International Journal of Information Security (2024) 23:299–314
https://doi.org/10.1007/s10207-023-00735-6

REGULAR CONTRIBUTION


# Detection of adversarial attacks based on differences in image entropy

**Gwonsang Ryu**[1] · **Daeseon Choi**[2]

Published online: 17 August 2023


**Abstract**

Although deep neural networks (DNNs) have achieved high performance across various applications, they are often deceived by adversarial examples generated by adding small perturbations. To combat adversarial attacks, many detection methods have been proposed, including feature squeezing and trapdoor. However, these methods rely on the output of DNNs or involve training a separate network to detect adversarial examples, which leads to high computational costs and low efficiency. In this study, we propose a simple and effective approach called the entropy-based detector (EBD) to protect DNNs from various adversarial attacks. EBD detects adversarial examples by comparing the difference in entropy between the input sample before and after bit depth reduction. We show that EBD can detect over 98% of the adversarial examples generated by attacks using fast-gradient sign method, basic iterative method, momentum iterative method, DeepFool and CW attacks when the false positive rate is 2.5% for CIFAR-10 and ImageNet datasets.

**Keywords** Adversarial attack · Adversarial defense · Adversarial examples · Deep neural network · Image entropy

## 1 Introduction

Deep neural networks (DNNs) have attained high performance in various fields such as image classification [1], object detection [2], and natural language processing [3]. Despite their great success, DNNs are known to be vulnerable to adversarial attacks which generate maliciously crafted inputs to misclassify the sample and generate an incorrect output. These maliciously crafted inputs, which are created by imperceptible perturbations to the original sample, are called adversarial examples, and various attack methods have been proposed to generate them [4–6]. The purpose of these attacks is to maximize the misclassification while minimizing the perturbations so that a human cannot distinguish between a legitimate sample and an adversarial example. It is important to defend against adversarial attacks. These adversarial attacks are able to deceive models trained on different network architectures or different subsets of training data [4, 7]. Moreover, they have been shown to be effective on DNNs deployed in real-world settings, such as self-driving cars [8, 9], facial recognition [10, 11], and object recognition [12].

Many methods to thwart adversarial attacks have been proposed, including adversarial training [13, 22], defensive distillation [14], detection [15, 30], and denoising [17, 18]. Feature squeezing [15] detects adversarial examples by comparing differences between model's predictions of an original input sample and input samples that have been denoised by bit depth reduction and median smoothing. However, if these denoising methods, such as bit depth reduction and median smoothing, do not mitigate the perturbations in adversarial examples, the feature squeezing cannot detect the adversarial examples because the model's predictions of the denoised input samples are not much different from the model's predictions of the input sample. Trapdoor [58] utilizes backdoor samples to train DNNs that are robust against adversarial attacks. However, trapdoor takes a long time to train DNNs because it generates backdoor samples for all classes.

In this study, we propose a simple and effective approach called the entropy-based detector (EBD), which is a defense method that compares the entropy of an input sample before and after reducing the bit depth, to defend DNNs against a variety of adversarial attacks. A defense method that uses bit-depth reduction as a denoising method would remain vulnerable to adaptive attacks, but bit-depth reduction is also an

✉ Daeseon Choi
sunchoi@ssu.ac.kr

Gwonsang Ryu
gsryu@ssu.ac.kr

[1] Cyber Security Research Center, Soongsil University, Seoul, Republic of Korea

[2] Department of Software, Soongsil University, Seoul, Republic of Korea

effective way to decrease the entropy of an image. Therefore, we do not use bit-depth reduction as a denoising method; instead, we only use bit-depth reduction to calculate the difference in the entropy of an input sample after the bit-depth has been reduced. Our proposed method is motivated by the following observations: (1) The entropy of an adversarial example increases because it has perturbations with different values. (2) The entropy of an image in which the bit depth has been reduced is lower than the entropy of the original image. (3) The change in the entropy of an adversarial example after bit-depth reduction is larger than that of a normal image after bit-depth reduction. We demonstrate that the EBD is able to detect adversarial examples with high accuracy and a low false positive rate.

The major contributions of this work can be summarized as follows:

- We propose a simple and effective detector for defending against various adversarial attacks.
- We analyze the entropy of legitimate samples and adversarial examples generated by each attack method before and after bit depth has been reduced. When the bit depth is reduced, we show that the average entropy of adversarial examples is similar to that of legitimate samples.
- We show that EBD can successfully detect over 99% and 98% of adversarial examples on the CIFAR-10 and ImageNet datasets, respectively, with a low false positive rate.

The rest of this paper is organized as follows. Section 2 reviews related work, and Sect. 3 describes the our proposed method for detecting adversarial attacks. In Sect. 4, we analyze the entropy of legitimate samples and adversarial examples, and evaluate the detection performance of our proposed method. Section 5 discusses the our detection method, and Sect. 6 concludes the paper.

## 2 Related work

### 2.1 Adversarial attack

Adversarial attacks can be divided into two types of threat model: white-box attacks and black-box attacks. A white-box adversary has full access to the DNN parameters, network architecture, and weights. A black-box adversary has no knowledge of the target DNN or cannot access it. As for white-box attacks, Szegedy et al. [4] were the first to show that the existence of small perturbations in images could fool deep DNNs into misclassification. They generated adversarial examples using box-constrained L-BFGS. Goodfellow et al. [19] proposed the fast-gradient sign method (FGSM) to generate adversarial examples. They only perform a one-step gradient update along the direction of the gradient at each pixel. Kurakin et al. [20] introduced the basic iterative method (BIM) by repeating FGSM for $n$ steps. BIM usually results in a higher attack success rate than FGSM. Seyed-Mohsen et al. [6] proposed the DeepFool attack, which iteratively pushes the adversarial image toward the decision boundary of the target class by estimating the minimum perturbation required for misclassification. Carlini et al. [21] proposed an optimization-based attack method that aims to find adversarial examples with minimal perturbations while considering different threat models. It formulates the attack as an optimization problem that minimizes both the perturbation magnitude and the distance to the target class. Xiao et al. [23] proposed AdvGAN, a method that utilizes a generative adversarial network (GAN) [24] to generate adversarial examples by training and approximating the distribution of original images. Na et al. [59] proposed a contour attack method aimed at evading steganalysis-based detection. The method involves extracting the contour region from an image and applying perturbations exclusively to the relevant regions. Athalye et al. [25] proposed the backward pass differentiable approximation (BPDA) attack aimed at overcoming the challenges posed by non-differentiable operations or functions that may exist within the network architecture.

