Neurocomputing 514 (2022) 162–181

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Adversarial attacks and defenses in deep learning for image recognition: A survey

Jia Wang, Chengyu Wang, Qiuzhen Lin, Chengwen Luo, Chao Wu, Jianqiang Li *

Shenzhen University, 3688 Nanhai Avenue, Nanshan District, Shenzhen 518060, Guangdong Province, China

ARTICLE INFO

Article history:
Received 18 April 2022
Revised 9 July 2022
Accepted 4 September 2022
Available online 9 September 2022

Keywords:
Deep neural network
Adversarial attack
Adversarial defense
Robustness

ABSTRACT

In recent years, researches on adversarial attacks and defense mechanisms have obtained much attention. It's observed that adversarial examples crafted with small malicious perturbations would mislead the deep neural network (DNN) model to output wrong prediction results. These small perturbations are imperceptible to humans. The existence of adversarial examples poses great threat to the robustness of DNN-based models. It is necessary to study the principles behind it and develop their countermeasures. This paper surveys and summarizes the recent advances in attack and defense methods extensively and in detail, analyzes and compares the pros and cons of various attack and defense schemes. Finally we discuss the main challenges and future research directions in this field.

1. Introduction

Deep Neural networks (DNNs) [1] have been shown to perform extremely well on many machine learning tasks, such as computer vision [2,3], speech recognition [4] and natural language processing [5,6]. However, recent researches found that the DNNs have a fatal flaw: a small amount of specially designed perturbations would fool the model, while these perturbations are typically invisible or visible but inconspicuous [7]. These perturbed inputs are called adversarial samples/examples, which enable attackers to disrupt the expected behavior of models, leading to undesirable consequences and security risks for these systems to be deployed in the real world, especially in areas that are safety-oriented, such as autonomous driving [8], traffic sign recognition [9], Image Guidance Technology [10], face recognition [11–13], etc.

In the image classification domain, the adversarial example usually corresponds to a benign picture. The two pictures are almost indistinguishable under the observation of the human perception, but for a well-trained neural network classifier, they belong to different categories. Adversarial attacks have two basic design goals: one is to mislead the target model to output error results, and the other is to make the perturbations inconspicuous, which is usually constrained by the l_p norm. To achieve these goals, various kinds of attack methods have been proposed: gradient-based methods, such as FGSM [14], PGD [15], MIT [16], JSMA [17]; optimization-based methods, such as L-BFGS [7], C&W attack [18], Deepfool [19] and GAN-based attack methods, etc. Moreover, the transferability of adversarial perturbations has also been concerned. The adversarial perturbations obtained in one model may also be effective for other models [20], which makes adversarial attacks more threatening. At the same time, in areas related to image classification, such as semantic segmentation [21], object detection [22] and video classification [23], adversarial attack methods have also been studied. Furthermore, attacks could not only be applied on the digital data but also could be carried out in the physical world [24,25].

In order to deal with the threat of adversarial attacks, on the one hand, researchers try to improve the model's resistance to these attacks, so that the model can make correct predictions on adversarial samples. This kind of methods can be divided into two subcategories: one is proactive defense, which aims to improve the performance of the model itself to obtain robustness against attacks; the other is passive defense, which aims to find ways to reduce or even eliminate adversarial perturbations, and can lead to correct predictions for adversarial examples without changing the model structure and parameters. On the other hand, they tried to avoid these attacks, making it difficult to input adversarial examples into the model. This type of methods can also be divided into two sub-categories: information masking and detection-based defense. For those algorithms that tried to enhance the robustness of the model against adversarial examples, defense methods can generally be divided into two main categories. One is the proactive defenses, which aim to improve the robustness of the model and make the model resistant to adversarial samples. Another is the passive defenses, which aim to find ways to reduce or even eliminate the adversarial perturbation instead of strengthening the

* Corresponding author.

https://doi.org/10.1016/j.neucom.2022.09.004
0925-2312/

model itself. Generally speaking, proactive defenses, such as adversarial training [14,15],neural network optimization [26,27], show good defensive effects against specified attacks, but requires large amount of computational power. Passive defenses, such as randomization [28,29], Denoising and input preprocessing [30,31], Generating benign images [32,33], can separate the training process of the classifier from the implementation of the defense method, and the well-trained model can still be used. But when the defense method is known, passive defenses may fail to defend against stronger attacks. In addition, the information masking based defense method defend against hostile attacks by preventing the leakage of key information. Detective defense is to find adversarial samples and reject abnormal predictions to achieve the defending goals.In addition, the information masking based defense methods prevent the leakage of key information, making some adversarial attack methods impossible to use, and avoiding the generation of adversarial samples. Detective defense is to find out and reject abnormal predictions through detection methods, so as to achieve the purpose of defense.

In this paper, we survey and summarize the recent theories, algorithms, and applications of adversarial attacks and defense methods in a broad and detailed manner, and give the analysis and comparison of the pros and cons of these attack and defense methods. Finally, we discuss the main challenges in this field and present the potential future research directions. We hope that this work will be further promoted for follow-up researchers. The rest of this paper is organized as follows. Section 2 introduces some important definitions and notations, Section 3 introduces the attack methods, Section 4 introduce the defense methods. In Section 5, we give some discussions, and Section 6 concludes this paper.

2. Definitions and notations

Generally speaking, the attacks and defense methods discussed in this section are mainly related to image classification deep learning models, but these methods can also be applied to other deep learning models. In this section, for the purpose of simpleness, we briefly introduce some general definitions and notations used in this paper.

2.1. Attack goals and performance assessment

2.1.1. Adversarial samples and benign samples

In the field of image recognition, unmodified images are generally called benign samples, while maliciously crafted attacking samples are called adversarial samples. Generally speaking, the victim models under discussion are trained on the benign images, which are supposed to make correct predictions on the benign images with high probability. The goal of malicious attacks is to slightly modify the benign samples to make the models output wrong prediction results on them.

2.1.2. Distance metrics

In order to meet the requirement of not being detected by humans, the adversarial sample needs to maintain a small distance from the benign sample. A distance matrix is used to describe this distance. The most commonly used distance matrix $D(\cdot)$ is the l_p norm. For a vector $x = (x_1, x_2, \ldots, x_n)$ in the n-dimensional real vector space R^n, and a real number $p \geqslant 1$, the p-norm or l_p-norm of x is defined as

$$||x||_p = \left(|x_1|^p + |x_2|^p + \ldots + |x_n|^p\right)^{\frac{1}{p}}. \quad (1)$$

Specifically, the l_2-norm is equivalent to the Euclidean distance, and the l_1-norm is the norm that corresponds to the Manhattan distance. In addition, there are two specials. The l_0 norm is the number of elements that are not zero. And the l_∞-norm or maximum norm is the limit of the l_p-norms for $p \to \infty$. Obviously, this norm can be equivalent to the following definition:

$$||x||_\infty = max\{|x_1| + |x_2| + \ldots + |x_n|\}. \quad (2)$$

It's notable that there are other methods to measure the distance between the adversarial samples and benign samples. These measurement methods will be introduced in their corresponding attacks and defense methods

2.1.3. Target attack and non-target attack

Given the classifier f and the benign data (x, y), where y is the ground truth label of x. And the adversarial perturbation added into x is denoted as δ. Target attack aims to mislead the classifier to classify the adversarial examples $x + \delta$ as a target label y'. If y' is not specified, but different from y, the attack is then called the non-target attack.

2.1.4. Adversarial robustness

The robustness of the model can be measured by its vulnerability to attack. Therefore, the robustness of the model can be defined by the minimum perturbation that can be attacked successfully. If we use the l_p norm to constrain the perturbations, then the robustness of the model for the data (x, y) is computed as following:

$$r(x, f) = \underset{\delta}{argmin}||\delta||_p \qquad s.t. \qquad f(x + \delta) \neq y \quad (3)$$

2.2. Information of the attacked models

2.2.1. White-box attack

For a white-box attack, the attacker can access all the information of the target neural network, including its architecture, parameters, gradients, etc. As a result, the attacker can perform the most powerful attacks on the model, and many effective defense methods against black-box attacks cannot resist these attacks.

2.2.2. Black-box attack

In the black-box attacks, the attacker does not know any information of the target network, and can only get the correlation between input and output by querying the model. These methods could be broadly divided into two classes. One is to use a additional network to approximate the relationship between the input and output of the target network, and then perform white box attacks on this shadow network. These attacks are expected to be effective on the target network. The second is to take advantage of the transferability of adversarial perturbations, and use transferable adversarial examples to attack.

2.2.3. Gray-box attack

There are two forms of gray-box attacks. One is that the attacker only has partial information about the target model, such as only knowing the model structure but having no knowledge of the model parameters. The other is to train a generative network for crafting adversarial examples through the target white-box model. To attack the target model, there is no need to know its additional information.

3. Adversarial attacks

Although the definition and constraints of adversarial attack are clear, it is inefficient to search adversarial examples randomly. Many researchers worked on designing more fast and effective attacking methods. Among all these algorithms, gradient-based

attacks try to obtain adversarial samples through gradient back-propagation, without any loss function nor optimizer, and usually have high generation speed. However, these methods could only be used for white-box attacks; Optimization-based attacks that utilize a loss function and an optimization process could achieve better attacking performance compared with gradient-based methods, but they usually have slower speed. For the generative-model-based methods, an auxiliary network is often used to generate adversarial examples. This method requires additional network training at a certain cost, and the attacks only work on some specified model. In addition, some other specific attack methods and practical applications of adversarial attack techniques are also introduced in this paper.

3.1. Attacks based on gradient

In this section, we introduce a series of methods and algorithms on how to craft adversarial examples, which are constructed based on gradients or derived from them. Due to the need of the knowledge of the structure and parameters of the model, these attacks typically belong to white-box attacks.

3.1.1. Fast gradient sign method (FGSM)

It's observed that, by injecting small but deliberate perturbations to benign data, the adversarial examples could be crafted, which makes machine learning models to output an incorrect answer with high confidence. Previous studies believed that the cause for this observation is due to the non-linearity and overfitting of the models, but Goodfellow et al. [14] pointed that the primary cause lies in the linear nature of the neural networks. In view of this, to increase the classification loss in maximum direction, they performed the one-step update along the direction of the gradient of the cost

$$J(\theta, x, y). \tag{4}$$

The formulation of constrained perturbation is:

$$\eta = \epsilon sign(\nabla J(\theta, x, y)). \tag{5}$$

Therefore, the adversarial examples of an non-targeted attacks can be generated as:

$$x' = x + \epsilon \cdot sign(\nabla J(\theta, x, y)). \tag{6}$$

Later, [34] improved it by changing the goal of increasing the loss of the original label to reduce the loss of the target label:

$$x' = x - \epsilon \cdot sign(\nabla J(\theta, x, y')), \tag{7}$$

where y is the correct label of the benign image x, and y' is the least likely class label. This method is called the "fast gradient sign method (FGSM)".

3.1.2. Basic iterative attack and projected gradient descent

FGSM is a typical attack algorithm which could efficiently craft adversarial examples. To improve the performance, Kurakin et al. [35] presented Basic iterative attack (BIM), which used multiple iterations instead of individual iteration relative to FGSM. Each iteration updates slightly, and clips the adversarial examples into a valid range. The iterative formulation is presented as following:

$$x'_{t+1} = Clip\{x'_t + \alpha \cdot sign(\nabla J(\theta, x', y))\}. \tag{8}$$

The constraint of perturbations

$$\epsilon = \alpha T \tag{9}$$

is obtained by T iterations, and α is the magnitude of perturbation in each step. Similar to the FGSM, [35] extended BIM to Iterative Least-likely Class Method (ILCM) by using the least likely class.

Madry et al. [15] introduced the projected gradient descent (PGD) as a variant of BIM. It is initialized with uniform random noise and iterates more times. Moreover, instead of clipping simply, PGD projects the adversarial examples into

$$\epsilon - l_\infty \tag{10}$$

neighbor of the benign image. The iterative formulation of PGD is defined as following:

$$x'_{t+1} = Proj\{x'_t + \alpha \cdot sign(\nabla J(\theta, x', y))\}. \tag{11}$$

Both BIM and PGD use multi-step iteration to improve FGSM, which are called iterative FGSM (I-FGSM). The purpose of I-FGSM is to find the strongest adversarial examples in the vicinity of the benign examples (Limited by),

$$l_p \tag{12}$$

so as to maximize the classification loss. Such attack methods are very aggressive to the target neural networks. Compared with FGSM, although the BIM-based attacks take more time, it showed stronger attacking performance with the same size of disturbance.

3.1.3. Momentum iterative attack

Although I-FGSM has a high success rate in white-box attack setting, it doesn't perform well when being transfered and it is not conducive to attack other black-box models. To further enhance the perfromance of FGSM, momentum iterative FGSM (MI-FGSM), presented by Dong et al. [16] applied momentum to improve the attack strength. The iteration step was changed to:

$$x'_{t+1} = x'_t + \alpha \cdot sign(g_{t+1}). \tag{13}$$

And g_t is updated by momentum as:

$$g_{t+1} = \mu \cdot g_{t+1} + \frac{\nabla J_t(\theta, x', y)}{||\nabla J_t(\theta, x', y)||_1}. \tag{14}$$

In order to further increase the success rate of the black-box attacks, the authors designed a scheme that could attack multiple white-box models at the same time, which improved the transferability of the adversarial examples. Specifically, an ensemble model is obtained by weighting the non-normalized probability values of several individual models. The ensemble model is then attacked with the MI-FGSM attack.

Xie et al. [36] further transformed the image in each iteration with a probability of p. Then, the DI2-FGSM and M-DI2-FGSM, based on I-FGSM and MI-FGSM, were proposed in this way.

3.1.4. Deepfool

FGSM searches for the strongest adversarial examples within a given perturbation range (usually constrained by l_p norm). In contrast, Moosavi-Dezfooli et al. [19] introduced Deepfool to find the smallest perturbation (l_2 norm) that can cause misclassifications. In their paper, a new classifier robustness evaluation index is proposed. For a classifier

$$\hat{k}, \tag{15}$$

the distance from x to decision boundary is designed as the following equation:

$$\triangle\left(x; \hat{k}\right) := min_r \|r\| \quad subject \quad to \quad \hat{k}(x + r) \neq \hat{k}(x), \tag{16}$$

and the robustness of classifier $\hat{k}$ is then defined as:

$$\rho_{adv}\left(\hat{k}\right) = \mathbb{E}_x \frac{\triangle\left(x; \hat{k}\right)}{\|x\|_2}, \tag{17}$$

where $\mathbb{E}_x$ is the expectation over the distribution of input. The farther the example is from the decision boundary, the smaller the

example's l_2 norm is, and the larger the evaluation value, the more robust the classifier is.

DeepFool transforms the goal of finding the minimum perturbations that could mislead the classification into finding the nearest decision boundary. For multi-classification tasks,

$$\hat{k}(x) = \underset{k}{argmax} f_k(x) \tag{18}$$

, and $f_k(x)$ is a sub-classifier of the k-th category. In order to change the classification results, it must be ensured that at least one classifier has a higher score than the target one. The k-th classification boundary is

$$\mathscr{F}_k = \left\{ x : f_{\hat{k}(x_0)}(x) - f_k(x) = 0 \right. \tag{19}$$

}. The next step is to find the shortest path that can go beyond the decision boundary. As shown in Fig. 1, by calculating the orthogonal vector to plane, the perturbation vector with the minimum distance can be obtained. By moving along the vector, the adversarial example can be found. For similar attacking performance, the perturbations crafted by DeepFool are much smaller than these of FGSM.

