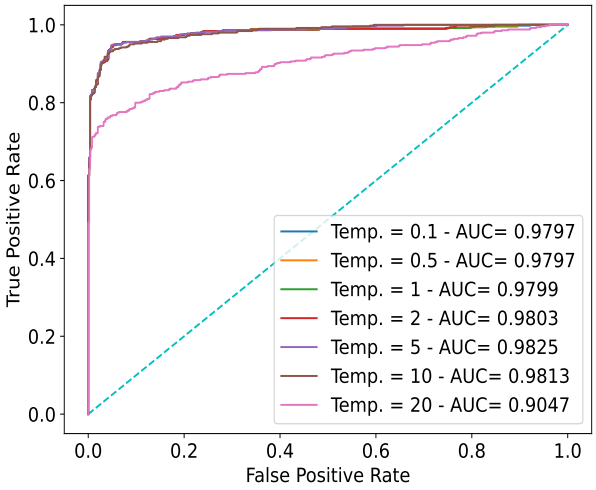


Figure 6.14. On validation set of Imagenette. ROC curve for different temperature scaling. The parameter $t = 5$ provides the best AUC.

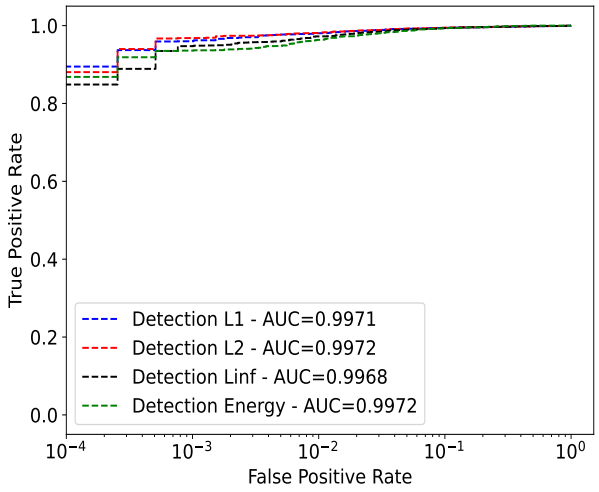


Figure 6.15. On test set of Imagenette. ROC curve of the detection algorithm using different metrics. All four metrics perform well in detecting adversarial data.

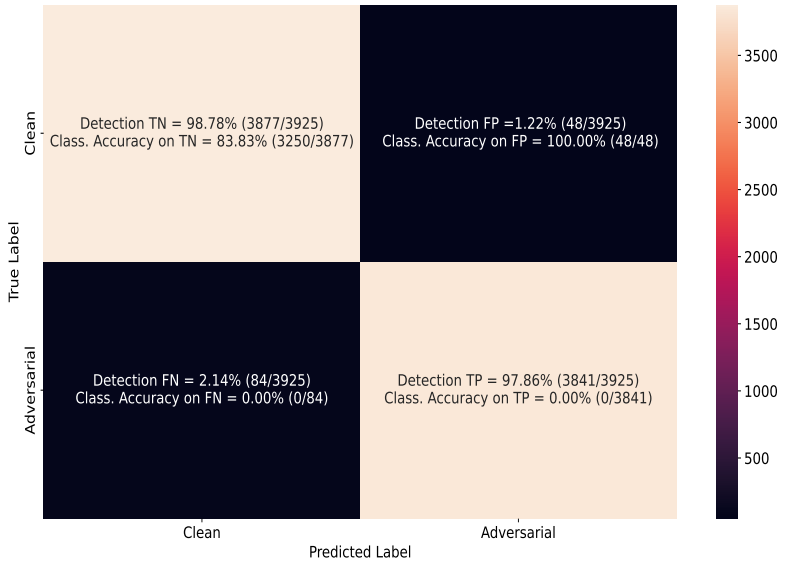
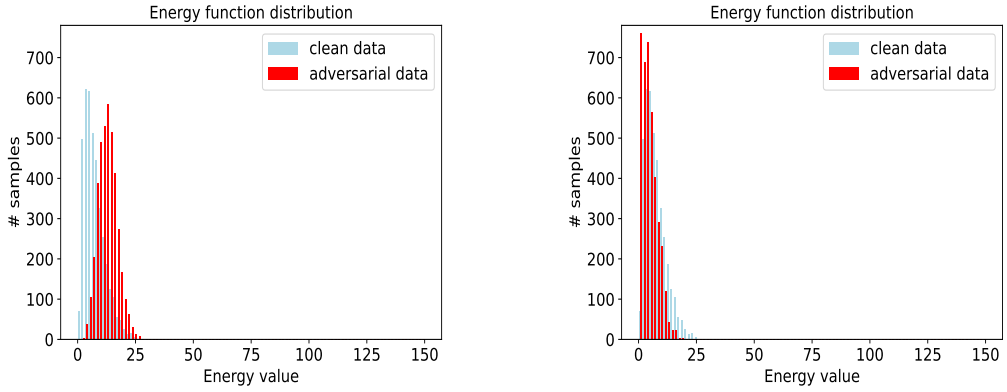