As for black-box attacks, Papernot et al. [26] introduced the first black-box attack method that utilizes a substitute model. This method involves training a local model to serve as a substitute for the target DNN. The training process involves using inputs that are synthetically generated by an adversary and labeled by the target DNN. Park et al. [27] proposed a novel black-box attack method that utilizes a substitute model. This method involves training the substitute model with a specific focus on shifting the decision boundary toward the desired class of the attacks." Dong et al. [28] proposed a momentum-based iterative method (MIM) to enhance the transferability of adversarial examples. This method incorporates a momentum term into the iterative attack process, allowing for the generation of more transferable adversarial examples. Inkawhich et al. [29] showed that the layer at which features are perturbed has a large impact on the transferability of adversarial examples. Furthermore, they found that the architecture of the black-box model does not affect the characteristics of layer-wise transferability. Xie et al. [7] proposed the DI$^2$-FGSM (Diverse Inputs Iterative-FGSM) attack to enhance the transferability of adversarial examples. This method incorporates image transformations into the inputs during each iteration of the BIM (basic iterative method) attack, aiming to alleviate the overfitting phenomenon.

### 2.2 Adversarial detection

For adversarial detection, Meng et al. [30] proposed a detection method called MagNet that uses the differences between the input sample and a sample reconstructed by an autoencoder. In addition, MagNet includes a reformer network that purifies adversarial perturbations on adversarial examples. Metzen et al. [42] proposed training a binary detector that obtains inputs from the intermediate features of a classifier and then discriminates between legitimate samples and adversarial examples. Lu et al. [43] proposed SafetyNet, which extracts the binary threshold of each rectified linear unit (ReLU) layer's output as features of the detector and detects adversarial examples using a radial basis function-based support vector machine. SafetyNet consists of the original classifier and a detector, which looks at the internal state of the later layers in the original classifier. If the detector declares that a sample is adversarial, it is rejected. Xu et al. [15] proposed feature squeezing, which detects adversarial examples by comparing the model's prediction for the original input samples with its prediction for the denoised image, which has been denoised using methods such as bit depth reduction, median smoothing, and non-local smoothing. Ye et al. [16] proposed a prediction inconsistency-based detection method, which is based on the view of enhancing model interpretability by exploiting the saliency map of the activated region to visualize the prediction process using DNNs. Ma et al. [52] analyzed the internals of DNNs under various adversarial attacks and identified two common exploitation channels: provenance and activation value distribution. Subsequently, they proposed a detection method that extracts DNN invariants and uses them to perform runtime adversarial detection. Ma et al. [53] also proposed local intrinsic dimensionality (LID) for the characterization of the adversarial regions of DNNs. They showed that the adversarial regions generated by various adversarial attacks share similar dimensional properties. Cohen et al. [54] proposed a detection strategy that uses nearest neighbors influence functions (NNIF) to ensure correspondence between the training data and the classification of DNNs. They used influence functions to measure the impact of every training sample on the validation samples. Then, they found the most supportive training samples for any given validation sample from the influence scores. Shan et al. [58] introduced the concept of trapdoors and trapdoor-enabled detection as honeypots to defend DNNs against various adversarial attacks. They used backdoor attacks on DNNs to train a trapdoored model and presented analytical proofs of the efficacy of trapdoors in influencing the generation of adversarial examples.

### 2.3 Other adversarial defense

Adversarial defenses can be divided into adversarial detection, adversarial training, defensive distillation, denoising, input transformation, and classifier robustness enhancement. For adversarial training, Madry et al. [13] suggested training with adversarial examples generated by projected gradient descent. Miyato et al. [40] proposed a virtual adversarial training method to smooth the output distributions of DNNs. Zheng et al. [22] proposed an adversarial training method that trains a target model with transferable adversarial examples to enhance the robustness of trained models. Ryu et al. [57] proposed an adversarial training method that trains DNNs to be robust against transferable adversarial examples and to maximize their classification performance for clean images in black-box settings. Ryu et al. [56] proposed a hybrid adversarial training approach that trains denoising networks and DNNs simultaneously. The objective is to improve the classification accuracy for normal samples and enhance the robustness of DNNs against adversarial attacks.

For defensive distillation, Papernot et al. [14] proposed the notion of distillation to ensure the robustness of DNNs against adversarial examples. Papernot et al. [41] also extended the distillation method by addressing numerical instabilities.

For denoising, Samangouei et al. [31] proposed Defense-GAN, which projects the adversarial examples into a trained generative adversarial network (GAN) to approximate the input sample using a generated clean sample. Song et al. [32] proposed PixelDefend, which reconstructs adversarial examples so that they follow the training distribution using PixelCNN [33]. Prakash et al. [17] proposed pixel deflection, which is a defense method in which some pixels are randomly replaced with nearby pixels and then uses wavelets to denoise the image. Naseer et al. [18] proposed NRP, which is a method that recovers a legitimate sample from a given adversarial example using an adversarially trained purifier network.

For input transformation, Guo et al. [34] proposed several input transformations to defend against adversarial examples such as image cropping and re-scaling, bit-depth reduction, and JPEG compression. Xie et al. [35] proposed defending against adversarial examples by adding a randomization that randomly rescales the input sample and then randomly zero-pads it. Buckman et al. [45] proposed thermometer encoding, which performs thermometer code discretization of the input values to improve the robustness of DNNs to adversarial attacks. Yin et al. [46] proposed a defense method, which performs two image transformations including WebP lossy compression and image flip to mitigate adversarial perturbations.

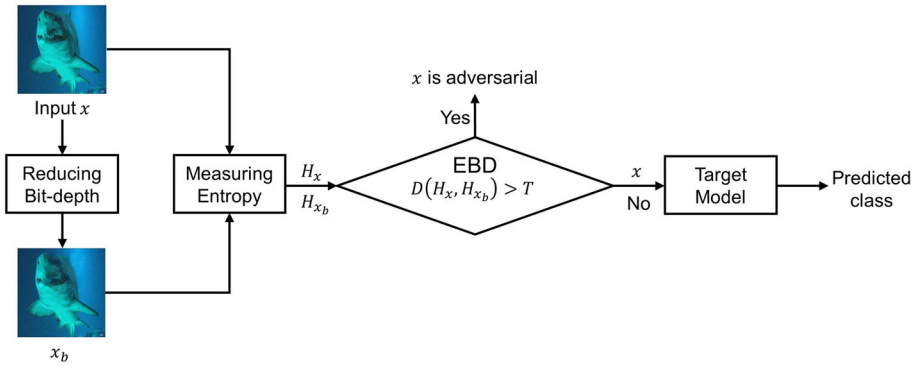


**Fig. 1** Illustration of the proposed entropy-based detector

To increase the robustness of classifiers, Bradshaw et al. [44] proposed Gaussian-processes hybrid DNNs (GPDNNs), which express the latent variables as a Gaussian distribution parameterized by the functions of the mean and covariance and encodes them with radial basis function kernels. Jeddi et al. [36] introduced the Learn2Perturb framework, an end-to-end feature perturbation learning approach to improve the adversarial robustness of DNNs. Pang et al. [48] proposed Max-Mahalanobis center loss to explicitly induce dense feature regions to maximize robustness. Xiao et al. [49] introduced k-winners-take-all activation that can be readily used in nearly all existing networks and training methods that do not possess a significant overhead.