3.1.5. Jacobian-based saliency map attack (JSMA)

The Saliency Map was used to calculate the grads of the input features through the probability of the class being predicted by the neural networks, so as to determine which input features have the greatest impact on the output category. Jacobian-based saliency map attack (JSMA) [17], presented by Papernol et al., used the effect of input perturbations on the output to craft adversarial examples. As a targeted attack method, the authors proposed to directly increase the predicted value of the neural network for the target category. The perturbation direction of the adversarial sample generated by this method is the gradient direction of the predicted value of the target category label, and this gradient is called the forward derivative:

$$\nabla \mathrm{F(X)} = \frac{\partial \mathrm{F(X)}}{\partial \mathrm{X}} = \left[\frac{\partial \mathrm{F_j(X)}}{\partial \mathrm{x_i}} \right]_{\mathrm{i}\in 1\cdots \mathrm{M}, \mathrm{j}\in 1\cdots \mathrm{N}} \tag{20}$$

By examining the impact of each pixel intensity to the output, at each iteration, the perturbed pixels selected to be modified are able to increase the output value of the target category as much as possible, and generally decrease the output value of the remaining category labels.

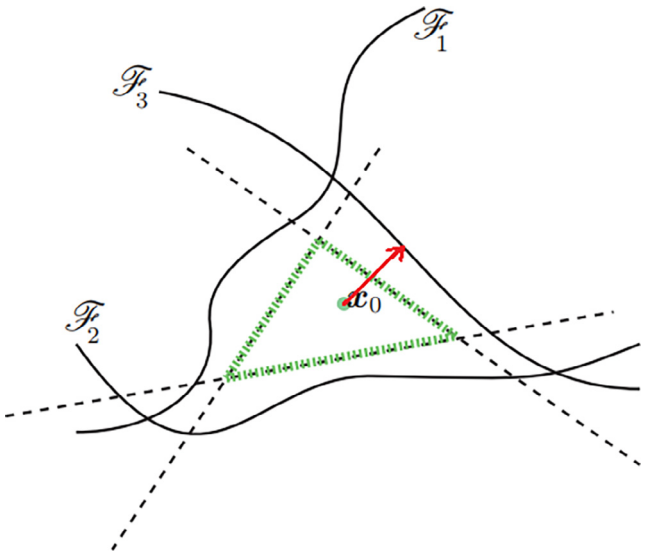

Fig. 1. $\mathscr{F}_3$ is the nearest classification boundary [19].

Unlike FGSM, JSMA uses l_0 to constrain the perturbations, which means that there is no limitation on the size of pixel intensity changes, but there is a limit to the numbers of pixels. Experiments show that adversarial samples can be crafted reliably with a 97.10% success rate by modifying samples by averagely 4.02% and modifying pixels less than 14.29% will not affect the correct classification of humans.

3.1.6. Decoupling direction and norm attack

Generally, the optimization method of finding adversarial examples is to consider distance metric and adversarial loss together to minimize a hybrid loss. For example, the CW_2 attack's optimization objective is:

$$min_{\delta} D(x, x+\delta) + c \cdot f(x+\delta) \tag{21}$$

, where the optimal c needs to be meticulously designed. Rony et al. [37] proposed a approach to craft gradient-based attacks by decoupling the direction and the norm of the adversarial perturbation. Specifically, it projects adversarial perturbations into the $\epsilon - sphere$ of the benign image x to control the norm instead of imposing a penalty. For a non-targeted attack, in each step, While doing the gradient iteration, this algorithm changes the norm by determining whether the current update is an adversarial example. Then it reduces the norm if it is adversarial, and increases the norm if it's not,

3.1.7. Attacks against obfuscated gradients defense

Due to the success of gradient-based attack methods, keeping gradients uncovered is a feasible class of targeted defenses. Athalye et al. [38] summarized these methods as obfuscated gradients, and divided them into three categories: shattered gradients, stochastic gradients, and Exploding & vanishing gradients. The authors found that the defense method of obfuscated gradients may not be able to completely defend against the adversarial attack, and designed three attack methods to deal with three different types of obfuscated gradients.

Shattered gradients mean nonexistent or incorrect gradients. The authors introduced Backward Pass Differentiable Approximation (BPDA) to attack this defense method. Specifically, in backpropagation, differentiable modules are used to approximate the non-differentiable modules in the defense model. For most non-differentiable defense models, their secured classifier is $\hat{f}(x) = f(g(x))$, where $f(\cdot)$ is a pre-trained classifier, and $g(x) \approx x$ is a pre-processor, which tranforms an adversarial image into a benign image. If $g(\cdot)$ is not differentiable. Let $\nabla_x g(x) \approx \nabla_x x = 1$, and the derivation of $f(g(x))$ can be approximated as

$$\nabla_x f(g(x))|_{x=\hat{x}} \approx \nabla_x f(x)|_{x=\hat{x}}. \tag{22}$$

For the more general case, let $f^i(\cdot)$ be a non-differentiable layer, which is a part of a neural network $f(\cdot) = f^{1..j}(\cdot)$. A differentiable approximation $g(x)$ satisfied $g(x) \approx f^i(x)$, is used to replace $f^i(x)$ in the backpropagation.

For Stochastic gradients of random transformations, Expectation over Transformation (EOT) [39] is used to calculate the expected gradients, which are assumed to be used to perform gradient-based attacks on the classifier.

Finally, vanishing/exploding gradients can be solved by reparameterization. A classifier $f(g(x))$, when $g(\cdot)$ is the transformation step, may cause the vanishing/exploding gradients. We can compute gradients and circumvent the defense by transforming the input x with $x = h(z)$, where $h(\cdot)$ is differentiable, and for all z, there are $g(h(z)) = h(z)$.

In the author's study, for 7 white-box-secure defenses relating on obfuscated grades at ICLR2018, those attack methods successfully circumcised 6 completely, and 1 partially.

3.2. Attacks based on optimization

3.2.1. L-BFGS

Szeged et al. [7] firstly found adversarial examples in deep neural network image classifiers. In order to find the minimum perturbations that changes the output of the classifier, authors proposed the target as follows:

$$\min_{x'} ||x - x'||_p \qquad s.t. \qquad f(x') \neq y'. \tag{23}$$

where y' is the incorrect label of benign image x, and

$$||x - x'||_p \tag{24}$$

is the

$$\ell_p \tag{25}$$

norm of the adversarial perturbations. For the convenience of optimization [40], the loss function is designed as:

$$\min_{x'} c||x - x'||_p + J(\theta, x', y'), \tag{26}$$

where

$$J(\theta, x', y') \tag{27}$$

is the loss function of the classifier for the adversarial example and the incorrect label, and c is a changing hyper parameter.

3.2.2. Carlini and Wagner attack

Similar to L-BFGS, the C&W attack method proposed by Szeged et al. [18] is also an optimization-based method. The perturbations added to the benign examples generated by C&W attack are almost imperceptible. Compared to FGSM and PGD, the perturbations crafted by C&W are smaller and the attack effect is more powerful. For the classification model with distillation defense, C&W attack can still attack successfully efficiently. As shown as follows, the authors put forward a new optimal formula:

$$\min_{\delta} \mathscr{D}(x, x+\delta) \quad s.t. \quad C(x+\delta) = t \quad x + \delta \in [0,1]^n. \tag{28}$$

where the C is the classifier, and t is the desired target label. However, the equation constraint is difficult to derive, so the authors transformed this equation and defined an objective function f such that $C(x+\delta) = t$ if and only if $f(x+\delta) \leqslant 0$. The alternative formulation could be written as:

$$\min_{\delta} D(x, x+\delta) + c \cdot f(x+\delta) \quad s.t. \quad x + \delta \in [0,1]^n. \tag{29}$$

By minimizing this loss, it can find the adversarial input $x + \delta$ that causes the misclassification. Next, in order to solve the box constraint problem ($x + \delta \in [0,1]^n$), the authors proposed three different approaches to this problem. Projection gradient descent performs one step of standard gradient descent, and then clips the calculation results to box. Clipped gradient descent directly adds constraints to the objective function, that is:

$$f(x+\delta) \Rightarrow f(min(max(x+\delta, 0), 1)).$$

Change of variables makes $x_i + \delta_i$ satisfy the box constraint by introducing variables w_i, and the method is:

$$\delta_i = \frac{1}{2}(tanh(w_i) + 1) - x_i,$$

Experimental results showed that Projection gradient descent is better than the other two strategies in dealing with box constraints, but the perturbation sought by Change of Variable is generally smaller. The distance $D(x, x+\delta)$ between benign examples and adversarial examples is constrained by the l_p norm. Corresponding to l_0, l_2 and l_∞ norm, C&W attacks can be divided into CW_0, CW_2 and CW_∞

Compared to L-BFGS, there are two main advantages. First, the authors defines a series of loss functions f to replace the constraint $C(x+\delta) = t$. And the cross-entropy loss of L-BFGS is replaced by a so-called margin loss, which is used to measure whether misclassification happened. Second, the authors used the tanh function to expand the optimization range from [0,1] to [-inf, inf], which is conducive to optimization.

3.2.3. Elastic-net attacks to deep neural networks

Chen et al. [41] proposed some new improvements based on the Carlini and Wagner attack, which enhanced the transferability of the attack while ensuring the success rate of the attack. On the basis of CW_2 attack, different from only using the l_2 norm regularization term, the authors proposed to add the elastic network regularization term, *i.e.*, to use the l_1 and l_2 norm regularization term at the same time. The following optimization equation is obtained:

$$\min_{\delta} c \cdot f(x+\delta) + \beta||\delta||_1 + ||\delta||_2 \quad s.t. \quad x + \delta \in [0,1]^n, \tag{30}$$

where c and β are hyper parameters.

Experimental results showed that, compared with CW_2, the perturbations including the l_1 penalty do produce a unique set of adversarial examples, which leads to an increase in attack transferability.

3.2.4. Universal adversarial perturbations

[20] firstly tried to find universal adversarial perturbation (UAP). Compared with the adversarial perturbations generated by the above mentioned attack methods, UAP can be applied to a batch of benign images. UAP is obtained by iterative optimization of a set of benign images. Iterative optimization is performed on each benign image. If the current perturbation cannot fool the classifier, then the method would find a minimum additional adversarial perturbation and add it to the original perturbation. The UAP crafted in one model could work on another model, which makes UAP have certain black-box attack capabilities.

Moreover, Mopuri et al. [42] generated universal adversarial perturbations by fooling the features learned at multiple layers in the absence of data. In their works, Mopuri et al. [43] further developed UAP and proposed a new method of crafting UAP called generalizable data-free universal adversarial perturbations (GD-UAP). GD-UAP is data-free and can take effect among different data sets and tasks, not limited to image classification tasks. The authors verified their methods in three tasks: image recognition, semantic segmentation and depth estimation. In another work [44], the authors achieved data-free adversarial sample generation by simulating data samples with class impressions.

There are also some other schemes to generate UAPs. For example, Hayes et al. [45] introduced universal adversarial networks (UAN) to generate UAPs on current image datasets. Mopuri et al. [46] trained a UAP generator using the GAN architecture. Khrulkov et al. [47] proposed a method of constructing UAPs based on the singular vector of the Jacobian matrix of DNN. This method allowed them to achieve a high fooling rate on a dataset consisting of 50,000 images, using only 64 images. Sarkar et al. [48] had studied two perturbation generation algorithms at the same time. The first is Universal Perturbations for Steering to Exact Targets (UPSET) for the specified category, and the second is Antagonistic Network for Generating Rogue Images (ANGRI) for the specified image.

3.3. Attacks based on generative model

The basic idea of the method of generative model is to train a fixed network, and any benign image as input can get its corresponding adversarial sample through this network. With the suc-

cess of generative adversarial networks (GAN), especially conditional GANs, in generating high-quality images, the method of directly generating adversarial examples through a pre-GAN has also been developed.

3.3.1. Adversarial Transformation Network (ATN)

Baluja et al. [49] proposed a network named Adversarial Transformation Network (ATN) to train a generation model to generate adversarial samples. This model takes benign samples as input and generates corresponding adversarial samples with minimal modification. ATN can be defined as:

$$g_{f,\theta}(\mathrm{x}) : \mathrm{x} \in \chi \rightarrow \mathrm{x}',$$

where θ is the parameter of the attacked classification network, and g is the attack network we want to generate. The optimization problem is described as:

$$\underset{\theta}{arg\min} \sum_{\mathrm{X_i} \in \chi} \beta L_{\chi}\left(g_{f,\theta}(\mathrm{X_i}), \mathrm{X_i}\right) + L_Y\left(f\left(g_{f,\theta}(\mathrm{X_i})\right), \left(f(\mathrm{X_i})\right)\right). \tag{31}$$

where L_{χ} is the visual loss, which is the L_2 loss in this paper, L_Y is the classification loss, and β is the weight coefficient that balances these two loss functions. L_Y is defined as:

$$L_{Y,t} = L_2(y', r(y,t)). \tag{32}$$

where $r(y,t)$ is the reranking formula. Its function is to modify the score of each category of the output. Specifically, it keeps the order of the classification confidence of the other categories unchanged, and only increases the confidence of the target category of the attack to the maximum. The advantage of this is that to a certain extent, the adversarial samples are closer to the original samples, because the prediction results of Top-2 are the true labels.

ATN uses two schemes to generate adversarial examples: Perturbation ATN (P-ATN) and Adversarial Autoencoding (AAE). Perturbation ATN only generates perturbations, and Adversarial Autoencoding directly generates adversarial examples. The AAE method can reduce the overall loss to the greatest extent, but the overall change of the generated image is relatively obvious, while the overall change of the P-ATN method image is smaller, but it will produce larger disturbances at the edges or corners of the image. In addition, it is found in experiments that ATN trained by a single network has no generalization ability, while ATN trained by multiple networks has better generalization ability.

Although ATN takes a lot of time to train, a trained ATN network can generate adversarial samples at a very small cost, which is faster than the optimization-based method.

Similarly, Hayes et al. [50] also Use another attacker neural network to learn to craft black-box adversarial examples. Experiments show that their method can reduce accuracy of the black-box neural network from 99.4% to 0.77% on the MNIST [51] dataset, and from 91.4% to 6.8% on the cifar-10 dataset [52], and can transfer to other machine learning models such as Random Forest, SVM, and K-Nearest Neighbor. Different from methods which create a one-time perturbation through a deterministic generator, Jang et al. [53] proposed a recursive and random generator that can generate more powerful and diverse perturbations and fully reveal the vulnerability of the target classifier.

3.3.2. GAN-based adversarial attack

Zhao et al. [54] combined the idea of Generative Adversarial Networks (GAN) into the generation of adversarial samples and named the method Natural GAN. The specific method is to first train a WGAN [55] model, where the generator G maps random noise to the input domain. A transformer is then trained to map the input data to a dense internal representation. This method generates adversarial noise by minimizing the distance of the victim network's internal representation.