Figure 6.16. On Imagenette test set, the confusion matrix describing the performances of the detection algorithm using energy $E_{\boldsymbol{\theta}}(\boldsymbol{x})$ on PGD-30. The percentage tells us the recognition accuracy (not detection accuracy) of the model for each type—TN, FP, FN, and TP— of data.



(a) **(b)**

Figure 6.17. Energy distribution on Imagenette. Adversarial data are produced by the LE-PGD algorithm with different λ hyper-parameter. On the left $\lambda = 2$. On the right $\lambda = 5$. LE-PGD produces adversarial data with low energy.

Chapter 7

Conclusion

In this master's thesis, we addressed a study of the sensitivity of neural networks to adversarial perturbations. We conducted several experiments using adversarial training with PGD perturbations and common transformations. By also conducting an analysis on the calibration of neural networks, we showed how adversarial training and data augmentation are two good methods to increase calibration. Our experiments indicate that it is not always possible to have, through data augmentation, both robustness to adversarial perturbations and common transformations. Under the assumption that adversarial data are OOD, we reinterpreted adversarial perturbations under the context of EBMs. With the experimental validation that adversarial examples have higher energy, we propose to use the energy function as an indicator to detect adversarial perturbations at inference time. The detector built over the energy function achieves excellent results on the high dimensional Imagenette data set. Trying to break the energy-based detection algorithm, we propose a variant of the PGD attack, which we call LE-PGD. This method allows us to produce adversarial data that has low energy, making the above detection algorithm worthless. However, there are a large number of questions that still remain outstanding, which surely can (and should) be addressed in future work. For example, what is the meaning of low-energy adversarial examples? Do they have any special properties? Is a model robust to these data different from a model robust to adversarial data produced by standard PGD? We believe that better exploring these and other questions could make new connections between EBMs and adversarial robustness.

Bibliography

[1] Maksym Andriushchenko and Nicolas Flammarion. Understanding and improving fast adversarial training. *Advances in Neural Information Processing Systems*, 33:16048–16059, 2020.

[2] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *International conference on machine learning*, pages 284–293. PMLR, 2018.

[3] Aharon Azulay and Yair Weiss. Why do deep convolutional networks generalize so poorly to small image transformations? *arXiv preprint arXiv:1805.12177*, 2018.

[4] Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang. Recent advances in adversarial training for adversarial robustness. *arXiv preprint arXiv:2102.01356*, 2021.

[5] Anand Bhattad, Min Jin Chong, Kaizhao Liang, Bo Li, and David A Forsyth. Unrestricted adversarial examples via semantic manipulation. *arXiv preprint arXiv:1904.06347*, 2019.

[6] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

[7] Nicholas Carlini and David Wagner. Audio adversarial examples: Targeted attacks on speech-to-text. In *2018 IEEE security and privacy workshops (SPW)*, pages 1–7. IEEE, 2018.

[8] Chayan Chatterjee, Linqing Wen, Foivos Diakogiannis, and Kevin Vinsen. Extraction of binary black hole gravitational wave signals from detector data using deep learning. *Physical Review D*, 104(6):064046, 2021.

[9] Nandish Chattopadhyay, Anupam Chattopadhyay, Sourav Sen Gupta, and Michael Kasper. Curse of dimensionality in adversarial examples. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

[10] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pages 1310–1320. PMLR, 2019.

[11] Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999. PMLR, 2016.

[12] 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.

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

[14] Ranjie Duan, Xingjun Ma, Yisen Wang, James Bailey, A Kai Qin, and Yun Yang. Adversarial camouflage: Hiding physical-world attacks with natural styles. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1000–1008, 2020.

[15] Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. Exploring the landscape of spatial robustness. In *International Conference on Machine Learning*, pages 1802–1811. PMLR, 2019.

[16] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1625–1634, 2018.

[17] Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, and Jose Garcia-Rodriguez. A review on deep learning techniques applied to semantic segmentation. *arXiv preprint arXiv:1704.06857*, 2017.

[18] Justin Gilmer, Nicolas Ford, Nicholas Carlini, and Ekin Cubuk. Adversarial examples are a natural consequence of test error in noise. In *International Conference on Machine Learning*, pages 2280–2289. PMLR, 2019.

[19] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[20] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[21] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[22] 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*, 2019.