## 3 Proposed method

In this section, we introduce the entropy-based detector (EBD) for defending against adversarial attacks. We will demonstrate that the difference in entropy between an original sample and a sample with reduced bit-depth in Sect. 4.2. This enables adversarial examples to be detected using the EBD. The EBD identifies an input sample as adversarial if the difference in entropy of the input sample before and after bit-depth reduction is larger than a threshold. If the input sample is not identified as adversarial, it is predicted by the target model. Figure 1 represents an illustration of the proposed EBD, which will be explained in more detail in the following section.

### 3.1 Overview

The EBD is based on two main ideas. First, a greater perturbation with different values on the adversarial example will lead to a higher measured entropy in adversarial examples. Adversaries generate adversarial examples by adding perturbations with different values or small values to minimize the distance between the original samples and the adversarial examples. Adversarial examples are represented using more than 256 intensity values for all color channels, whereas normal images are represented using 256 intensity values for all color channels. The occurrence probability of each intensity value decreases as the number of intensity values increases. This means that the distribution of intensity values approaches a uniform distribution, and the entropy of the adversarial examples is high. We will show that the average entropy of adversarial examples generated by each attack method is larger than that of the normal samples in Sect. 4.2.1.

Second, the entropy of the image after bit-depth reduction decreases. An original color image has three color channels (red, green, and blue), and all color channels represent the original color image using $2^8 = 256$ intensity values, e.g., [0, 1/255, 2/255, ..., 1]. If we reduce the bit depth to six bits, the distribution of the image becomes far from the uniform distribution because all of the color channels represent the image using $2^6 = 64$ intensity values, e.g., [0, 1/63, 2/63, ..., 1]. We will show that the entropy of the image decreases as the bit depth decreases in Sect. 4.2.2. Thus, the changes in the entropy of the legitimate samples after the bit depth has been reduced are small, whereas those of the adversarial examples after bit-depth reduction are large.

### 3.2 Entropy-based detector

To detect adversarial examples, the EBD first measures the occurrence probabilities for all intensity values as follows.

$$p_i = \frac{\text{Number of occurrences of the intensity value } i}{\text{Number of pixels of image} \times 3} \tag{1}$$

We measure the number of occurrences of each intensity value and then calculate the occurrence probabilities of each intensity value, divided by three times the number of pixels in the image. We do this because the image consists of three color channels: red, green, and blue. The EBD then calculates the entropy of the input image as follows.

$$H = -\sum_i p_i \log_2 p_i, \tag{2}$$

**Algorithm 1** Entropy-Based Detector

**Require:** input sample $x$, threshold $T$, bit-depth $b$.
**EBD:**
$H_x \leftarrow \text{Entropy}\,(x)$
$x_b \leftarrow \frac{\text{round}(x \times (2^b - 1))}{2^b - 1}$
$H_{x_b} \leftarrow \text{Entropy}\,(x_b)$
$D\left(H_x, H_{x_b}\right) \leftarrow \left|H_x - H_{x_b}\right|$
**if** $D\left(H_x, H_{x_b}\right) > T$ **then**
  **return** $x$ is adversarial
**else**
  **return** $x$ is normal
**end if**

where $p_i$ represents the occurrence probability of the $i$-th intensity value and $p_i$ is calculated as follows. To reduce the bit depth of the input image, The EBD first multiplies the input image by $2^b - 1$ and then rounds the result to an integer value. The EBD then divides by $2^b - 1$ to return the values to the range [0, 1], as follows.

$$x_b = \frac{\text{round}\left(x \times \left(2^b - 1\right)\right)}{2^b - 1}, \tag{3}$$

where $x$ is the original input image, $b$ is a bit depth, and round represents the rounding function, which rounds the intensity value to the nearest integer value. The EBD calculates the difference between the entropy of the input image and the entropy of the image with reduced bit depth as follows.

$$D\left(H_x, H_{x_b}\right) = \left|H_x - H_{x_b}\right|, \tag{4}$$

where $H_x$ is the entropy of the original input image $x$ and $H_{x_b}$ is the entropy of image $x_b$, which has a bit depth of $b$ bits. The EBD compares $D\left(H_x, H_{x_b}\right)$ with a predefined threshold $T$. If $D\left(H_x, H_{x_b}\right)$ is more than $T$, the input image $x$ is an adversarial example; otherwise, it is a legitimate sample. The procedure of the EBD is described in detail in Algorithm 1. Entropy () in Algorithm 1 represents Eq. 2. Note that purpose of the EBD is only to detect adversarial examples.

## 4 Experiments

### 4.1 Crafting adversarial examples

#### 4.1.1 Setting of adversarial attack

To generate adversarial examples, we used a vanilla CNN comprising three convolutional layers followed by two fully connected layers on the MNIST dataset [60], VGG16 [55], ResNet20 [1] on the CIFAR-10 dataset [61], and pre-trained ResNet152v2 [38] and Inception-ResNetv2 [39] on the ImageNet dataset as target models. We used the Adam optimizer to train the target model for the MNIST dataset and the stochastic gradient descent optimizer to train the target models for the CIFAR-10 dataset. Additionally, we applied a learning rate scheduler to train the target models, and we set the initial learning rate to 0.01. Table 1 shows the classification performance of the target models on the respective datasets. To analyze and detect adversarial examples, we generated adversarial examples for five different general attacks: FGSM [19], BIM [20], MIM [28], DeepFool [6], and CW [21] attacks. We generated adversarial examples for both targeted and untargeted attacks. In a targeted attack, the target model misclassifies the adversarial example as a target class chosen by the adversary. In an untargeted attack, the target model misclassifies the adversarial example as any class other than the correct class. For the targeted attack, we employed two different targets: 1) the next class setting ("Next" in the results), in which the targeted class is the next class with respect to the original class (e.g., misclassifying an input sample of the digit "5" as a "6") and 2) the least-likely (LL) class setting, in which the target class is the most dissimilar to the original class (e.g., misclassifying a "1" as an "8"). We performed untargeted FGSM, BIM, MIM, and DeepFool attacks as well as targeted CW attacks. We set the adversaries to generate adversarial perturbations within a range of $\epsilon = 0.1$ and $\epsilon = 0.3$ on the MNIST dataset and a range of $\epsilon = 0.01$ and $\epsilon = 0.03$ on the CIFAR-10 and ImageNet datasets for the FGSM, BIM, and MIM attacks (e.g., labeled as FGSM (0.01), FGSM (0.03), BIM (0.01), BIM (0.03), MIM (0.01), and MIM (0.03)). In addition, we set the step size $\alpha$ of the adversaries to be one-tenth of each $\epsilon$ with 10 attack iterations for the BIM attack. For the DeepFool attack, we set the number of candidate classes to 10 and the maximum number of iterations to 50. We used the implementations of the FGSM, BIM, MIM, and DeepFool attacks provided by the CleverHans library [63]. For the CW attack, we used the implementation from the original authors [21]. We used two PCs, each equipped with an i7-6700K 4 GHz CPU, 64 GB of RAM, and two NVIDIA Titan XP GPUs, to conduct the experiments.