Xiao et al. [56] proposed the AdvGAN method, which can perform black-box attacks without relying on the transferability of adversarial examples. AdvGAN uses a generator $\mathscr{G}$ to craft adversarial perturbations, a discriminator $\mathscr{D}$ to distinguish the perturbations data $X + \mathscr{G}(X)$ from the source data X, and a classifier f to be the targeted attacked network. The final optimization objective consists of adversarial loss $\mathscr{L}_{GAN}$, misclassification loss $\mathscr{L}_{adv}^{f}$ and soft hinge loss on the l_2 norm $\mathscr{L}_{hinge}$. $\mathscr{G}$ and $\mathscr{D}$ are obtained by solving the minmax game $arg\ min_{\mathscr{G}} max_{\mathscr{D}} \mathscr{L}$, where $\mathscr{L} = \mathscr{L}_{adv}^{f} + \alpha \mathscr{L}_{GAN} + \beta \mathscr{L}_{hinge}$, and α and β are the hyperparameters. For black-box attack, the authors used the distilled network f to replace the black-box model for optimization. Further, they proposed a substitute minimization approach and train the distilled model f and the generator $\mathscr{G}$ jointly.

Odena et al. [57] proposed a variant of the GAN architecture called Auxiliary Classifier GANs(AC-GAN). In short, AC-GAN adds an auxiliary classifier to the output part of $\mathscr{D}$ to improve the performance of conditional GAN. Song et al. [58] trained an AC-GAN to model the class-conditional distribution over data samples. Then, given a class, it searches over the AC-GAN latent space to find adversarial examples, which is classified as another class. The adversarial examples crafted in this way are not based on any existing benign examples. They are proposed as unrestricted adversarial examples. These unrestricted adversarial examples can bypass some strong adversarial training and certified defense methods.

3.4. Attacks by miscellaneous methods

Sabour et al. [59] proposed to manipulate the image representation in a deep neural network (DNN) to generate adversarial examples. Specifically, given a source image, a guide image, and a trained DNN. By adding a small perturbation to make the source image have an internal representation similar to the guide image, the adversarial image thus generated has the same visual effect as the original image, but the classification results are consistent with the guide image. Dong et al. [60] proposed a translation-invariant attack method to generate more transferable adversarial examples for the defense model. By optimizing the perturbation to a set of translated images, the generated adversarial samples become less sensitive to the white box model being attacked and have better transferability.

3.4.1. Decision-based adversarial attacks

In a real attack scenario, the victim usually does not release the output score of the model, but directly gives the decision. Brendel et al. [61] proposed a boundary-based attack method. This method does not require the gradient information and score information of the model. Only the decision results of the model are needed to perform white-box and black-box attacks. It is still effective against the defense method of passing hidden gradients. The specific method is to select an adversarial sample as the initial point for a benign image (regardless of the size of the perturbation to ensure adversarial performance). Then perform a random walk, while making the picture still an adversarial sample, and the distance from the original picture cannot be too large.

3.4.2. Search-based

Using search-based algorithms to find adversarial examples does not require the internal information of the victim network and can often be used for black-box attacks. Andriushchenko et al. [62] proposed a score-based black-box Square Attack. The model does not rely on the local gradient information of the model,

so it can bypass the gradient hiding defense attack. Square Attack is a random search method. In each iteration, a square location is randomly selected, and a random update perturbation is sampled. If the perturbation improves the objective function, it is updated, so that the perturbation is approximately at the boundary of the feasible set in each iteration.

Narodytska et al. [63] proposed a black-box attack method named LOC-SEARCHADV (LSA) that modifies only a few pixels and only needs to observe the output of the network on the probed input. LSA utilizes a local-search based technique to construct a numerical approximation of the network gradient. In each round, local neighborhoods are used to optimize the current image, and in the process optimize some objective function that depends on the network output. Although LSA is more computationally expensive, but compared to FGSM perturbing all pixels, it can modify only a small fraction of pixels, and the average perturbation is much less.

3.4.3. l_0 attack

The adversarial attack algorithms mentioned earlier all make tiny perturbations on all pixels of the entire image to fool the model. The method proposed by Su et al. [64] only changes a small number of pixels, and even in the extreme case of only one pixel, to obtained a good attack effect. This method named One-pixel attack is a gray-box attack (only need probabilistic labels) algorithm based on differential evolution (DE) to craft one-pixel adversarial samples. If the image is viewed as a long vector, the authors argue that single-pixel modification can be seen as perturbing the data points along an axis parallel to one of the n dimensions. Thus a single-pixel perturbation can modify an image in a chosen direction with any intensity from n possible directions. In a one-pixel attack, only the number of modified pixels is limited, but no modification strength is limited. Differential evolution (DE) [65] is a population-based optimization algorithm that can solve complex multi-modal optimization problems without the need for gradients [66]. In a one-pixel attack DE is used to find the key pixel that needs to be modified. Grafting adversarial images using DE requires less information from the target network and is more likely to find globally optimal pixels.

3.4.4. Non-l_p-measured attack

Using l_p distance to measure the similarity between the original image and the adversarial example is sometimes inconsistent with human visual senses. Brown et al. [67] presented Adversarial Patch attacks. These patches are added to the benign samples, and the classifier can output target labels in any scene. Further, Liu et al. [68] proposed a perceptual-sensitive generative adversarial network (PS-GAN) to generate adversarial patches. This method improves visual fidelity through patch-to-patch translation and enhances the attacking ability by using attention mechanism. Furthermore, Liu et al. [69] applied the adversarial patch attack to the object detection field, and they proposed DPatch to attack the bounding box regression and object classification simultaneously. Thys et al. [70] successfully attacked the object detector by printing the adversarial patch on a cardboard and hanging in front of the person. Xu et al. [71] printed the adversarial patch on a white T-shirt, and studied the performance of the adversarial sample under deformation and movement.

Xiao et al. [72] first proposed a type of adversarial patch called spatial transformation, which generates adversarial samples by shifting some pixels. This kind of adversarial sample is still a real image from the perspective of human perception, even though it has a large l_p distance from the original image.

Zhao et al. [73] found that perceptual color distance can improve imperceptibility, especially in smooth and saturated areas. They proposed two methods (PerC–C&W and PerC-AL) for creating adversarial images, which have lager RGB Lp norms than methods perturbing directly in RGB space. Shamsabadi et al. [74] proposed a content-based black-box adversarial attack method (ColorFool), which used image semantics to selectively modify colors in a selected range that humans consider natural, thereby generating unrestricted perturbations.

Rozsa et al. [75] proposed a Hot/Cold method, which defines a new metric perceptual adversarial similarity score (PASS) to measure the difference between samples before and after attack, and can generate multiple adversarial samples for each input image. PASS is performed in two steps, first aligning the modified image with the original image, and then measuring the similarity between the aligned modified image and the original image. In the Hot/Cold method, Hot and Cold represent the target class and the original class, respectively, and after each iteration, the algorithm moves the samples towards the Hot class and away from the Cold class. Compared with FGSM, the algorithm generates a diversity of adversarial samples.

3.4.5. Black-box attacks by approximate gradient

The gradient of the black-box model cannot be obtained directly, and approximating it with a white-box model is a viable approach [76]. The success of this kind of attack methods is due to the transferability of adversarial perturbations, which requires that the approximation model and the black-box model do the same tasks and have the same distribution of the input data. Chen et al. [77] proposed zeroth order optimization (ZOO) based attacks. This algorithm is a typical black-box attack algorithm and can be directly deployed in black-box attacks without model transfer. By using the gradient estimation of the Hessian matrix, there is no need to obtain the gradient information of the target model. However, it requires expensive computation to query and estimate gradients. Experiments show that the ZOO attack achieves comparable performance to the C&W attack.

The other kind of method to approximate gradient is query-based, proposed by Bhagoji et al. [78]. The gradient of the target model is estimated by querying the class probability of the target black-box model. These attacks, which require query access to the target model's class probabilities, but do not rely on transferability, can complete target attack and non-target category attack at the same time.

3.4.6. Transfer-based Attacks

The transferability of perturbations provides a feasible solution for black-box attacks. Papernot et al. [79] introduced the concept of portability of adversarial examples. It is proposed that adversarial examples that mislead a classification model can also mislead another classification model with a very different structure. Attackers can generate adversarial samples by training their own substitute models, and these samples can work on the target model. They further greatly enhanced the effect of training substitute models through their reservoir sampling method. Shi et al. [80] proposed a black-box attack method called Curls iteration, which can generate more transferable adversarial samples, and they also successfully applied Curl&Whey attack to black-box targeted attacks.

Liu et al. [81] deeply studied the transferability of adversarial perturbations between models, and proposed an ensemble based approach to generate adversarial examples, which is effective for another black-box model. Cheng et al. [82] proposed a prior-guided random gradient-free (P-RGF) method to enhance black-box adversarial attacks. Compared with methods that using surrogate models or query feedback to approximate the gradient of the black-box model, P-RGF established a gradient estimation framework, and used the transfer-based prior, given by the gradient of

a surrogate white-box model, to do query-efficient black-box attacks.

Wu et al. [83] found that prioritizing the key features of the attack concerned by various models can improve the transferability of the adversarial perturbations. In light of this discovery, it computed model attention over extracted features, and searched for adversarial examples, which can be corrupted by the critical features.

3.4.7. Data poisoning

Shafahi et al. [84] adopted a method called "data poisoning attacks" to attack the neural network. Compared with data pollution attacks with incorrect labels, data poisoning attacks use fine-tuned adversarial examples with correct labels. These fine-tuned examples have no effect on the general training and performance of the neural network. But for a specific sample (the sample that the attacker is about to attack), the attacked model would give wrong results. Specifically, it would classify it into the category selected by the attacker.

3.5. Attacks on applications

3.5.1. Attacks in physical world

The previously mentioned adversarial examples all need to add adversarial perturbations to the input image. However, in the physical world, there are always various situations that make the images obtained with natural noises. These natural noises may destroy the adversarial perturbations and fail the attacks. Nevertheless, attacks in the physical world still exist. As show in Fig. 2, Eykholt et al. [24] attached stickers to road signs and successfully fooled the sign recognizer. Their attack method is to first find the vulnerable region, then generate the adversarial examples on this region, and finally print it out and paste it on the road sign. (See Fig. 3, Fig. 4).

Sharif et al. [25] verified the feasibility of physical world attacks on face recognition tasks. Authors found that perturbations in a restricted region of the benign samples can also fool deep learning models. These perturbations are called adversarial patches. Further, the authors added the adversarial patches into 3D printed sunglasses frames. As shown in Fig. 1, the first column is the non-target attack, and in the remaining, the first row is an adversarial image with 3D printed sunglasses frames, which is the same as the face recognition result in the second row. In order to make the adversarial patches effective in the physical world, in the process of optimization generation, authors proposed to maintain the smoothness of perturbations by minimizing total variation (TV), and ensure that the image can be printed by minimizing non-printability score (NPS).

In addition, [85] proposed a method to generate universal 3D adversarial objects to deceive LiDAR detectors. Attackers can print out the 3D mesh and make any vehicle "invisible" without any prior knowledge of the scene by placing the mesh on it. [86] proposed a new method called AdvCam, which combines neural style transfer and adversarial attack techniques to make and disguise adversarial samples into a natural looking style. (See Table 1).

Fig. 2. Adversarial attack on road sign [24].

3.5.2. Attacks on semantic segmentation and object detection

Image classification tasks usually output only one category label for one input, while in semantic segmentation [87] and object detection tasks, the output is usually the relationships between an input and a series of categories. Semantic segmentation and object detection can be seen as an extension of image classification. In this way, adversarial examples can also appear in semantic segmentation and object detection [21,88,89,22]. (See Table 2).

Xie et al. [21] proposed the Dense Adversary Generation (DAG) algorithm to generate adversarial examples. DAG considers each classification target by assigning an adversarial label to it, and executes the generation of adversarial examples for each target. It is noteworthy that DAG considers all the targets simultaneously and optimizes the overall loss. The perturbations generated by DAG can be transferred on different training sets, different network architectures and even different tasks. DAG obtains more effective performance in black-box attacks by combining heterogeneous perturbations. Li et al.[90] designed a loss function combining label loss and novel shape loss, and attacked against Region Proposal Network (RPN). Their proposed method is effective for 6 object detectors and 2 instance segmentation algorithms on the MS COCO 2014 dataset.

Some researchers apply the adversarial attack of object detection to the physical world. Lu et al.[88] generated adversarial examples for stop sign detection and face detection, and produced adversarial physical objects. Chen et al.[89] also implemented physical attacks against Faster-RCNN and adopted the idea of expectation-overtransformation, making these attacks still effective under various transformations.

For semantic image segmentation, Hendrik et al. [22] studied the universal attack, and generated two types of universal adversarial perturbations. By applying these two perturbations to benign image, they implemented the static target segmentation, where the semantic segmentation results had nothing to do with the input, and the dynamic target segmentation, which only removed the segmentation results of some objects, and kept others unchanged.

3.5.3. Attacks on video classifiers

Different from static images, video is a kind of temporarily varying inputs. Whether there are adversarial examples on video classification systems needs further verification. Li et al. [23] believed that temporary structure is the key to generate adversarial examples, and used generative adversarial network (GAN) architectures to generate adversarial samples on video classification systems. And the authors found that by applying the same perturbations to each frame in the video can also mislead the model to output wrong results.

4. Adversarial defense

Adversarial defense methods aim to prevent malicious attacks from misleading the models. The goal could be achieved by either improving the robustness of the model or destroying the attacking processes. The former could be further divided into proactive defense algorithms (Section 4.1) and and passive defense algorithms (Section 4.2). The difference between these two categories is that proactive defense aims to improve the performance of the model itself, while passive defense aims to mitigate or eliminate

Fig. 3. 3D printed sunglasses frames [25].

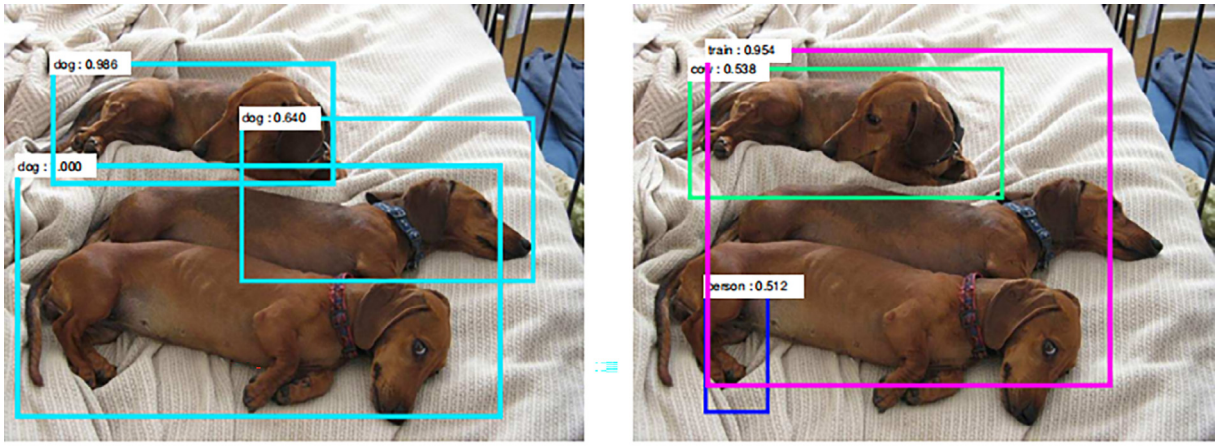

Fig. 4. An adversarial example for object detection. [21].