[23] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

[24] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[25] Hossein Hosseini and Radha Poovendran. Semantic adversarial examples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1614–1619, 2018.

[26] Imagenette. `https://github.com/fastai/imagenette`. [Online; accessed 15-August-2022].

[27] Sandesh Kamath, Amit Deshpande, Subrahmanyam Kambhampati Venkata, and Vineeth N Balasubramanian. Can we have it all? on the trade-off between spatial and adversarial robustness of neural networks. In M. Ranzato, A. Beygelzimer, K. Nguyen, P. S. Liang, J. W. Vaughan, and Y. Dauphin, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 27462–27474. Curran Associates, Inc., 2021.

[28] Can Kanbak, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Geometric robustness of deep networks: analysis and improvement. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4441–4449, 2018.

[29] George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

[30] Baris Kayalibay, Grady Jensen, and Patrick van der Smagt. Cnn-based segmentation of medical imaging data. *arXiv preprint arXiv:1701.03056*, 2017.

[31] Hoki Kim. Torchattacks: A pytorch repository for adversarial attacks. *arXiv preprint arXiv:2010.01950*, 2020.

[32] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[33] Klim Kireev, Maksym Andriushchenko, and Nicolas Flammarion. On the effectiveness of adversarial training against common corruptions. *arXiv preprint arXiv:2103.02325*, 2021.

[34] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[35] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[36] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[37] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

[38] Zewen Li, Fan Liu, Wenjie Yang, Shouheng Peng, and Jun Zhou. A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE transactions on neural networks and learning systems*, 2021.

[39] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 33:21464–21475, 2020.

[40] 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.

[41] Tom M Mitchell and Tom M Mitchell. *Machine learning*, volume 1. McGraw-hill New York, 1997.

[42] Ali Bou Nassif, Ismail Shahin, Imtinan Attili, Mohammad Azzeh, and Khaled Shaalan. Speech recognition using deep neural networks: A systematic review. *IEEE access*, 7:19143–19165, 2019.

[43] Subash C Pakhrin, Bikash Shrestha, Badri Adhikari, and Dukka B Kc. Deep learning-based advances in protein structure prediction. *International Journal of Molecular Sciences*, 22(11):5553, 2021.

[44] Sylvestre-Alvise Rebuffi, Sven Gowal, Dan Andrei Calian, Florian Stimberg, Olivia Wiles, and Timothy A Mann. Data augmentation can improve robustness. *Advances in Neural Information Processing Systems*, 34, 2021.

[45] Edgar Riba, Dmytro Mishkin, Daniel Ponsa, Ethan Rublee, and Gary Bradski. Kornia: an open source differentiable computer vision library for pytorch. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3674–3683, 2020.

[46] Hadi Salman, Mingjie Sun, Greg Yang, Ashish Kapoor, and J Zico Kolter. Denoised smoothing: A provable defense for pretrained classifiers. *Advances in Neural Information Processing Systems*, 33:21945–21957, 2020.

[47] Vikash Sehwag, Jack W Stokes, and Cha Zhang. Beyond $l\_\{p\}$ norms: Delving deeper into robustness to physical image transformations. In *MILCOM 2021-2021 IEEE Military Communications Conference (MILCOM)*, pages 189–196. IEEE.

[48] 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! *Advances in Neural Information Processing Systems*, 32, 2019.

[49] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition.

In *Proceedings of the 2016 acm sigsac conference on computer and communications security*, pages 1528–1540, 2016.

[50] Dinggang Shen, Guorong Wu, and Heung-Il Suk. Deep learning in medical image analysis. *Annual review of biomedical engineering*, 19:221, 2017.

[51] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[52] Yang Song and Diederik P Kingma. How to train your energy-based models. *arXiv preprint arXiv:2101.03288*, 2021.

[53] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[54] Kalaivani Sundararajan and Damon L Woodard. Deep learning for biometrics: A survey. *ACM Computing Surveys (CSUR)*, 51(3):1–34, 2018.

[55] 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.

[56] Ke Wang, Bin Fang, Jiye Qian, Su Yang, Xin Zhou, and Jie Zhou. Perspective transformation data augmentation for object detection. *IEEE Access*, 8:4935–4943, 2019.

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

[58] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, pages 7472–7482. PMLR, 2019.

[59] Zhengyu Zhao, Zhuoran Liu, and Martha Larson. Towards large yet imperceptible adversarial image perturbations with perceptual color distance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1039–1048, 2020.

[60] Yao Zhu, Jiacheng Ma, Jiacheng Sun, Zewei Chen, Rongxin Jiang, Yaowu Chen, and Zhenguo Li. Towards understanding the generative capability of adversarially robust classifiers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7728–7737, 2021.