**Table 1** Performance of the target models

| | Target model | Accuracy |
|---|---|---|
| MNIST | Vanilla CNN | 99.0% |
| CIFAR-10 | VGG16 | 90.2% |
| | ResNet20 | 89.7% |
| ImageNet | ResNet152v2 | 78.0% |
| | Inception-ResNetv2 | 80.3% |

#### 4.1.2 Results of adversarial attack

Table 2 reports the attack results from our evaluation on the MNIST, CIFAR-10, and ImageNet datasets. For the FGSM,

**Table 2** Results of adversarial attacks

| Dataset | Target model | Configuration Attack | Mode | Success rate | Prediction confidence | Distortion |
|---|---|---|---|---|---|---|
| MNIST | Vanilla CNN | FGSM | 0.10 | 19.2% | 0.7874 | 1.8018 |
| | | | 0.30 | 69.8% | 0.6161 | 5.2971 |
| | | BIM | 0.10 | 61.4% | 0.8365 | 1.1943 |
| | | | 0.30 | 100.0% | 0.9483 | 2.7769 |
| | | MIM | 0.10 | 56.8% | 0.7989 | 1.6820 |
| | | | 0.30 | 100% | 0.9193 | 4.7600 |
| | | DeepFool | | 99.2% | 0.4973 | 0.7906 |
| | | CW | Next | 95.4% | 0.3206 | 0.9956 |
| | | | LL | 93.6% | 0.2624 | 1.1076 |
| CIFAR-10 | VGG16 | FGSM | 0.01 | 35.0% | 0.9576 | 0.5522 |
| | | | 0.03 | 41.4% | 0.9564 | 1.6510 |
| | | BIM | 0.01 | 43.2% | 0.9887 | 0.4123 |
| | | | 0.03 | 57.6% | 0.9984 | 1.0083 |
| | | MIM | 0.01 | 41.0% | 0.9826 | 0.4883 |
| | | | 0.03 | 51.2% | 0.9997 | 1.3184 |
| | | DeepFool | | 100.0% | 0.6374 | 0.3331 |
| | | CW | Next | 99.4% | 0.3844 | 0.3974 |
| | | | LL | 100.0% | 0.3452 | 0.4336 |
| | ResNet20 | FGSM | 0.01 | 89.2% | 0.8924 | 0.5512 |
| | | | 0.03 | 94.40% | 0.8542 | 1.6476 |
| | | BIM | 0.01 | 97.20% | 0.9647 | 0.3889 |
| | | | 0.03 | 98.80% | 0.9792 | 0.8897 |
| | | MIM | 0.01 | 97.00% | 0.9570 | 0.4778 |
| | | | 0.03 | 99.00% | 0.9780 | 1.2917 |
| | | DeepFool | | 100.0% | 0.5170 | 0.1064 |
| | | CW | Next | 100.0% | 0.3723 | 0.1419 |
| | | | LL | 100.0% | 0.2995 | 0.1797 |
| ImageNet | ResNet152v2 | FGSM | 0.01 | 85.0% | 0.7602 | 3.8374 |
| | | | 0.03 | 88.5% | 0.7402 | 11.4514 |
| | | BIM | 0.01 | 99.0% | 0.9952 | 3.3635 |
| | | | 0.03 | 99.50% | 0.9992 | 8.9754 |
| | | MIM | 0.01 | 97.0% | 0.9520 | 3.0298 |
| | | | 0.03 | 98.0% | 0.9911 | 8.5041 |
| | | DeepFool | | 98.5% | 0.4827 | 0.3074 |
| | | CW | Next | 100.0% | 0.2563 | 0.6182 |
| | | | LL | 100.0% | 0.1624 | 0.8330 |
| | Inception- ResNetv2 | FGSM | 0.01 | 50.0% | 0.6709 | 5.1229 |
| | | | 0.03 | 57.5% | 0.6394 | 15.2786 |
| | | BIM | 0.01 | 99.0% | 0.9946 | 4.3355 |
| | | | 0.03 | 100.0% | 0.9999 | 11.3639 |
| | | MIM | 0.01 | 78.0% | 0.8386 | 4.0040 |
| | | | 0.03 | 92.0% | 0.9298 | 11.3884 |
| | | DeepFool | | 96.0% | 0.4583 | 1.1230 |
| | | CW | Next | 100.0% | 0.3079 | 0.9289 |
| | | | LL | 98.50% | 0.2317 | 1.1164 |