Table 1
Summary of adversarial attack.

attack	Algorithm	Distance metrics	Victim Information	Apply domain
FGSM[14]	Attacks based on gradient	l_2, l_∞	White-box	Image classification
BIM[35]&PGD[15]	Attacks based on gradient	l_∞	White-box	Image classification
MI-FGSM[16]	Attacks based on gradient	l_2, l_∞	Black-box	Image classification
Deepfool[19]	Attacks based on gradient	l_2	White-box	Image classification
JSMA[17]	Attacks based on gradient	l_0	White-box	Image classification
BPDA[38]	Attacks based on gradient	l_p	White-box	Image classification
L-BFGS[7]	Attacks based on optimization	l_2	White-box	Image classification
C&W[18]	Attacks based on optimization	l_p	White-box	Image classification
Elastic-net attacks[41]	Attacks based on optimization	l_2	White-box	Image classification
UAP[20]	Attacks based on optimization	l_2, l_∞	Black-box	Image classification
ATN[49]	Attacks based on generative model	l_2	Black-box	Image classification
AdvGAN[56]	Attacks based on based on generative model	l_2	Black-box	Image classification
AC-GAN[57]	Attacks based on generative model	l_p	Black-box	Image classification
one-pixel	differential evolution	l_0	White-box	Image classification
Eykholt[24]	Attacks based on gradient	Physical world	White-box	road sign recognizer
DAG[21]	Attacks based on gradient	l_∞	White-box	Object detection

adversarial perturbations. Defense algorithms that try to defend by destroying these attacks are introduced in Section 4.3 and 4.4, including information masking algorithms and detection-based defense algorithms.

4.1. Proactive defense: enhanced deep learning model

proactive defense schemes are aimed to strengthen the model to obtain better robustness. These proactive defense methods enable the model give correct predictions to adversarial samples. Typically, proactive defense requires additional training of the model, or additional optimization on model parameters and structure, which means that it will be different from the original model. In this section, we concentrate on the representative proactive defenses, mainly including adversarial training, optimize neural network and so on.

4.1.1. Adversarial training

To put it simply, adversarial training is to treat adversarial samples as additional inputs during the training process. The adversar-

Table 2
Summary of adversarial defense.

Category	Methods	Reference
Proactive defense	Adversarial training	[14,91,15,92–95,97,98,96,99–102,104,105,94]
	Neural network optimization	[7,110,106,111,26,107,112–114,108,109,115,116]
	Bayesian model-based defense	[117–119]
	Defense by ensemble learning	[120–124]
	Randomization	[28,29,125–130]
	Denoising and input preprocessing	[30,131,30,132,31,133,133–135,133,136–141]
	Generate benign Images	[32,33,142–144,148,149,151,152]
Information masking	Defensive distillation	[154]
	Gradient masking/obfuscation	[155,27]
	Boundary masking	[156]
Detection-based defense	Detection on input	[157–161]
	Detection on feature	[162,163]
	Detecting by auxiliary model	[164,165,167,168,166]
	Detecting by analyzing DNN's performance	[169–172,161]
	Detecting by consistency	[174,175,132,161]

ial samples are generated by the attack methods mentioned above, and they are marked with the correct labels. Through such training, it's respected that the data distribution learned by the model can contain these adversarial examples. The basic adversarial training steps are described as follows:

(1) Given a training set $\mathscr{X} = \{(x^1, \hat{y}^1), (x^2, \hat{y}^2), \ldots, (x^N, \hat{y}^N)\}$, and a model f to be trained. Firstly we use the $\mathscr{X}$ to train the model f.
(2) For each benign image x^n, generate its corresponding adversarial examples $\tilde{x}^n$ by an attack algorithm, and then create a new training set $\mathscr{X}' = \{(\tilde{x}^1, \hat{y}^1), (\tilde{x}^2, \hat{y}^2), \ldots, (\tilde{x}^N, \hat{y}^N)\}$
(3) Using both $\mathscr{X}$ and $\mathscr{X}'$ to update the model f.

Steps 2 and 3 can be repeated multiple iterations. Adversarial training can also be seen as a special data augmentation method. From the perspective of optimization, adversarial training can also be regarded as a minmax optimization problem. The optimization formula is presented as follows:

$$\min_{\delta} \mathbb{E}_{(x,y)\sim\mathscr{D}} \left[\max_{r_{adv}} L(\theta, x + r_{adv}, y) \right] \tag{33}$$

where $\mathscr{D}$ is the training set of benign image x and label y, r_{adv} is the adversarial perturbations, and $L(\theta, x + r_{adv}, y)$ is the adversarial classification loss. For adversarial training, the minimization is over the network parameters and the maximization is over the adversarial perturbation. Some specific methods are discussed as follows.

4.1.1.1. FGSM adversarial training. While [14] proposed the FGSM attack method, they firstly put the adversarial examples into the training process. In this adversarial training, the adversarial examples x' are generated by non-target FGSM: $x' = x + \epsilon \cdot sign(\nabla J(\theta, x, y))$. And the adversarial objective function is formulated as follows:

$$\widetilde{J}(\theta, x, y) = \alpha J(\theta, x, y) + (1 - \alpha) J(\theta, \epsilon \cdot sign(\nabla J(\theta, x, y))), \tag{34}$$

where α is a hyperparameter, and the authors used $\alpha = 0.5$. From the formula, it can be seen that adversarial training can be regarded as an effective regularization. Experimental results showed that after FGSM adversarial training, the model's robustness to the adversarial examples generated by FGSM could be enhanced.

4.1.1.2. PGD adversarial training. FGSM generates adversarial examples through only one step iteration. Such adversarial examples may not be the strongest adversarial examples. Adversarial training based on FGSM is still vulnerable to iterative attacks [91]. To solve this problem, Madry et al. [15] tried to find the strongest adversarial samples through PGD, and used this adversarial sample for adversarial training. Different from FGSM adversarial training, PGD adversarial training is only trained on adversarial examples. Surprisingly, PGD adversarial training can improve the robustness of the model against a series of attack methods, including FGSM, PGD, C&W, on the MNIST and CIFAR10 datasets. PGD adversarial training is still the baseline method for many state-of-the-art defense methods.

4.1.1.3. Ensemble adversarial training. The method of Ensemble adversarial training (EAT), proposed by Tramèr et al. [92], is that when adversarial training is performed on a target model, the adversarial example is not generated by itself, but by several other pre-trained models of the same task. For each benign sample, each additional model generates an adversarial sample by the FGSM method. Then use these samples to conduct adversarial training on the target network. The main advantage of EAT is that the generation of adversarial samples and adversarial training can be carried out separately, which greatly improves the efficiency of adversarial training process compared with PGD. And although the adversarial examples are generated by the single-step attack method FGSM, the ensemble of multiple networks makes the robustness of adversarial training comparable to PGD.

4.1.1.4. Speed up adversarial training. From FGSM to PGD, it is mainly to optimize the strength of adversarial perturbations. Although better results have been achieved, the calculation has also increased. The computational load of PGD is $k + 1$ times that of ordinary training, and the huge computational consumption makes it difficult to apply PGD adversarial training to large data sets such as ImageNet [93]. To solve this problem, Adversarial Training for Free (FreeAT) [94] was proposed. The idea of FreeAT is to continuously repeat training m times for each sample x. and reuse the gradient of the previous step when calculating r. To ensure efficiency, the overall epoch will be divided by m. The update formula of r is:

$$r_{t+1} = r_t + \eta \cdot sign(g_t), \tag{35}$$

where g_t is the gradient at step t. As the same as PGD adversarial training, the training data in this case are all adversarial examples.

You Only Propagate Once (YOPO) [95] is also a method to increase the speed of PGD training. It reuses the loss gradient of the first layer output of the model during the generation of the PGD attack. For each set of updates, YOPO effectively reduce the total number of full forward and backward propagation to only one, In this way, YOPO reduces a lot of gradient calculations, thereby reducing computational costs.

4.1.1.5. Other adversarial training methods. Zheng et al. [96] showed that there's high transferability between models from neighboring epochs in the same training process. Therefore they proposed a method named Adversarial Training with Transferable Adversarial samples (ATTA), which used adversarial samples generated from the previous batch to generate the adversarial samples in the next batch. As the model was trained and the epoch increased, the adversarial perturbations were also cumulatively updated. The adversarial samples obtained in this way are more powerful, and could enhance the robustness of the model after adversarial training as well as greatly reduce the training time.

Zhang et al. [97] proposed an adversarial training method by using feature scattering. It takes the variation of the feature level as adversarial perturbations into account instead of the label layer. As an unsupervised training method, it does not leak the label. At the same time, it considers the whole batch samples, which makes the model obtain better generalization ability. Wang et al. [98] proposed Bilateral Adversarial Training (BAT) to generated both perturbed images and perturbed labels during adversarial training. Specifically, BAT adopts one-step PGD to generate adversarial examples, and uses the gradient with respect to input label to derive a formula to get perturbed label. For preventing the label leakage and solving the gradient masking problem, BAT uses the random start and the most confusing target attack. Yan et al. [99] proposed a training method called "Depth Defense", which integrates an adversarial perturbation-based regularizer into the classification objective. The trained model can resist potential attacks without reducing its accuracy. By characterizing the potential adversarial examples around a natural one under an entropic regularizer, Dong et al. [100] proposed adversarial distributional training (ADT) to improve robustness against unknown attacks. Shaham et al. [101] improved the local stability of the artificial neural network (ANN) through an alternating minimization-maximization process. The specific method is to minimize the loss of the network of the adversarial examples at each parameter update during training. Many studies also add additional mechanisms to adversarial training. Liu et al. [102] added a generator to the adversarial training process for joint optimization, and at the same time obtained a higher-quality generator and a more robust classifier. Pang et al. [103] incorporated the hypersphere embedding (HE) mechanism into the adversarial training.

Lee et al. [104] found the adversarial feature overfitting (AFO), which may lead to poor adversarial robust generalization, and that adversarial training can exceed the optimal point in terms of robust generalization, resulting in simple AFO in the Gaussian model. They proposed the adversarial vertex mixing (AVmixup) as a data augmentation solution to the AFO problem. Specifically, for each original input vector, a virtual vector in the adversarial direction is defined, and the training distribution is expanded through the linear interpolation of the virtual vector.

Generating adversarial examples for each benign picture requires a lot of computing resources, which has become the main reason that restricts the application of adversarial training to enlarge data sets. universal adversarial perturbations (UAPs) are perturbations that can attack multiple images at the same time. Shafahi et al. [105] attempted to apply universal adversarial perturbations to adversarial training to greatly reduce the training time. They applied stochastic gradient methods to craft adversarial perturbations, and provided a 'low-cost' algorithm for defending against universal perturbations by introducing FreeAT [94] method.

4.1.2. Neural network optimization

Some researchers suggest that the vulnerability of deep neural networks lies in the fragility of the network's structure and parameters, hence small image-level adversarial perturbations that can have serious effects in deep features. The neural network optimization based methods are to improve the robustness of the model by improving the parameters of the network, such as regularization [7,106,107], or adjusting the structure of the network, such as adding [108] or changing [109] some network layers.

4.1.2.1. Regularization. Szegedy et al. [7] firstly pointed that input gradient regularization [110] can improve the resistance of the model to adversarial perturbations, and suggested adding regularization term during training to enhance the robustness of the model. Moosavi et al. [106] proposed a regularization training method. Their methods trained differentiable models (for example, DNN), and at the same time penalized the degree of change in the output due to input changes. This means that a small adversarial perturbation has little effect on the output of the trained model. Lyu et al. [111] proposed a family of gradient regularization methods through simulated attacks, designed an unified framework to build robust machine learning models. Miyato et al. [26] used local distributional smoothness (LDS) as a regularization term to promote the smoothness of the model distribution, and named it as virtual adversarial training (VAT). Similar to adversarial training, LDS measures the robustness of the model to perturbations at each point and penalizes the loss. But unlike adversarial training, VAT does not require label information, which can be called semi-supervised learning and reduces computational consumption.

Cisse et al. [107] proposed Parseval networks and used Lipschitz constant for regularization. It improves robustness by keeping a small Lipschitz constant in each hidden layer.

Jakubovitz et al. [112] applied regularization using the Frobenius norm of the Jacobian of the network after regular training. It was proved that the Forbenius norm of the Jacobian at a given point is related to its distance to the closest adversarial example and to the curvature of the network's decision boundaries. In addition, using the Jacobian regularization requires only little additional computational resources. The above two points ensure that their method can effectively improve the robustness.

Training the network with Gaussian noise is an effective technique to realize the regularization of the model, thereby improves the robustness of the model when the inputs are distorted. He et al. [113] proposed a method named Parametric Noise Injection (PNI), which solves the Min–Max optimization problem, by applying trainable Gaussian noise on the activation or weight of each layer, and then doing the adversarial training.

4.1.2.2. Optimize neural architecture. Guo et al. [114] designed robust neural architecture search framework based on one-shot NAS. The concrete method was to design a super-net, and randomly change the architecture parameters $\alpha(\alpha \in [0,1])$ in the adversarial training against PGD. The target networks are sampled based on the super-net's performance. The networks searched out by this way were called Robnet family, which had less parameters. The experimental results demonstrated that densely connected pattern could benefit the network robustness and adding convolution operations in direct edges was more effective to improve the model's robustness. The robustness of Robnet wass about 5% higher than that of the basic network (such as Resnet and Densenet) under white-box and black-box attacks.

Gao et al. [108] introduced a defense mechanism called DeepCloak, by inserting a mask layer in the DNN model before the linear layer processing classification. This mask layer is trained by clean samples and their corresponding adversarial samples, and can identify and delete unnecessary features. Those features are exploited by the attacker and therefore should be removed to improve the robustness of the model.

In order to avoid small perturbations from the perspective of the activation function to have a huge impact on the model output,

Zantedeschi et al. [109] used bounded RELU (BRELU) [115] to limit the changes in the forward propagation, thereby improved the robustness of the model.

4.1.2.3. Random in DNNs structure. Xie et al. [116] proposed a new CNN structure. The output of the feature maps in this method are randomly masked. Thus, it forces filters to learn complete information from partial features, like dropout. As a result, it can enhance the ability of CNN and resist the adversarial attacks effectively.

4.1.3. Bayesian model-based defense

Uncertainty of bayesian neural networks (BNN) increases with attacking strengths. Rawat et al. [117] proposed that adversarial samples can be detected by quantizing this observation. Liu et al. [118] firstly used the BNN in adversarial training. They assumed that all weights in the network are random, and used the normal techniques in BNN for training. In their experiment, authors observed that BNN itself had no defense ability, but when combined with adversarial training, its robustness against the attack was significantly improved. Li et al. [119] improved the classical Naive Bayes with conditional deep generative models and proposed deep Bayes classifiers. Under generative models, adversarial examples are detected by rejecting inputs with low likelihood. Experimental results show that deep Bayesian classifiers are more robust than deep discriminative classifiers.

4.1.4. Defense by ensemble learning

Ensemble learning for adversarial defense can be summarized as: training a model ensemble, so that each model of this ensemble can achieve the same classification function, but each model should be distinguished as much as possible for better generalization ability.