BIM, MIM, and DeepFool attacks, the reported attack success rate is $1 - accuracy$. For the FGSM attacks, the attack success rate against ResNet20 is higher than that against VGG16 on the CIFAR-10 dataset, and the attack success rate against ResNet152v2 is higher than that against Inception-ResNetv2 on the ImageNet dataset. The prediction confidence of FGSM (0.03) is slightly lower than that of FGSM (0.01) on the CIFAR-10 and ImageNet datasets. For the BIM attacks, the attack success rate of BIM (0.03) is higher than that of BIM (0.01) against VGG16, and the attack success rate of BIM (0.01) is similar to that of BIM (0.03) against ResNet20 on the CIFAR-10 dataset. In addition, the attack success rate of BIM (0.01) is similar to that of BIM (0.03) against both ResNet152v2 and Inception-ResNetv2 on the ImageNet dataset. The average distortion for BIM (0.03) is higher than that for BIM (0.01) for all DNNs on the CIFAR-10 and ImageNet datasets, and the prediction confidence is higher than 0.99 on the ImageNet dataset. In addition, the attack success rates for the BIM attacks are higher than those for the FGSM attacks for all DNNs on the all datasets. The attack success rate, prediction confidence, and distortion of the MIM attacks tended to be similar to those of the BIM attacks because the BIM and MIM attack methods both transform the FGSM attacks. The attack success rate of the DeepFool attacks is higher than that of the FGSM, BIM, and MIM attacks for some DNNs on the all datasets. In addition, the distortion of the DeepFool attacks is lower than that of the FGSM, BIM, and MIM attacks for all DNNs on the all datasets. A CW attack is only considered a success if the target model predicts the targeted class. The attack success rates of Next and LL are almost 100% for all target models on the CIFAR-10 and ImageNet datasets. The distortion of Next is lower than that of LL on the ImageNet dataset. Next is an easier targeted attack than LL because the next class on the ImageNet dataset is usually similar to the current class. The prediction confidence and distortions for the CW attacks are substantially lower than those of the FGSM, BIM, and MIM attacks against all target models on both datasets. This means that a CW attack is a much more sophisticated attack than the FGSM, BIM, and MIM attacks.

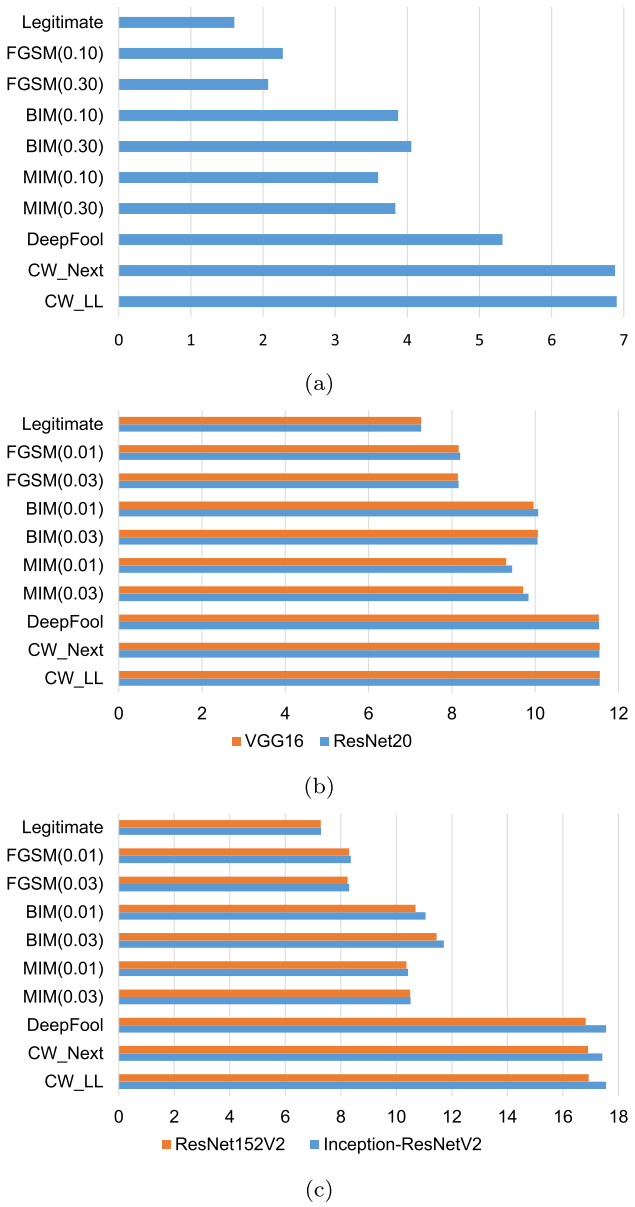


**Fig. 2** Average entropy of legitimate samples and adversarial examples generated by the FGSM, BIM, MIM, DeepFool, and CW attacks for **a** vanilla CNN on the MNIST dataset; **b** VGG16 and ResNet20 on the CIFAR-10 dataset and **c** ResNet152v2 and Inception-ResNetv2 on the ImageNet dataset

## 4.2 Entropy analysis

### 4.2.1 Entropy comparison for legitimate sample and adversarial example

We analyze the entropy of legitimate samples and adversarial examples. To analyze this entropy, we only use the successful adversarial examples generated in 4.1.2 and assume that the input samples are on the numerical scale [0, 1]. We compare the average entropy of a validation set from the ImageNet dataset with that of the adversarial examples generated by each attack method. Figure 2 shows the average entropy of the legitimate samples and adversarial examples. The average entropies of the legitimate samples are 1.6025 for vanilla CNN on the MNIST dataset and 7.2629 for VGG16 and 7.2629 for ResNet20 on the CIFAR-10 dataset, and 7.2775 for ResNet152v2 and 7.2871 for Inception-ResNetv2 on the ImageNet dataset.

For the FGSM attack, the average entropies are 2.2735 when $\epsilon = 0.1$ and 2.0711 when $\epsilon = 0.3$ for vanilla CNN on the MNIST dataset, and the average entropies are 8.1639 and 8.2003 when $\epsilon = 0.01$, and 8.1431 and 8.1610 when

$\epsilon = 0.03$ for VGG16 and ResNet20 on the CIFAR-10 dataset, respectively. Moreover, the average entropies are 8.2988 and 8.3592 when $\epsilon = 0.01$, and 8.2447 and 8.3013 when $\epsilon = 0.03$ for ResNet152v2 and Inception-ResNetv2 on the ImageNet dataset. The average entropies of the adversarial examples when $\epsilon = 0.01$ are slightly higher than when $\epsilon = 0.03$. In addition, the difference in the average entropies of the legitimate samples and adversarial examples are less than 0.4686 for the MNIST dataset, 0.9374 for the CIFAR-10 dataset and 1.0721 for the ImageNet dataset, respectively.

For the BIM attack, the average entropies are 3.8723 $\epsilon = 0.1$ and 4.0547 $\epsilon = 0.3$ for vanilla CNN on the MNIST dataset, and the average entropies are 9.9622 and 10.0733 when $\epsilon = 0.01$, and 10.065 and 10.0594 when $\epsilon = 0.03$ for VGG16 and ResNet20 on the CIFAR-10 dataset, respectively. The average entropies are 10.6928 and 11.0531 when $\epsilon = 0.01$, and 11.449 and 11.7107 when $\epsilon = 0.03$ for ResNet152v2 and Inception-ResNetv2 on the ImageNet dataset. The average entropies of the adversarial examples when $\epsilon = 0.03$ are slightly higher than when $\epsilon = 0.01$. In addition, the differences in the average entropies of the legitimate samples and adversarial examples for the BIM attacks are less than 2.8104 and 4.4236 for both datasets, respectively.

For the MIM attack, the average entropies are 3.5958 when $\epsilon = 0.1$ and 3.8334 when $\epsilon = 0.3$ for vanilla CNN on the MNIST dataset, and the average entropies are 9.3055 and 9.4482 when $\epsilon = 0.01$, and 9.7118 and 9.8409 when $\epsilon = 0.03$ for VGG16 and ResNet20 on the CIFAR-10 dataset, respectively. Moreover, the average entropies are 10.3630 and 10.4200 when $\epsilon = 0.01$, and 10.4989 and 10.5132 when $\epsilon = 0.03$ for ResNet152v2 and Inception-ResNetv2 on the ImageNet dataset. The average entropies of the adversarial examples generated by the MIM attack when $\epsilon = 0.03$ are also slightly higher than when $\epsilon = 0.01$. In addition, the differences in the average entropies of the legitimate samples and adversarial examples for the BIM attacks are less than 2.2309, 2.5780 and 3.2261 for all datasets, respectively. All of the average entropies for the MIM attacks are higher than those for the FGSM attacks. The reason why the average entropies for the BIM and MIM attacks are higher than those for the FGSM attacks is that BIM and MIM attacks generate adversarial examples using 10 attack iterations.

For the DeepFool attack, the average entropy is 5.3188 for vanilla CNN on the MNIST dataset, and the average entropies are 11.5291 and 11.5348 for VGG16 and ResNet20 on the CIFAR-10 dataset, respectively. The average entropies are 16.8208 and 17.5579 for ResNet152v2 and Inception-ResNetv2 on the ImageNet dataset, respectively. In addition, the differences in the average entropies of the legitimate samples and adversarial examples for the DeepFool attacks are less than 3.7163, 4.2719 and 10.2708 for all datasets, respectively. All the average entropies for the DeepFool attacks are higher than those for the FGSM, BIM, and MIM attacks because the DeepFool attack is based on the $L2$-norm.

For the CW attack, the average entropies are 6.8790 when the target class is Next and 6.9031 when the target class is LL for vanilla CNN on the MNIST dataset, and the average entropies are 11.5509 and 11.5442 when the target class is Next, and 11.5542 and 11.5508 when the target class is LL for VGG16 and ResNet20 on the CIFAR-10 dataset, respectively. The average entropies are 17.1326 and 17.4188 when the target class is Next and 17.1483 and 17.557 when the target class is LL for ResNet152v2 and Inception-ResNetv2 on the ImageNet dataset, respectively. The average entropies of the adversarial examples are similar for both target classes, Next and LL. The average entropy for a CW attack is more than twice the average entropy of the legitimate samples. One reason why the average entropy for the CW attacks is higher than that of the FGSM, BIM, and MIM attacks is that the CW attack uses a high number of attack iterations to generate adversarial examples. The average entropy for the CW attacks is similar to that for the DeepFool attacks because they are both based on the $L2$-norm. Moreover, the adversarial examples include perturbations with different values to minimize the difference between the original samples and the adversarial examples. In other words, the adversarial examples generated by a CW attack have different intensity values for each color channel.

#### 4.2.2 Comparison of entropy before and after reducing bit depth

To determine the effect of reducing bit depth, we compare the average entropy before and after reducing the bit depth using the adversarial examples generated in Sect. 4.1.2. We adjusted the bit depth reduction from a reduction of three bits (eight bits to five bits) to a reduction of one bit (eight bits to seven bits) to investigate the change in entropy according to bit depth. Figure 3 shows the average entropy of the legitimate samples and adversarial examples before and after bit-depth reduction. As the bit depth gradually decreases, the average entropy of the legitimate samples and that of the adversarial examples also decreases. In addition, all the average entropies of the legitimate samples and adversarial examples become similar when the bit depth is reduced. This is because both the legitimate samples and adversarial examples are represented by the same number of intensity values for all color channels. When bit depth is reduced, the entropy of the legitimate samples decreases slightly, but the decrease in the entropy of the adversarial examples is larger. Thus, the difference in the entropy of the legitimate samples before and after bit-depth reduction is smaller than that of the adversarial examples generated by the FGSM, BIM, MIM, DeepFool, and CW attacks.