Abbasi et al. [120] proposed a method that determined specialists on different subsets of classes by confusion matrices, and achieved the classification by using voting mechanism. Bagnall et al. [121] used multiple instances of a base model. The models were trained to minimize the cross-entry loss while reducing the consistency of their classification scores. In the prediction stage, rank voting mechanism was used to detect adversarial samples. It works well on FGSM attack, but was not so well for iterative attack (26.4% accuracy on classification and 27.1% on detection). Pang et al. [122] proposed the adaptive diversity promoting (ADP) training method. After training by the ADP method, the non-maximal predictions of each network would tend to be mutually orthogonal, and lead to different feature distributions and decision domains. The above mentioned methods cannot optimize the generalization ability of the ensemble well, which leads to low resistance rate against white-box attacks. Kariyappa et al. [123] presented the joint gradient phase and magnitude regularization (GPMR) scheme for improving the robustness of the ensemble against adversarial perturbations. Dabouei et al. [124] defined the gradient diversity promoting loss to increase the angle between the gradients on different models, and defined the gradient magnitude regularization loss to regularize the joint interactions of the members.

4.2. Passive defense: reduce or eliminate perturbations

Passive defense is not intended to strengthen the model itself, but to mitigate the damage to the model caused by the adversarial perturbation. The main advantage of this type of method is that it can decouple model training and adversarial defense, without changing the well-trained model, which is conducive to the application of defense methods.

4.2.1. Randomization

In the studies of DNN, it's observed that DNN is usually robust when we add randomization to the image. But for the adversarial samples, especially those generated by the gradient, perturbations are fragile under randomization. Based on this observation, researchers developed various defensive schemes based on randomization techniques to mitigate adversarial attacks, which are introduced as follows.

4.2.1.1. Random input transformation. Xie et al. [28] transformed the input image randomly before feeding it into CNNs. According to this paper, there are two kinds of transitions. The first one is to randomly resize the image, and the second one is to randomly zero padding around the image. Guo et al. [29] used more transformation methods, such as bit-depth, JPEG compression and image quilting. Experimental results demonstrated that these transformation methods have significant effects on gray-box attacks and black-box attacks, but their performance against the white-box attacks could be improved, especially for the EoT method [39]. Inspired by this, Raff et al. [125] considered that only a single transformation model is not reliable enough, and the exposure of transformation models may bring risks. Thus, they randomly selected some transformations and applied them in a random order. A fly in the ointment is that it may lead to the decline of classification accuracy. In order to alleviate this faultiness, Kou et al. [126] indicated that the distribution of SoftMax after image transformation contains the right prediction. They trained a distribution classifier on SoftMax outputs to make the final predictions, leading to an increase in accuracy. Taran et al. [127] introduced the multi-channel architecture before the input layer, and performed its own randomization on each channel.

4.2.1.2. Random nosing. Zantedeschi et al. [128] added Gaussian noises into the input data. It both enhanced the generalization ability of the model and mitigated the adversarial attacks. Liu et al. [129] designed a new defense method named Random Self-Ensemble (RSE). RSE inserts Gaussian noise layers into the DNN before each convolution layer. The different random noise constitutes the ensemble, and stable result is obtained by combining them. Li et al. [130] added random noises to the pixels of adversarial examples before inputting them into the classifier to eliminate the effects of adversarial perturbations.

4.2.2. Denoising and input preprocessing

As mentioned previously, many attack methods generate adversarial samples by adding adversarial perturbations to benign samples. Regarding adversarial perturbation as a special kind of noise, denoising is a feasible defense method. On the other hand, treating benign samples and adversarial samples as two different distributions, and converting the input to the distribution of benign samples through feasible preprocessing methods is also a solution.

4.2.2.1. Image denoising. [30] used a denoising autoencoder (DAE) [131] to remove arial pations. The results showed that DAEs can reduce adversarial noise, but superimposing it with the original network can make the result network more susceptible to perturbation. Based on this observation, Gu et al. [30] further proposed Deep Contractive Network (DCN), an end-to-end training model to eliminate adversarial perturbations. [132] introduced scalar quantization and smoothing spatial filter to reduce the impact of adversarial attacks. Liao et al. [31] proposed a high-level representation guided denoiser, which solved the problem of the error amplification effect of the standard denoiser.

4.2.2.2. Feature denoising. Dziugaite et al. [133] showed that small perturbations in the pixel space may lead to very substantial

"noise" in the feature maps. They added denoising blocks in intermediate layers of convolutional networks to improve adversarial robustness. This denoising block wraps the denoising operation with a 1×1 convolution and an identity skip connection. They compared four denoising operations (non-local means, bilateral filtering, mean filtering, and median filtering). Non-local means have the best performance. It shows that, by end-to-end adversarial training, denoising blocks work well on white-box attacks (e.g. PGD [16]) and black-box attacks.

4.2.2.3. Compression-based denoising. Researchers attempted to use compression methods to mitigate the adversarial examples. [133,134] removed adversarial perturbations from the inputs by using standard JPEG compression. [135,133] pointed out that standard JPEG compression cannot effectively remove the adversarial perturbations, and may reduce classification accuracy on the benign images. Thus, Liu et al. [133] reconstructed the JPEG compression framework. The proposed method was to insert a new pair of quantization/dequantization processes on standard JPEG decompression after the original dequantization stage.

4.2.2.4. Pixel-level preprocessing. Buckman et al. [136] believed that the reason for the adversarial samples is that there is a linear layer in the neural network. In order to break the linear extrapolation, they proposed Thermometer Encoding to replace each pixel with a corresponding binary vector. Due to the discrete nature of thermometer codes, it is impossible to directly perform gradient descent calculations on thermometer codes. Prakash et al. [137] proposed a method called pixel deflection to redistribute pixel values locally. Adversarial perturbation will be destroyed by pixel deflection, while the benign image has less influence, so as to achieve defensive goals.

Bhagoji et al. [138] proposed a strategy of integrating various data transformations (including dimensionality reduction) through principal component analysis, which can resist effectively against the best known evasion attacks, and is suitable for various machine learning classifiers including DNN.

Adversarial Perturbations is imperceptible for its slight change on benign image. Addepalli et al. [139] pointed that adversarial perturbations only appear in the lower bit planes. They proposed the Bit Plane Feature Consistency (BPFC) that used the higher bit planes to limit the range of predicted results, and used the lower bit planes only to refine the prediction.

4.2.3. Modified input

Generally speaking, adversarial perturbations are imperceptible, and adversarial examples are close to the classification boundary. Based on this observation, compared to traditional classifiers that classify based on only one input point, Cao et al. [140] integrated the information in a hypercube centered at the example to predict its label. Evaluation results on MNIST and CIFAR-10 datasets show that their method can significantly improve adversarial robustness without reducing classification accuracy. Xiao et al. [141] proposed a method where they make corresponding adversarial examples on a pre-trained external simple model before feeding the images into the target model for classification. They then feed these transformed images as input to the target model for training and prediction.

4.2.4. Generate benign Images

In order to ensure that the input is a benign sample, it is a very straightforward method to directly generate a benign sample corresponding to the original image. Unlike input preprocessing, the generation methods do not change pixel values of the original image, but give benign samples directly.

4.2.4.1. Rebuild the image. Akhtar et al. [32] proposed Perturbation Rectifying Network (PRN), which was trained by real and synthetic image-agnostic perturbations. And based on the input–output difference of this PRN, a detector was trained. In the prediction stage, according to the result of the detector, it is determined whether the original image or the output of the PRN is used as the input for prediction. Song et al. [33] proposed a method called PixelDefend to purify a maliciously perturbed image by moving the adversarial sample back to the distribution of training data. Jia et al. [142] proposed a method similar to auto-encoder method to purify adversarial perturbations. They designed a network named ComCNN to compress image and remove the redundant information. After adding the Gaussian noise, they used the networks named RecCNN to reconstruct the benign images. Treating each image as a point in the data space, this image space can be processed, such as clustering analysis using the density-based clustering framework [143]. Similarly, image reconstruction can also be performed on the image space to defend against attackers. Sun et al. [144] proposed a network layer structure called sparse transformation layer (STL) to generate benign images. By stratifying convolutional sparse coding, authors converted images (including adversarial images and benign images) from a natural image space to a low-dimensional quasi-natural space, and then rebuild the image from the quasi-natural space.

4.2.4.2. Approximate screening. The application of probability models in neural networks, especially probabilistic neural networks, can effectively improve the effect of pattern recognition [145–147]. Theagarajan et al. [148] proposed the Probabilistic Adversarial Robustness (PAR). It transforms the adversarial images to benign images by learning a probabilistic model, and the benign images sampled on clean space are close to the original images. In their work, they used the PixelCNN as the probabilistic model. It was proved that the lower bound of the loss function of PAR can be achieved. Thus, the model can sample images from the clean space. Hwang et al. [149] proposed Purifying Variational Autoencoder (PuVAE) to eliminate the adversarial perturbation. The PuVAE projects the adversarial example on the manifold of each class, and determines the closest projection as a purified sample. Dubey et al. [150] proposed to perform nearest-neighbor search to approximate projection on the unknown image manifold, and replace the original image with the nearest neighbor image for prediction.

4.2.4.3. Generate benign images based on GAN. It's a straightforward way to use a GAN to get the benign image distribution. Shen et al. [151] proposed a GAN named APE-GAN to obtain benign images. APE-GAN is trained by the adversarial inputs and its corresponding benign inputs. The APE-GAN performs well on attacks that have been used during training, but weakly on the adaptive white-box CW_2 attack [18]. Samangouei et al. [152] proposed the Defense-GAN to learn the benign image distribution. Defense-GAN is trained by image adding random noise instead of adversarial examples. When the adversary example is input, an approximate sample satisfying the benign sample distribution, which is close to the adversary sample, is generated by Defense-GAN. Then the approximate sample, instead of original image, is feed into the classifier for classification. The main problem of GAN-Based methods is their unstable training.

4.3. Information masking

Almost all white-box attacks require the knowledge of model information, and some black-box attack methods require approximate fitting of the model. Defending through information masking is a viable solution for many attacks.

4.3.1. Defensive distillation

Distillation is proposed by Hinton et al. [153]. This approach is to train a smaller model to fit the original model. Usually the original input and the output of the original model before SoftMax are used as the training data of the distillation model. Papernot et al. [154] apply distillation to defend adversarial perturbation, and named this mechanism as defensive distillation. If the gradient of the model is large, it is easy to generate adversarial examples near benign samples using gradient-based methods. Defensive distillation can reduce the gradient of the model and increase the robustness of the model.

4.3.2. Gradient masking/obfuscation

Many attack methods rely on the gradient information of the victim model. The gradient masking/obfuscation defense methods try to defend by hiding the gradient information. In a sense, almost all passive defense methods have the effect of gradient masking/obfuscation, because these defense methods hide the actual relationship between model input and output. There are also some methods to defend directly through the gradient masking/obfuscation. Naseer et al. [155] proposed a method called Local Gradient Smoothing (LGS) to defend localized adversarial attacks. LGS performs gradient smoothing in the region of interest with the highest probability of adversarial noise. In general, noise suppression is performed as follows:

$$\mathscr{T}(\mathbf{x}) = \mathbf{x} \odot (1 - \lambda * g(\mathbf{x})), \tag{36}$$

where $g(x)$ is the normalized gradient. To mitigate the drop in accuracy on benign examples, LGS first divides the gradient magnitude map into total k overlapping blocks of the same size (τ) and evaluates the gradient strength within a local window, then filters these blocks according to a threshold (γ):

$$\hat{\mathbf{g}}_{h,w} = \begin{cases} \mathbf{g}'_{h,w}, \textbf{if } \frac{1}{|\mathbf{g}'_{h,w}|}\sum_i\sum_j g'_{h,w}(i,\ j) > \gamma \\ 0, \quad \text{otherwise}, \end{cases} \tag{37}$$

where h,w denote the vertical and horizontal components of the top left corner of the extracted window.

Lecuyer et al. [27] proposed a defense method based on differential privacy called PixelDP. PixelDP adds a differential privacy noise layer to the deep neural network to randomize the calculation of the network. The specific noise addition method is:

$$A_{Q(x)} = h\big(\tilde{g}(x) + noise\big(\Delta_{p,q}, L, \varepsilon, \delta\big)\big), \tag{38}$$

where Q is the scoring function of the original DNN,and $A_{Q(x)}$ is a random function transformed from Q that satisfies Differential Privacy (DP). $\tilde{g}$ is the pre-noise calculation, and h is the subsequent calculation. For noise layer $noise\big(\Delta_{p,q}, L, \varepsilon, \delta\big)$, where L is the bound of the p-norm attack, $\Delta_{p,q}$ is the sensitivity of $\tilde{g}$ to changes in the p-norm input:

$$\Delta_{p,q} = \Delta^g_{p,q} = \max_{x,x':x\neq x'} \frac{\|g(x) - g(x')\|_q}{\|x - x'\|_p} \tag{39}$$

4.3.3. Boundary masking

Nguyen et al. [156] proposed noise augmented classifier (NAC), which randomly moves the classifier separator by injecting a very small noise into the last layer of the DNN classifier during runtime. This small noise will slightly change the decision boundary. For clean samples, it will not affect the classification accuracy, but it will destroy the small perturbation in adversarial samples.

4.4. Detection-based defense

Some researchers focus on detecting adversarial examples. The purpose of detection defense is to find adversarial examples and reject abnormal prediction results.

4.4.1. Detection on input

Melis et al. [157] determined whether to reject the sample by measuring the distance between the input and the known training data. Tian et al. [158] detected adversarial examples through image transformation. Xiao et al. [159] used spatial consistency information to detect adversarial examples. Specifically, in the image segmentation task, for each pixel m, the authors calculate its self-entropy:

$$\mathscr{H}(m) = -\sum_j \mathscr{V}_m[j] log \mathscr{V}_m[j], \tag{40}$$

where $\mathscr{V}_m[j]$ represents the number of times pixel m is predicted as class j. For benign instances, the boundary of the original object has higher entropy. Ma et al. [160] used Local Intrinsic Dimensionality (LID) to characterize the dimensionality characteristics of the adversarial subspace, and uses LID to identify adversarial samples. The maximum likelihood estimation (MLE) of LID is used in the experiment to approximate the expression:

$$\widehat{LID} = -\left(\frac{1}{k}\sum_{i=1}^{k} log\frac{r_i(x)}{r_k(x)}\right)^{-1}, \tag{41}$$

where $r_i(x)$ denotes the distance between x and its i-th nearest neighbor,and $r_k(x)$ is the maximum of the neighbor distances. Hendrycks et al. [161] found that there is almost no difference in principal components between adversarial samples and clean images. Therefore, principled component analysis (PCA) can be used to detect adversarial examples.

4.4.2. Detection on feature

Li et al. [162] used the statistical data of the output of the convolutional layer to determine whether the input is adversarial, and performed a small average filter on the adversarial example. Lee et al. [163] used Gaussian discriminant analysis (GDA) to predict the feature distribution, and then use Mahalanobis distance to calculate the confidence of adversarial samples. The authors used the output of all DNN layers, instead of only the last layer. This method can simultaneously detect out-of-distribution samples and adversarial samples.