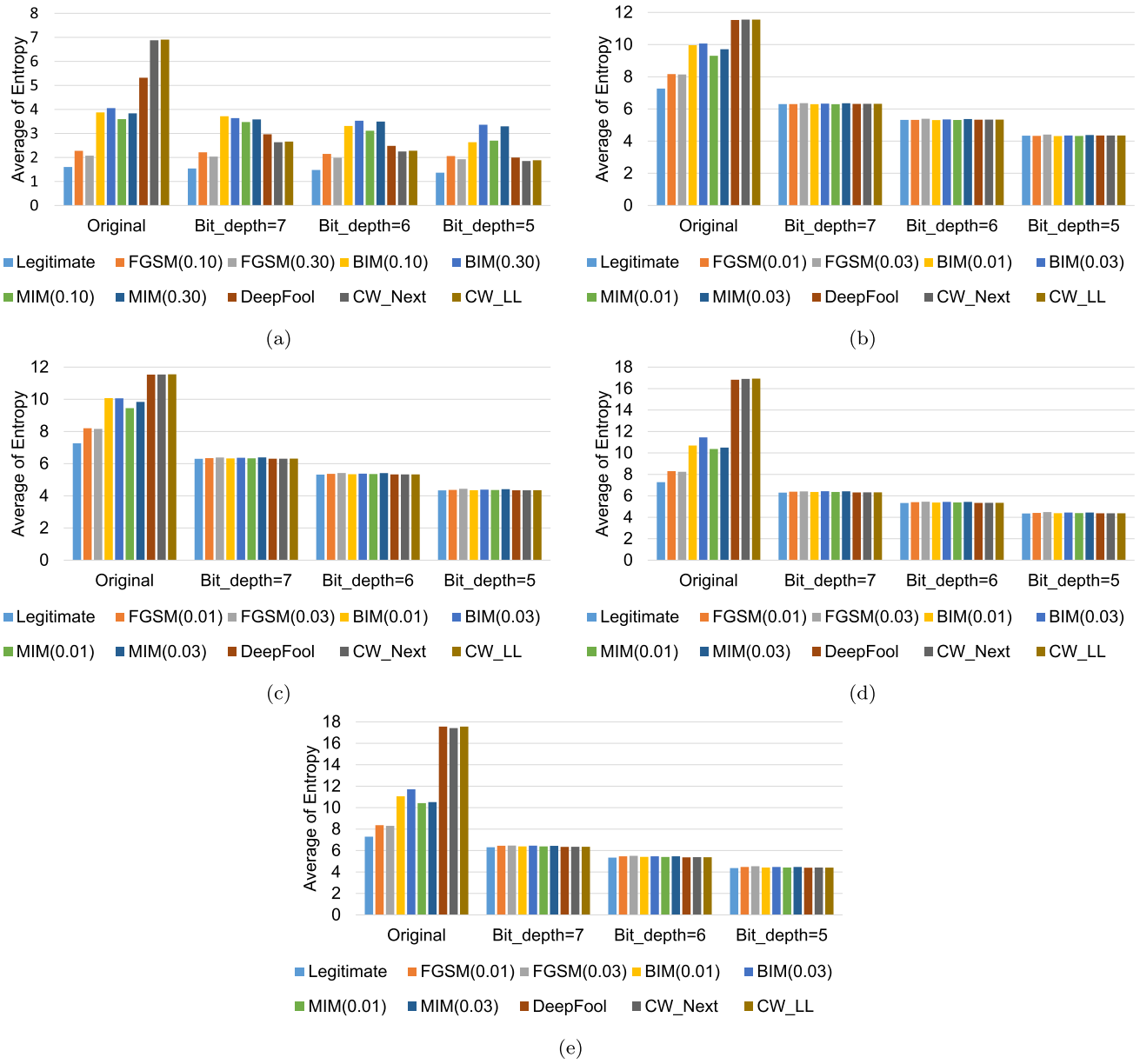


**Fig. 3** Comparison of average entropy of general adversarial examples before and after bit-depth reduction for **a** vanilla CNN on the MNIST dataset; **b** VGG16 and **c** ResNet20 on the CIFAR-10 dataset; **d** ResNet152v2 and **e** Inception-ResNetv2 on the ImageNet dataset

## 4.3 Detecting adversarial examples

### 4.3.1 Selecting the optimal threshold

To evaluate the performance of the EBD, we must select an optimal threshold. A defender must select the threshold only using the benign samples that they have because the defender does not know whether the input sample of the model is normal or adversarial, and they do not know that the adversarial examples inputted into the model were generated by which attack method. Therefore, we only used legitimate samples to select the optimal threshold, so the selected threshold does not depend on adversarial examples. In addition, the defender must select a threshold with a low false positive rate because a detector with a high false positive rate would not be ideal for many security-sensitive applications. Thus, the low false positive rate indicates that the detector is unlikely to misdetect a legitimate sample inputted into the model as an adversarial. However, the lower the false positive rate is, the higher the threshold, which ultimately causes a decrease in the detection rate. Therefore, it is important to select the threshold at which the detector detects adversarial examples with a low false positive rate.

To select the optimal threshold, we selected all the samples in the MNIST and CIFAR-10 datasets for the training set and randomly selected an additional 10 samples per class for the training set from the ImageNet dataset. Then, we measured the differences in entropy before and after bit-depth reduction for the selected legitimate samples. We selected the optimal threshold to minimize the false positive rate and maximize the detection rate. The feature squeezing [15] and trapdoor [58] selected the threshold by setting the false positive rate as 5.0%, but we set the false positive rates as 2.5% and 5.0% for all bit-depths from five to seven bits to compare the detection rate according to the false positive rate. We then set the entropy differences, which are the points of the false positive rates, as the thresholds. For the MNIST dataset, the selected thresholds for the false positive rate of 2.5% are 0.4047, 0.2392, and 0.1531 for bit depths of five, six, and seven bits, respectively. For the CIFAR-10 dataset, the selected thresholds for the false positive rate of 2.5% are 2.9968, 1.9884, and 0.9803 for both DNN architectures for bit depths of five, six, and seven bits, respectively. The selected thresholds for the false positive rate of 5.0% are 2.9937, 1.9847, 0.9786 for both DNN architectures for bit depths of five, six, and seven bits, respectively. For the ImageNet dataset, the selected thresholds for the false positive rate of 2.5% are 3.0355, 2.0191, and 0.9992 for ResNet152v2, and 3.0367, 2.0192, and 0.9995 for Inception-ResNetv2 for bit depths of five, six, and seven bits, respectively. The selected thresholds for the false positive rate of 5.0% are 3.0344, 2.0170, and 0.9991 for ResNet152v2, and 3.0352, 2.0175, and 0.9994 for Inception-ResNetv2 for bit depths of five, six, and seven bits, respectively. We selected the false positive rate and the threshold after evaluating the detection performance for each bit depth and threshold in Sect. 4.3.2.

#### 4.3.2 Selecting the optimal bit depth

The defender must select an optimal bit depth that can well detect any adversarial examples. To select the optimal bit depth, we performed the experiments while adjusting the bit depth from five to seven bits for the adversarial examples generated by the FGSM, BIM, MIM, DeepFool, and CW attacks in Sect. 4.1.2. In our experiments, we fixed the thresholds selected for each bit depth and assumed that the defender did not know that adversaries generate adversarial examples by using any attack method. For the FGSM attack, the detection rates for the adversarial examples increased as the bit depth increased. The difference in the entropy of the legitimate samples before and after bit-depth reduction decreases as the bit depth increases. The selected threshold is therefore a low value. For the MNIST dataset, the EBD cannot detect the most adversarial examples generated by the FGSM attack and the EBD best detects the adversarial examples generated by other adversarial attacks when the bit depth is reduced to six bits. The reason why the EBD cannot detect the most adversarial examples generated by the FGSM attack is that the MNIST dataset contains many black and white colors. Therefore, new intensity values are not created by FGSM attacks. When the bit depth is reduced to seven bits, the EBD detects more than 98% of the adversarial examples generated by the FGSM attack for the CIFAR-10 and ImageNet datasets. For the BIM, MIM, DeepFool, and CW attacks, the EBD detects nearly 100% of the adversarial examples for all bit depths and for the CIFAR-10 and ImageNet datasets. This is because the changes in entropy of the adversarial examples generated by the BIM, MIM, DeepFool, and CW attacks before and after bit-depth reduction are larger than those of the legitimate samples before and after bit-depth reduction. When the bit depth is seven bits, the EBD best detects the adversarial examples generated by all attack methods. In addition, the detection rates when the false positive rate is 2.5% were the same as those for all bit depths when the false positive rate is 5.0% because the thresholds when the false positive rate is 2.5% are not significantly different from the thresholds when the false positive rate is 5.0%. Therefore, we hence selected the optimal bit depth to be seven bits and the threshold when the false positive rate is 2.5%, and fixed the bit depth and the threshold in subsequent experiments. Figure 4 shows the detection rates for adversarial examples for each attack method using bit depths from five to seven bits when the false positive rate is 2.5%.