4.4.3. Detecting by auxiliary model

The detection network method will directly predict whether a given sample is an adversarial sample through the neural network, *i.e.*, directly transform the recognition task of the adversarial sample into a binary classification problem trained in an end-to-end manner [164–166]. Metzen et al. [164] first trained the classifier on the original training data set. When the classifier training was completed, the attack method was used to generate the adversarial sample corresponding to each clean sample. Then they used the original data set and the same size adversarial sample data set to train the detector. Gong et al. [165] performed a second-round attacking test on the trained detector to verify the robustness of the detector. Grosse et al. [167] used the adversarial sample as an additional label for training, and The trained model would be able to classify and detect adversarial samples at the same time. Feinman et al. [168] proposed to extract the Bayesian uncertainty estimation in the dropout neural network and the density in the deep feature subspace as features to train a binary classifier, in order to distinguish adversarial samples.

The previously mentioned detector training methods all require adversarial sample data sets or need to understand the generation

process of adversarial samples, which is not only computationally expensive but also easy to be bypassed by slightly modified attacks. Meng et al. [166] proposed two new methods for detecting adversarial samples: detection based on reconstruction error and detection based on probability divergence. The method of detection based on reconstruction error used autoencoder to reconstruct the image, used the reconstruction error to measure the difference between the original image and the reconstructed image, and used a threshold to distinguish between normal samples and adversarial samples. The detector based on probability divergence used a classifier to predict both the previously reconstructed image and the original image, and the Jensen-Shannon divergence was used to measure the degree of difference in their prediction results.

4.4.4. Detecting by analyzing DNN's performance

Corneanu et al. [169] proposed that the performance of DNN can be represented by functional graph. Different test samples brought different changes to the functional graph. And thus, this could be used to determine whether the network works correctly during testing and reliably identifies adversarial attacks. Ma et al. [170] analyzed the internals of DNN models under various attacks, then extracted DNN invariants and used them to perform runtime adversarial sample detection.

Starting from the interpretability of the model, Tao et al. [171] identifies neurons that are critical to individual attributes. Enhance the activation value of key neurons and weaken the values of other neurons to obtain a converted model, and compare the output result of this model with the original model to identify adversarial examples. Zheng et al. [172] studied the output distribution of hidden neurons in DNN classifier. Then they proposed that the hidden states of DNN are quite different between adversarial samples and benign samples, which was used to reject adversarial inputs.

Hendrycks et al. [173] pointed out that correctly classified and out-of-distribution examples often have different SoftMax distributions. Based on this observation, they developed detection schemes on adversarial examples [161].

4.4.5. Detecting by consistency

Xu et al. [174] indicated that too large feature input spaces lead to the possibility of adversarial samples. They studied two methods called feature squeezing: reducing color bit depth and both local and non-local spatial smoothing. They compare the model's prediction of the original sample with the prediction of the sample after squeezing to determine whether the input is adversarial. In their follow-up work [175], feature Squeezing can also resist C&W attacks.

While using denoising technology to alleviate adversarial attacks, Liang et al. [132] detected adversarial examples by comparing the output results of the image before and after denoising in the classifier. Hendrycks et al. [161] added the classification information to the decoder during image reconstruction, and the adversarial sample has a completely different performance from the clean sample.

5. Discussion

5.1. Existence of adversarial examples

The term of adversarial examples was first proposed by Szegedy et al. [7]. By adding small perturbations to the images in the test set, it can cause the neural network to misclassify them. In fact, Feinman et al. [168] proposed a similar concept of fooling neural networks in 2013, which is called Evasion Attacks. After the researchers discovered this phenomenon, they gave various explanations for the existence of adversarial examples.

Szegedy et al. [7] believed that the existence of adversarial examples is caused by the non-linear nature or the high complexity of DNN models. The adversarial examples represent low-probability "pockets" in the manifold. In other words, the data distribution we sampled does not represent the complete real data distribution, and the adversarial samples cannot be correctly identified by the model because the model itself has not learned those adversarial samples. However, it was mentioned in the work of [15] that increasing the complexity of the model improves its robustness, and some linear models are also vulnerable to adversarial examples [14].

Another view is completely opposite, Goodfellow et al. [14] believes that linear behavior in high-dimensional spaces leads to the existence of adversarial samples. This view successfully explains the linear attack methods such as FGSM [14], PGD [15], etc.

Other research explained it from the perspective of decision-making boundaries. The decision boundary may be very close to the manifold of our data, so that a small perturbation may change to the opposite place of the decision boundary [176]. Subsequent studies explained the nature of the decision boundary from various perspectives. For example, the decision boundary is too flat [177], has a large local curvature [178], or is not flexible enough [178], etc.

5.2. Why deep neural networks are not robust?

Generally speaking, deep learning networks obtained by routine training are not resistant to malicious attacks without applying defense methods. Several studies have attempted to answer the underlying reasons for this phenomenon. Tsipras et al. [179] pointed out a phenomenon, the more robust the model achieved, the lower the accuracy on the clean test set would be. They believed that this phenomenon is the resulted from that robust classifiers learning feature representations are fundamentally different from standard classifiers. In addition, Jetley et al. [180] showed that the vulnerability of the model to adversarial attacks is essentially related to its excellent performance.

Some studies have made various explanations for this phenomenon. One of the most widely accepted among these is that Ilyas et al. [181] attributed it to the existence of non-robust features. They divided features into two categories: useful, robust features F_r, and useful, non-robust features F_{nr}. The F_r refers to the features that can help the classifier do classification tasks and the F_{nr} here may be some high-frequency features. When humans perform classification tasks, they mainly rely on F_r. However, in the training process of deep networks, both F_r and F_{nr} are relied on, and these non-robust features are the key to the model's resistance to fragility: once an attacker attacks the model by manipulating F_{nr}, it makes the accuracy of the model directly drop to 0, while humans will not be affected.

Starting from the relationship between robustness and input data, [182] believes that using more data is helpful to improve the robustness against adversarial samples, even unlabeled data [183]. Revisiting the misclassified examples [184] or using additional data for pre-training [185] can also enhance robustness. Andriushchenko et al. [62] found that the adversarial robustness increases as the number of target tasks increase.

Although some studies [186] had shown that for basic classification tasks, adversarial examples can always be unavoidable. However, the robustness of the deep neural model is still important. Firstly, it can help us understand the essential attributes of the deep learning model. More importantly, the security of the model should also be considered for deep learning deployment.

5.3. Measurement function

Generally speaking, adversarial samples should have the same attributes as the original image in human perception, but machine learning classifier has different results. Two definitions are actually

made here. One is to regard human perception as an absolutely correct label, and the other is to regard whether human perception is consistent or not as a measure of whether the perturbation is small.

Regarding the first point, Tramèr et al. [187] pointed out that if time is limited, humans may also misclassify adversarial samples. Regarding the second point, we are accustomed to using the l_p norm to measure the size of the perturbation, but this is not necessarily completely consistent with human perception. Some non-l_p-measured attacks [67,72] can also be effective, and it is possible for humans to ignore these attacks. Even if they all use the l_p norm, different samples should not be measured by a uniform perturbation size [188]. Some studies already proposed the usage of other measurement methods, such as Wasserstein distance [189], Trace-Norm [190], etc. However, simple mathematical expressions may not truly conform to the distance measurement of human perception. Laidlaw et al. [191] proposed the neural perceptual distance, which approximates the true perceptual distance with the help of the feature space of the deep network classifier.

5.4. The benefits of adversarial learning

Improving the robustness of the model is the main advantage of adversarial learning. In addition, adversarial learning can also bring some extra gains in other applications. For example, [192] applied game theory to image adversarial attacks and adversarial defenses, and proposed game strategies to protect the privacy of user photos in social media. Oh et al. [193] introduced the notion of semantic adversarial examples, and used images with reversed brightness as semantic adversarial examples to study the performance of the model.

Feature extraction is a key step in machine learning applications. For example, the innovative locally linear discriminant embedding (LLDE) [13] and constrained maximum variance mapping (CMVM) [194] two feature extraction methods have effectively improved the effect of pattern recognition. The success of deep neural networks can also be attributed to it being an excellent feature extraction methods. An important point of adversarial learning is that it can help us understand the inner expression of deep learning. Such as, [195] found that adversarial training helps CNN learn more shape-biased representations. [196] understood adversarial robustness as the lower bound of model robustness. They analyzed the generalization and robustness of the model by comparing ordinary noise and adversarial perturbation.

Generally speaking, increasing the robustness will bring about a decrease in accuracy [179], but Zhang et al. [197] proposed that using attack algorithms with early end iteration for adversarial training will improve the robustness without reducing the accuracy of the model. Xie et al. [198] found that using adversarial samples skillfully can improve the accuracy of image classifiers. A specific method is to use additional BN (Batch Normalization) [199] layers to generate adversarial examples, and deal with adversarial examples during adversarial training. [200] further applied this method to object detection.

Regarding the adversarial example as a special kind of data augmentation will also have a surprising effect, for example, better novelty detection performance [201], better transferability [202,203], improved Co-Training efficiency [204], etc.

5.5. The outlook for future research

Adversarial attacks will severely reduce the performance of deep learning technology on multiple computer vision tasks. In particular, deep learning is also vulnerable to adversarial attacks in the real world. There have been different views [7,14,176] trying to explain the reasons behind the vulnerability of deep neural networks to subtle adversarial perturbations. But these views are not consistent, nor can they explain these phenomena individually. We still need to conduct research on this issue, which is essential for building a robust deep learning model.

From another perspective, whether there exists the most robust model is another issue that should be addressed. Although some studies indicate that adversarial examples cannot be avoided [186], it's of much importance to develop more stronger defensive strategies against adversarial attacks.

The transferability of adversarial perturbation or universal adversarial perturbation is also worthy of attention, since that any model may be successfully attacked by an attacker without revealing its information using adversarial samples with high transferability. In the other hand, transferability can also be used for defense, for example, to create data sets for adversarial training [161].

Most of the defense methods mentioned above, such as adversarial training, adding noise, and denoising, are all heuristic. The effectiveness of these defenses has only been experimentally verified, not theoretically proven. Since there is no theoretical guarantee, in general, these heuristic defenses can only work against certain attack effects, which may be broken by a new attack in the future. Therefore, in order to study comprehensive defense methods that can deal with different attacks, many researchers strive to develop provable defense methods [205–207,27]. These provable defenses compute the worst-case loss after being attacked for a given network, and then minimize a lower bound on the worst-case loss. These methods maintain a certain degree of accuracy for a limited range of attack methods. But the existing problem is that these provable defense methods are usually aimed at small models or only used on simple datasets such as MNIST and CIFAR. In the future, it is necessary for researchers to expand these provable defense methods and design comprehensive defense methods that can target different attacks.

6. Conclusions

Although deep neural networks have made great achievements in various tasks, the emergence of adversarial examples is worthy of vigilance. We briefly summarized recent advances in adversarial attack and defense techniques, including the basic ideas and techniques of these methods. The reason for adversarial examples' existence and the way of their generation still to be further studied. Existing defense methods have made great progress in improving the adversarial robustness of models, but still could not provide comprehensively satisfying performance on all attacks.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.
[2] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Advances in neural information processing systems 25 (2012) 1097–1105.
[3] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
[4] G. Hinton, L. Deng, D. Yu, G.E. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath, et al., Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups, IEEE Signal processing magazine 29 (6) (2012) 82–97.
[5] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.

[6] I. Sutskever, O. Vinyals, Q.V. Le, Sequence to sequence learning with neural networks, in: Advances in neural information processing systems, 2014, pp. 3104–3112.
[7] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, R. Fergus, Intriguing properties of neural networks, arXiv preprint arXiv:1312.6199.
[8] S. Grigorescu, B. Trasnea, T. Cocias, G. Macesanu, A survey of deep learning techniques for autonomous driving, Journal of Field Robotics 37 (3) (2020) 362–386.
[9] A. Møgelmose, D. Liu, M.M. Trivedi, Traffic sign detection for us roads: Remaining challenges and a case for tracking, in: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), IEEE, 2014, pp. 1394–1399.
[10] J.W. Beletic, R. Blank, D. Gulbransen, D. Lee, M. Loose, E.C. Piquette, T. Sprafke, W.E. Tennant, M. Zandian, J. Zino, Teledyne imaging sensors: infrared imaging technologies for astronomy and civil space, in: High Energy, Optical, and Infrared Detectors for Astronomy III, Vol. 7021, SPIE, 2008, pp. 161–174.
[11] Z.-Q. Zhao, D.-S. Huang, B.-Y. Sun, Human face recognition based on multi-features using neural networks committee, Pattern recognition letters 25 (12) (2004) 1351–1358.
[12] W.-S. Chen, P.C. Yuen, J. Huang, D.-Q. Dai, Kernel machine-based one-parameter regularized fisher discriminant method for face recognition, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 35 (4) (2005) 659–669.
[13] B. Li, C.-H. Zheng, D.-S. Huang, Locally linear discriminant embedding: An efficient method for face recognition, Pattern Recognition 41 (12) (2008) 3813–3821.
[14] I.J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, arXiv preprint arXiv:1412.6572.
[15] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, A. Vladu, Towards deep learning models resistant to adversarial attacks, arXiv preprint arXiv:1706.06083.
[16] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, J. Li, Boosting adversarial attacks with momentum, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 9185–9193.
[17] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z.B. Celik, A. Swami, The limitations of deep learning in adversarial settings, in: 2016 IEEE European symposium on security and privacy (EuroS&P), 2016, pp. 372–387.
[18] N. Carlini, D. Wagner, Towards evaluating the robustness of neural networks, in: 2017 ieee symposium on security and privacy (sp), IEEE, 2017, pp. 39–57.
[19] S.-M. Moosavi-Dezfooli, A. Fawzi, P. Frossard, Deepfool: a simple and accurate method to fool deep neural networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2574–2582.
[20] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard, Universal adversarial perturbations, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1765–1773.
[21] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, A. Yuille, Adversarial examples for semantic segmentation and object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 1369–1378.
[22] J. Hendrik Metzen, M. Chaithanya Kumar, T. Brox, V. Fischer, Universal adversarial perturbations against semantic image segmentation, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2755–2764.
[23] S. Li, A. Neupane, S. Paul, C. Song, S.V. Krishnamurthy, A.K.R. Chowdhury, A. Swami, Adversarial perturbations against real-time video classification systems, arXiv preprint arXiv:1807.00458.
[24] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, D. Song, Robust physical-world attacks on deep learning visual classification, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 1625–1634.
[25] M. Sharif, S. Bhagavatula, L. Bauer, M.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, 2016, pp. 1528–1540.
[26] T. Miyato, S.-I. Maeda, M. Koyama, K. Nakae, S. Ishii, Distributional smoothing with virtual adversarial training, arXiv preprint arXiv:1507.00677.
[27] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, S. Jana, Certified robustness to adversarial examples with differential privacy, in: 2019 IEEE Symposium on Security and Privacy (SP), IEEE, 2019, pp. 656–672.
[28] C. Xie, J. Wang, Z. Zhang, Z. Ren, A. Yuille, Mitigating adversarial effects through randomization, arXiv preprint arXiv:1711.01991.
[29] C. Guo, M. Rana, M. Cisse, L. Van Der Maaten, Countering adversarial images using input transformations, arXiv preprint arXiv:1711.00117.
[30] S. Gu, L. Rigazio, Towards deep neural network architectures robust to adversarial examples, arXiv preprint arXiv:1412.5068.
[31] F. Liao, M. Liang, Y. Dong, T. Pang, X. Hu, J. Zhu, Defense against adversarial attacks using high-level representation guided denoiser, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1778–1787.
[32] N. Akhtar, J. Liu, A. Mian, Defense against universal adversarial perturbations, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 3389–3398.
[33] Y. Song, T. Kim, S. Nowozin, S. Ermon, N. Kushman, Pixeldefend: Leveraging generative models to understand and defend against adversarial examples, arXiv preprint arXiv:1710.10766.
[34] A. Kurakin, I. Goodfellow, S. Bengio, Adversarial machine learning at scale, arXiv preprint arXiv:1611.01236.
[35] A. Kurakin, I. Goodfellow, S. Bengio, et al., Adversarial examples in the physical world (2016).
[36] C. Xie, Z. Zhang, Y. Zhou, S. Bai, J. Wang, Z. Ren, A.L. Yuille, Improving transferability of adversarial examples with input diversity, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 2730–2739.
[37] J. Rony, L.G. Hafemann, L.S. Oliveira, I.B. Ayed, R. Sabourin, E. Granger, Decoupling direction and norm for efficient gradient-based l2 adversarial attacks and defenses, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 4322–4330.
[38] A. Athalye, N. Carlini, D. Wagner, Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples, in: International conference on machine learning, PMLR, 2018, pp. 274–283.
[39] A. Athalye, L. Engstrom, A. Ilyas, K. Kwok, Synthesizing robust adversarial examples, in: International conference on machine learning, PMLR, 2018, pp. 284–293.
[40] R. Fletcher, Practical methods of optimization, John Wiley & Sons, 2013.
[41] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, C.-J. Hsieh, Ead: elastic-net attacks to deep neural networks via adversarial examples, in: Thirty-second AAAI conference on artificial intelligence, 2018.
[42] K.R. Mopuri, U. Garg, R.V. Babu, Fast feature fool: A data independent approach to universal adversarial perturbations, arXiv preprint arXiv:1707.05572.
[43] K.R. Mopuri, A. Ganeshan, R.V. Babu, Generalizable data-free objective for crafting universal adversarial perturbations, IEEE transactions on pattern analysis and machine intelligence 41 (10) (2018) 2452–2465.
[44] K.R. Mopuri, P.K. Uppala, R.V. Babu, Ask, acquire, and attack: Data-free uap generation using class impressions, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 19–34.
[45] J. Hayes, G. Danezis, Learning universal adversarial perturbations with generative models, in: 2018 IEEE Security and Privacy Workshops (SPW), IEEE, 2018, pp. 43–49.
[46] K.R. Mopuri, U. Ojha, U. Garg, R.V. Babu, Nag: Network for adversary generation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 742–751.
[47] V. Khrulkov, I. Oseledets, Art of singular vectors and universal adversarial perturbations, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 8562–8570.
[48] S. Sarkar, A. Bansal, U. Mahbub, R. Chellappa, Upset and angri: Breaking high performance image classifiers, arXiv preprint arXiv:1707.01159.
[49] S. Baluja, I. Fischer, Adversarial transformation networks: Learning to generate adversarial examples, arXiv preprint arXiv:1703.09387.
[50] J. Hayes, G. Danezis, Machine learning as an adversarial service: Learning black-box adversarial examples, arXiv preprint arXiv:1708.05207 2.
[51] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural computation 1 (4) (1989) 541–551.
[52] A. Krizhevsky, G. Hinton, et al., Learning multiple layers of features from tiny images.
[53] Y. Jang, T. Zhao, S. Hong, H. Lee, Adversarial defense via learning to generate diverse attacks, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 2740–2749.
[54] Z. Zhao, D. Dua, S. Singh, Generating natural adversarial examples, arXiv preprint arXiv:1710.11342.
[55] M. Arjovsky, S. Chintala, L. Bottou, Wasserstein gan. arxiv 2017, arXiv preprint arXiv:1701.07875 30 (2017) 4.
[56] C. Xiao, B. Li, J.-Y. Zhu, W. He, M. Liu, D. Song, Generating adversarial examples with adversarial networks, arXiv preprint arXiv:1801.02610.
[57] A. Odena, C. Olah, J. Shlens, Conditional image synthesis with auxiliary classifier gans, in: International conference on machine learning, PMLR, 2017, pp. 2642–2651.
[58] Y. Song, R. Shu, N. Kushman, S. Ermon, Constructing unrestricted adversarial examples with generative models, arXiv preprint arXiv:1805.07894.
[59] S. Sabour, Y. Cao, F. Faghri, D.J. Fleet, Adversarial manipulation of deep representations, arXiv preprint arXiv:1511.05122.
[60] Y. Dong, T. Pang, H. Su, J. Zhu, Evading defenses to transferable adversarial examples by translation-invariant attacks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 4312–4321.
[61] W. Brendel, J. Rauber, M. Bethge, Decision-based adversarial attacks: Reliable attacks against black-box machine learning models, arXiv preprint arXiv:1712.04248.
[62] M. Andriushchenko, F. Croce, N. Flammarion, M. Hein, Square attack: a query-efficient black-box adversarial attack via random search, in: European Conference on Computer Vision, Springer, 2020, pp. 484–501.
[63] N. Narodytska, S.P. Kasiviswanathan, Simple black-box adversarial attacks on deep neural networks., in: CVPR Workshops, Vol. 2, 2017.
[64] J. Su, D.V. Vargas, K. Sakurai, One pixel attack for fooling deep neural networks, IEEE Transactions on Evolutionary Computation 23 (5) (2019) 828–841.
[65] S. Das, P.N. Suganthan, Differential evolution: A survey of the state-of-the-art, IEEE transactions on evolutionary computation 15 (1) (2010) 4–31.
[66] J.-X. Du, D.-S. Huang, X.-F. Wang, X. Gu, Shape recognition based on neural networks trained by differential evolution algorithm, Neurocomputing 70 (4–6) (2007) 896–903.

[67] T.B. Brown, D. Mané, A. Roy, M. Abadi, J. Gilmer, Adversarial patch, arXiv preprint arXiv:1712.09665.
[68] A. Liu, X. Liu, J. Fan, Y. Ma, A. Zhang, H. Xie, D. Tao, Perceptual-sensitive gan for generating adversarial patches, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 33, 2019, pp. 1028–1035.
[69] X. Liu, H. Yang, Z. Liu, L. Song, H. Li, Y. Chen, Dpatch: An adversarial patch attack on object detectors, arXiv preprint arXiv:1806.02299.
[70] S. Thys, W. Van Ranst, T. Goedemé, Fooling automated surveillance cameras: adversarial patches to attack person detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2019.
[71] K. Xu, G. Zhang, S. Liu, Q. Fan, M. Sun, H. Chen, P.-Y. Chen, Y. Wang, X. Lin, Adversarial t-shirt! evading person detectors in a physical world, in: European Conference on Computer Vision, Springer, 2020, pp. 665–681.
[72] C. Xiao, J.-Y. Zhu, B. Li, W. He, M. Liu, D. Song, Spatially transformed adversarial examples, arXiv preprint arXiv:1801.02612.
[73] Z. Zhao, Z. Liu, M. 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, 2020, pp. 1039–1048.
[74] A.S. Shamsabadi, R. Sanchez-Matilla, A. Cavallaro, Colorfool: Semantic adversarial colorization, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 1151–1160.
[75] A. Rozsa, E.M. Rudd, T.E. Boult, Adversarial diversity and hard positive generation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2016, pp. 25–32.
[76] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z.B. Celik, A. Swami, Practical black-box attacks against machine learning, in: Proceedings of the 2017 ACM on Asia conference on computer and communications security, 2017, pp. 506–519.
[77] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, C.-J. Hsieh, Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models, in: Proceedings of the 10th ACM workshop on artificial intelligence and security, 2017, pp. 15–26.
[78] A.N. Bhagoji, W. He, B. Li, D. Song, Practical black-box attacks on deep neural networks using efficient query mechanisms, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 154–169.
[79] N. Papernot, P. McDaniel, I. Goodfellow, Transferability in machine learning: from phenomena to black-box attacks using adversarial samples, arXiv preprint arXiv:1605.07277.
[80] Y. Shi, S. Wang, Y. Han, Curls & whey: Boosting black-box adversarial attacks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 6519–6527.
[81] Y. Liu, X. Chen, C. Liu, D. Song, Delving into transferable adversarial examples and black-box attacks, arXiv preprint arXiv:1611.02770.
[82] S. Cheng, Y. Dong, T. Pang, H. Su, J. Zhu, Improving black-box adversarial attacks with a transfer-based prior, arXiv preprint arXiv:1906.06919.
[83] W. Wu, Y. Su, X. Chen, S. Zhao, I. King, M.R. Lyu, Y.-W. Tai, Boosting the transferability of adversarial samples via attention, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 1161–1170.
[84] A. Shafahi, W.R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, T. Goldstein, Poison frogs! targeted clean-label poisoning attacks on neural networks, arXiv preprint arXiv:1804.00792.
[85] J. Tu, M. Ren, S. Manivasagam, M. Liang, B. Yang, R. Du, F. Cheng, R. Urtasun, Physically realizable adversarial examples for lidar object detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 13716–13725.
[86] R. Duan, X. Ma, Y. Wang, J. Bailey, A.K. Qin, Y. Yang, Adversarial camouflage: Hiding physical-world attacks with natural styles, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 1000–1008.
[87] X.-F. Wang, D.-S. Huang, H. Xu, An efficient local chan–vese model for image segmentation, Pattern Recognition 43 (3) (2010) 603–618.
[88] J. Lu, H. Sibai, E. Fabry, Adversarial examples that fool detectors, arXiv preprint arXiv:1712.02494.
[89] S.-T. Chen, C. Cornelius, J. Martin, D.H.P. Chau, Shapeshifter: Robust physical adversarial attack on faster r-cnn object detector, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2018, pp. 52–68.
[90] Y. Li, D. Tian, M.-C. Chang, X. Bian, S. Lyu, Robust adversarial perturbation on deep proposal-based models, arXiv preprint arXiv:1809.05962.
[91] H. Kim, W. Lee, J. Lee, Understanding catastrophic overfitting in single-step adversarial training, arXiv preprint arXiv:2010.01799.
[92] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, P. McDaniel, Ensemble adversarial training: Attacks and defenses, arXiv preprint arXiv:1705.07204.
[93] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE conference on computer vision and pattern recognition, Ieee, 2009, pp. 248–255.
[94] A. Shafahi, M. Najibi, A. Ghiasi, Z. Xu, J. Dickerson, C. Studer, L.S. Davis, G. Taylor, T. Goldstein, Adversarial training for free!, arXiv preprint arXiv:1904.12843.
[95] D. Zhang, T. Zhang, Y. Lu, Z. Zhu, B. Dong, You only propagate once: Accelerating adversarial training via maximal principle, arXiv preprint arXiv:1905.00877.
[96] H. Zheng, Z. Zhang, J. Gu, H. Lee, A. Prakash, Efficient adversarial training with transferable adversarial examples, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 1181–1190.
[97] H. Zhang, J. Wang, Defense against adversarial attacks using feature scattering-based adversarial training, arXiv preprint arXiv:1907.10764.
[98] J. Wang, H. Zhang, Bilateral adversarial training: Towards fast training of more robust models against adversarial attacks, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 6629–6638.
[99] Z. Yan, Y. Guo, C. Zhang, Deep defense: Training dnns with improved adversarial robustness, arXiv preprint arXiv:1803.00404.
[100] Y. Dong, Z. Deng, T. Pang, H. Su, J. Zhu, Adversarial distributional training for robust deep learning, arXiv preprint arXiv:2002.05999.
[101] U. Shaham, Y. Yamada, S. Negahban, Understanding adversarial training: Increasing local stability of neural nets through robust optimization, arXiv preprint arXiv:1511.05432.
[102] X. Liu, C.-J. Hsieh, Rob-gan: Generator, discriminator, and adversarial attacker, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 11234–11243.
[103] T. Pang, X. Yang, Y. Dong, K. Xu, J. Zhu, H. Su, Boosting adversarial training with hypersphere embedding, arXiv preprint arXiv:2002.08619.
[104] S. Lee, H. Lee, S. Yoon, Adversarial vertex mixup: Toward better adversarially robust generalization, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 272–281.
[105] A. Shafahi, M. Najibi, Z. Xu, J. Dickerson, L.S. Davis, T. Goldstein, Universal adversarial training, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 5636–5643.
[106] A.S. Ross, F. Doshi-Velez, Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients, in: Thirty-second AAAI conference on artificial intelligence, 2018.
[107] M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, N. Usunier, Parseval networks: Improving robustness to adversarial examples, in: International Conference on Machine Learning, PMLR, 2017, pp. 854–863.
[108] J. Gao, B. Wang, Z. Lin, W. Xu, Y. Qi, Deepcloak: Masking deep neural network models for robustness against adversarial samples, arXiv preprint arXiv:1702.06763.
[109] V. Zantedeschi, M.-I. Nicolae, A. Rawat, Efficient defenses against adversarial attacks, in: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, 2017, pp. 39–49.
[110] H. Drucker, Y. Le Cun, Improving generalization performance using double backpropagation, IEEE Transactions on Neural Networks 3 (6) (1992) 991–997.
[111] C. Lyu, K. Huang, H.-N. Liang, A unified gradient regularization family for adversarial examples, in: 2015 IEEE international conference on data mining, IEEE, 2015, pp. 301–309.
[112] D. Jakubovitz, R. Giryes, Improving dnn robustness to adversarial attacks using jacobian regularization, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 514–529.
[113] Z. He, A.S. Rakin, D. Fan, Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 588–597.
[114] M. Guo, Y. Yang, R. Xu, Z. Liu, D. Lin, When nas meets robustness: In search of robust architectures against adversarial attacks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 631–640.
[115] S.S. Liew, M. Khalil-Hani, R. Bakhteri, Bounded activation functions for enhanced training stability of deep neural networks on visual pattern recognition problems, Neurocomputing 216 (2016) 718–734.
[116] C. Xie, Y. Wu, L.v.d. Maaten, A.L. Yuille, K. He, Feature denoising for improving adversarial robustness, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 501–509.
[117] A. Rawat, M. Wistuba, M.-I. Nicolae, Adversarial phenomenon in the eyes of bayesian deep learning, arXiv preprint arXiv:1711.08244.
[118] X. Liu, Y. Li, C. Wu, C.-J. Hsieh, Adv-bnn: Improved adversarial defense through robust bayesian neural network, arXiv preprint arXiv:1810.01279.
[119] Y. Li, J. Bradshaw, Y. Sharma, Are generative classifiers more robust to adversarial attacks?, in: International Conference on Machine Learning, PMLR, 2019, pp 3804–3814.
[120] M. Abbasi, C. Gagné, Robustness to adversarial examples through an ensemble of specialists, arXiv preprint arXiv:1702.06856.
[121] A. Bagnall, R. Bunescu, G. Stewart, Training ensembles to detect adversarial examples, arXiv preprint arXiv:1712.04006.
[122] T. Pang, K. Xu, C. Du, N. Chen, J. Zhu, Improving adversarial robustness via promoting ensemble diversity, in: International Conference on Machine Learning, PMLR, 2019, pp. 4970–4979.
[123] S. Kariyappa, M.K. Qureshi, Improving adversarial robustness of ensembles with diversity training, arXiv preprint arXiv:1901.09981.
[124] A. Dabouei, S. Soleymani, F. Taherkhani, J. Dawson, N.M. Nasrabadi, Exploiting joint robustness to adversarial perturbations, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 1122–1131.
[125] E. Raff, J. Sylvester, S. Forsyth, M. McLean, Barrage of random transforms for adversarially robust defense, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 6528–6537.

[126] C. Kou, H.K. Lee, E.-C. Chang, T.K. Ng, Enhancing transformation-based defenses against adversarial attacks with a distribution classifier, in: International Conference on Learning Representations, 2019.
[127] O. Taran, S. Rezaeifar, T. Holotyak, S. Voloshynovskiy, Defending against adversarial attacks by randomized diversification, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 11226–11233.
[128] V. Zantedeschi, M.-I. Nicolae, A. Rawat, Efficient defenses against adversarial attacks, in: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, 2017, pp. 39–49.
[129] X. Liu, M. Cheng, H. Zhang, C.-J. Hsieh, Towards robust neural networks via random self-ensemble, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 369–385.
[130] B. Li, C. Chen, W. Wang, L. Carin, Certified adversarial robustness with additive noise, arXiv preprint arXiv:1809.03113.
[131] Y. Bengio, Learning deep architectures for AI, Now Publishers Inc, 2009.
[132] B. Liang, H. Li, M. Su, X. Li, W. Shi, X. Wang, Detecting adversarial image examples in deep neural networks with adaptive noise reduction, IEEE Transactions on Dependable and Secure Computing.
[133] G.K. Dziugaite, Z. Ghahramani, D.M. Roy, A study of the effect of jpg compression on adversarial images, arXiv preprint arXiv:1608.00853.
[134] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, L. Chen, M.E. Kounavis, D.H. Chau, Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression, arXiv preprint arXiv:1705.02900.
[135] R. Shin, D. Song, Jpeg-resistant adversarial images, in: NIPS 2017 Workshop on Machine Learning and Computer Security, Vol. 1, 2017.
[136] J. Buckman, A. Roy, C. Raffel, I. Goodfellow, Thermometer encoding: One hot way to resist adversarial examples, in: International Conference on Learning Representations, 2018.
[137] A. Prakash, N. Moran, S. Garber, A. DiLillo, J. Storer, Deflecting adversarial attacks with pixel deflection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 8571–8580.
[138] A.N. Bhagoji, D. Cullina, C. Sitawarin, P. Mittal, Enhancing robustness of machine learning systems via data transformations, in: 2018 52nd Annual Conference on Information Sciences and Systems (CISS), IEEE, 2018, pp. 1–5.
[139] S. Addepalli, A. Baburaj, G. Sriramanan, R.V. Babu, Towards achieving adversarial robustness by enforcing feature consistency across bit planes, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 1020–1029.
[140] X. Cao, N.Z. Gong, Mitigating evasion attacks to deep neural networks via region-based classification, in: Proceedings of the 33rd Annual Computer Security Applications Conference, 2017, pp. 278–287.
[141] C. Xiao, C. Zheng, One man's trash is another man's treasure: Resisting adversarial examples by adversarial examples, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 412–421.
[142] X. Jia, X. Wei, X. Cao, H. Foroosh, Comdefend: An efficient image compression model to defend adversarial examples, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 6084–6092.
[143] X.-F. Wang, D.-S. Huang, A novel density-based clustering framework by using level set method, IEEE Transactions on knowledge and data engineering 21 (11) (2009) 1515–1531.
[144] B. Sun, N.-H. Tsai, F. Liu, R. Yu, H. Su, Adversarial defense by stratified convolutional sparse coding, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 11447–11456.
[145] D.-S. Huang, Radial basis probabilistic neural networks: Model and application, International Journal of Pattern Recognition and Artificial Intelligence 13 (07) (1999) 1083–1101.
[146] D.-S. Huang, J.-X. Du, A constructive hybrid structure optimization methodology for radial basis probabilistic neural networks, IEEE Transactions on neural networks 19 (12) (2008) 2099–2115.
[147] J.-X. Du, D.-S. Huang, G.-J. Zhang, Z.-F. Wang, A novel full structure optimization algorithm for radial basis probabilistic neural networks, Neurocomputing 70 (1–3) (2006) 592–596.
[148] R. Theagarajan, M. Chen, B. Bhanu, J. Zhang, Shieldnets: Defending against adversarial attacks using probabilistic adversarial robustness, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 6988–6996.
[149] U. Hwang, J. Park, H. Jang, S. Yoon, N.I. Cho, Puvae: A variational autoencoder to purify adversarial examples, arXiv preprint arXiv:1903.00585.
[150] A. Dubey, L. v. d. Maaten, Z. Yalniz, Y. Li, D. Mahajan, Defense against adversarial images using web-scale nearest-neighbor search, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 8767–8776.
[151] S. Shen, G. Jin, K. Gao, Y. Zhang, Ape-gan: Adversarial perturbation elimination with gan, arXiv preprint arXiv:1707.05474.
[152] P. Samangouei, M. Kabkab, R. Chellappa, Defense-gan: Protecting classifiers against adversarial attacks using generative models, arXiv preprint arXiv:1805.06605.
[153] G. Hinton, O. Vinyals, J. Dean, Distilling the knowledge in a neural network, arXiv preprint arXiv:1503.02531.
[154] N. Papernot, P. McDaniel, X. Wu, S. Jha, A. Swami, Distillation as a defense to adversarial perturbations against deep neural networks, in: 2016 IEEE symposium on security and privacy (SP), IEEE, 2016, pp. 582–597.
[155] M. Naseer, S. Khan, F. Porikli, Local gradients smoothing: Defense against localized adversarial attacks, in: 2019 IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE, 2019, pp. 1300–1307.
[156] L. Nguyen, S. Wang, A. Sinha, A learning and masking approach to secure learning, in: International Conference on Decision and Game Theory for Security, Springer, 2018, pp. 453–464.
[157] M. Melis, A. Demontis, B. Biggio, G. Brown, G. Fumera, F. Roli, Is deep learning safe for robot vision? adversarial examples against the icub humanoid, in: Proceedings of the IEEE International Conference on Computer Vision Workshops, 2017, pp. 751–759.
[158] S. Tian, G. Yang, Y. Cai, Detecting adversarial examples through image transformation, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
[159] C. Xiao, R. Deng, B. Li, F. Yu, M. Liu, D. Song, Characterizing adversarial examples based on spatial consistency information for semantic segmentation, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 217–234.
[160] X. Ma, B. Li, Y. Wang, S.M. Erfani, S. Wijewickrema, G. Schoenebeck, D. Song, M.E. Houle, J. Bailey, Characterizing adversarial subspaces using local intrinsic dimensionality, arXiv preprint arXiv:1801.02613.
[161] D. Hendrycks, K. Gimpel, Early methods for detecting adversarial images, arXiv preprint arXiv:1608.00530.
[162] X. Li, F. Li, Adversarial examples detection in deep networks with convolutional filter statistics, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 5764–5772.
[163] K. Lee, K. Lee, H. Lee, J. Shin, A simple unified framework for detecting out-of-distribution samples and adversarial attacks, arXiv preprint arXiv:1807.03888.
[164] J.H. Metzen, T. Genewein, V. Fischer, B. Bischoff, On detecting adversarial perturbations, arXiv preprint arXiv:1702.04267.
[165] Z. Gong, W. Wang, W.-S. Ku, Adversarial and clean data are not twins, arXiv preprint arXiv:1704.04960.
[166] D. Meng, H. Chen, Magnet: a two-pronged defense against adversarial examples, in: Proceedings of the 2017 ACM SIGSAC conference on computer and communications security, 2017, pp. 135–147.
[167] K. Grosse, P. Manoharan, N. Papernot, M. Backes, P. McDaniel, On the (statistical) detection of adversarial examples, arXiv preprint arXiv:1702.06280.
[168] R. Feinman, R.R. Curtin, S. Shintre, A.B. Gardner, Detecting adversarial samples from artifacts, arXiv preprint arXiv:1703.00410.
[169] C.A. Corneanu, M. Madadi, S. Escalera, A.M. Martinez, What does it mean to learn in deep networks? and how does one detect adversarial attacks?, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp 4757–4766.
[170] S. Ma, Y. Liu, Nic: Detecting adversarial samples with neural network invariant checking, in: Proceedings of the 26th Network and Distributed System Security Symposium (NDSS 2019), 2019.
[171] G. Tao, S. Ma, Y. Liu, X. Zhang, Attacks meet interpretability: Attribute-steered detection of adversarial samples, arXiv preprint arXiv:1810.11580.
[172] Z. Zheng, P. Hong, Robust detection of adversarial attacks by modeling the intrinsic properties of deep neural networks, in: Proceedings of the 32nd International Conference on Neural Information Processing Systems, 2018, pp. 7924–7933.
[173] D. Hendrycks, K. Gimpel, A baseline for detecting misclassified and out-of-distribution examples in neural networks, arXiv preprint arXiv:1610.02136.
[174] W. Xu, D. Evans, Y. Qi, Feature squeezing: Detecting adversarial examples in deep neural networks, arXiv preprint arXiv:1704.01155.
[175] W. Xu, D. Evans, Y. Qi, Feature squeezing mitigates and detects carlini/wagner adversarial examples, arXiv preprint arXiv:1705.10686.
[176] T. Tanay, L. Griffin, A boundary tilting persepective on the phenomenon of adversarial examples, arXiv preprint arXiv:1608.07690.
[177] A. Fawzi, S.-M. Moosavi-Dezfooli, P. Frossard, Robustness of classifiers: from adversarial to random noise, arXiv preprint arXiv:1608.08967.
[178] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard, S. Soatto, Analysis of universal adversarial perturbations, ArXiv e-prints (2017) arXiv–1705.
[179] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, A. Madry, Robustness may be at odds with accuracy, arXiv preprint arXiv:1805.12152.
[180] S. Jetley, N.A. Lord, P.H. Torr, With friends like these, who needs adversaries?, arXiv preprint arXiv:1807.04200.
[181] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, A. Madry, Adversarial examples are not bugs, they are features, arXiv preprint arXiv:1905.02175.
[182] L. Schmidt, S. Santurkar, D. Tsipras, K. Talwar, A. Mdry, Adversarially robust generalization requires more data, arXiv preprint arXiv:1804.11285.
[183] Y. Carmon, A. Raghunathan, L. Schmidt, P. Liang, J.C. Duchi, Unlabeled data improves adversarial robustness, arXiv preprint arXiv:1905.13736.
[184] Y. Wang, D. Zou, J. Yi, J. Bailey, X. Ma, Q. Gu, Improving adversarial robustness requires revisiting misclassified examples, in: International Conference on Learning Representations, 2019.
[185] D. Hendrycks, K. Lee, M. Mazeika, Using pre-training can improve model robustness and uncertainty, International Conference on Machine Learning, PMLR (2019) 2712–2721.
[186] A. Shafahi, W.R. Huang, C. Studer, S. Feizi, T. Goldstein, Are adversarial examples inevitable?, arXiv preprint arXiv:1809.02104.
[187] F. Tramèr, J. Behrmann, N. Carlini, N. Papernot, J.-H. Jacobsen, Fundamental tradeoffs between invariance and sensitivity to adversarial perturbations, in: International Conference on Machine Learning, PMLR, 2020, pp. 9561–9571.

[188] M. Cheng, Q. Lei, P.-Y. Chen, I. Dhillon, C.-J. Hsieh, Cat: Customized adversarial training for improved robustness, arXiv preprint arXiv:2002.06789.
[189] E. Wong, F. Schmidt, Z. Kolter, Wasserstein adversarial examples via projected sinkhorn iterations, in: International Conference on Machine Learning, PMLR, 2019, pp. 6808–6817.
[190] E. Kazemi, T. Kerdreux, L. Wang, Trace-norm adversarial examples, arXiv preprint arXiv:2007.01855.
[191] C. Laidlaw, S. Singla, S. Feizi, Perceptual adversarial robustness: Defense against unseen threat models, arXiv preprint arXiv:2006.12655.
[192] S.J. Oh, M. Fritz, B. Schiele, Adversarial image perturbation for privacy protection a game theory perspective, in: 2017 IEEE International Conference on Computer Vision (ICCV), IEEE, 2017, pp. 1491–1500.
[193] H. Hosseini, B. Xiao, M. Jaiswal, R. Poovendran, On the limitation of convolutional neural networks in recognizing negative images, in: 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), IEEE, 2017, pp. 352–358.
[194] B. Li, D.-S. Huang, C. Wang, K.-H. Liu, Feature extraction using constrained maximum variance mapping, Pattern Recognition 41 (11) (2008) 3287–3294.
[195] T. Zhang, Z. Zhu, Interpreting adversarially trained convolutional neural networks, International Conference on Machine Learning, PMLR (2019) 7502–7511.
[196] N. Ford, J. Gilmer, N. Carlini, D. Cubuk, Adversarial examples are a natural consequence of test error in noise, arXiv preprint arXiv:1901.10513.
[197] J. Zhang, X. Xu, B. Han, G. Niu, L. Cui, M. Sugiyama, M. Kankanhalli, Attacks which do not kill training make adversarial learning stronger, in: International Conference on Machine Learning, PMLR, 2020, pp. 11278–11287.
[198] C. Xie, M. Tan, B. Gong, J. Wang, A.L. Yuille, Q.V. Le, Adversarial examples improve image recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 819–828.
[199] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: International conference on machine learning, PMLR, 2015, pp. 448–456.
[200] X. Chen, C. Xie, M. Tan, L. Zhang, C.-J. Hsieh, B. Gong, Robust and accurate object detection via adversarial learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 16622–16631.
[201] M. Salehi, A. Arya, B. Pajoum, M. Otoofi, A. Shaeiri, M.H. Rohban, H.R. Rabiee, Arae: Adversarially robust training of autoencoders improves novelty detection, Neural Networks 144 (2021) 726–736.
[202] H. Salman, A. Ilyas, L. Engstrom, A. Kapoor, A. Madry, Do adversarially robust imagenet models transfer better?, arXiv preprint arXiv:2007.08489.
[203] F. Utrera, E. Kravitz, N.B. Erichson, R. Khanna, M.W. Mahoney, Adversarially-trained deep nets transfer better, arXiv preprint arXiv:2007.05869.
[204] S. Qiao, W. Shen, Z. Zhang, B. Wang, A. Yuille, Deep co-training for semi-supervised image recognition, in: Proceedings of the european conference on computer vision (eccv), 2018, pp. 135–152.
[205] E. Wong, Z. Kolter, Provable defenses against adversarial examples via the convex outer adversarial polytope, International Conference on Machine Learning, PMLR (2018) 5286–5295.
[206] E. Wong, F. Schmidt, J.H. Metzen, J.Z. Kolter, Scaling provable adversarial defenses, Advances in Neural Information Processing Systems 31.
[207] A. Sinha, H. Namkoong, R. Volpi, J. Duchi, Certifying some distributional robustness with principled adversarial training, arXiv preprint arXiv:1710.10571.