Figure 5 shows the distribution of the differences in the entropies of legitimate samples and adversarial examples generated by the FGSM, BIM, and CW attacks before and after the bit depth has been reduced to five, six, and seven bits. The change in entropy of the legitimate samples decreases as the bit depth is reduced.

### 4.4 Comparison with previous work

We only compared the detection performance of the EBD with that of feature squeezing because pixel deflection and NRP are not detection methods. To compare the detection performance, we used the adversarial examples generated for each attack method in Sect. 4.1.2 and used the best parameters for feature squeezing [15], trapdoor [58], and EBD. Table 3 shows the detection rates of each detector for the FGSM, BIM, MIM, DeepFool, and CW attacks when the false positive rate is 2.5%. Feature squeezing detects fewer than 3% of the adversarial examples generated by the FGSM attack and fewer than 5% of the adversarial examples generated by the BIM and MIM attacks. In addition, feature squeezing does not detect the adversarial examples generated by the DeepFool and CW attacks. Trapdoor detects more than 80% of the adversarial examples generated by the FGSM attack and more than 99% of the adversarial examples generated by the BIM and MIM attacks. In addition, trapdoor detects 86.92%

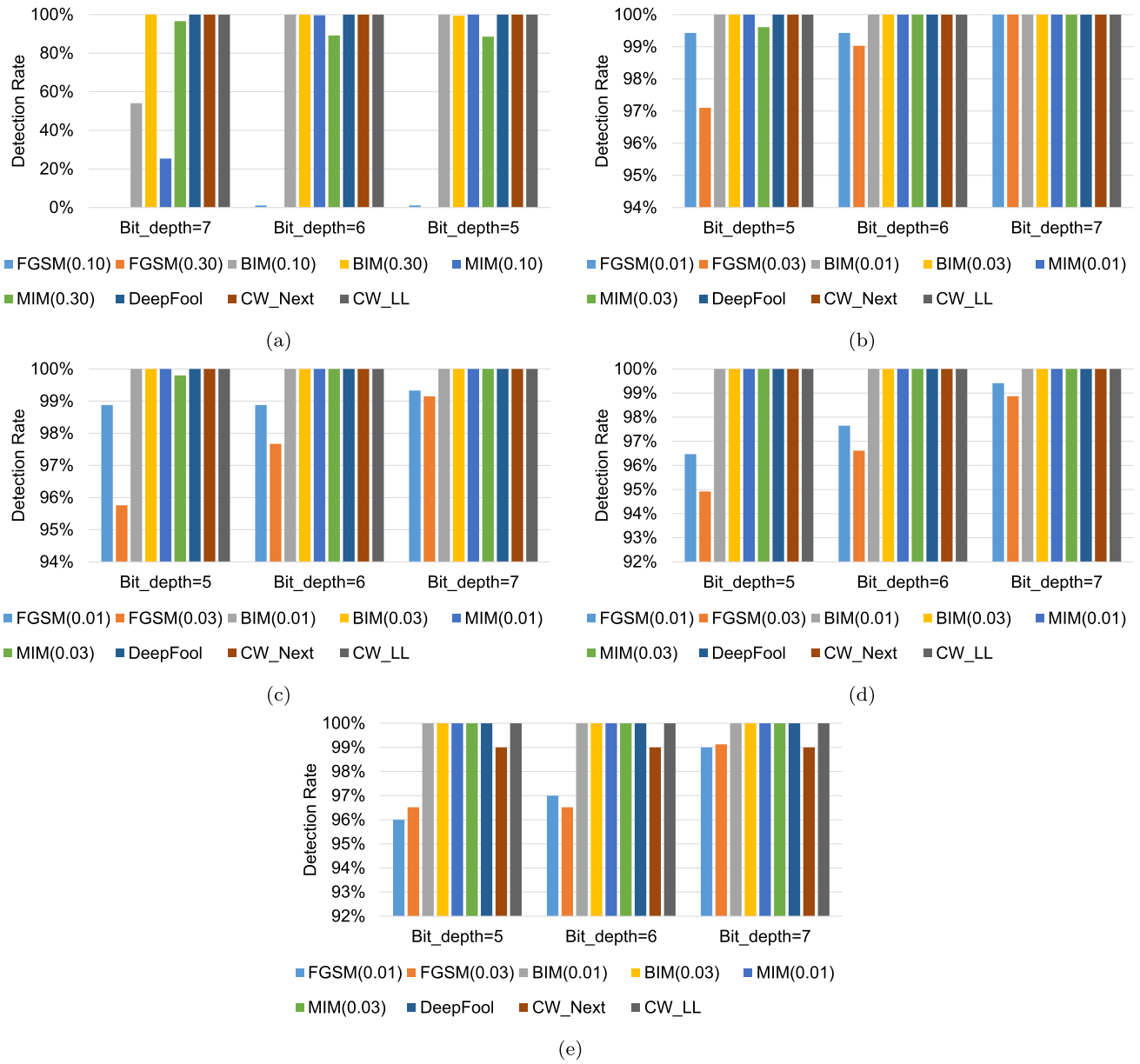


**Fig. 4** Detection rate for adversarial examples generated by the FGSM, BIM, MIM, DeepFool, and CW attacks for **a** vanilla CNN on the MNIST dataset; **b** VGG16 and **c** ResNet20 on the CIFAR-10 dataset; **d** ResNet152v2 and **e** Inception-ResNetv2 on the ImageNet dataset

of the adversarial examples generated by the DeepFool attack and 85.32% and 98.80% of the adversarial examples generated by the CW attacks when the target classes are Next and LL, respectively. EBD detects more than 99% of the adversarial examples generated by the FGSM, BIM, MIM, DeepFool, and CW attacks. EBD achieves higher detection rates than feature squeezing and trapdoor against all adversarial attack methods. The elapsed times were also measured as average 0.0004 s, average 0.0282 s, and average 1.5083 s per sample on the CIFAR-10 datasets for the EBD, feature squeezing, and trapdoor, respectively.

## 5 Discussion

### 5.1 Justification for using bit-depth reduction

In EBD, we use bit-depth reduction to detect adversarial examples according to the entropy difference. However, we considered other methods, such as median smoothing, non-local smoothing, pixel deflection, and NRP. Figure 6 shows the entropies of a normal sample and those of adversarial examples before and after applying bit-depth reduction, median smoothing, and non-local smoothing. Median smoothing runs a sliding window over each pixel of the

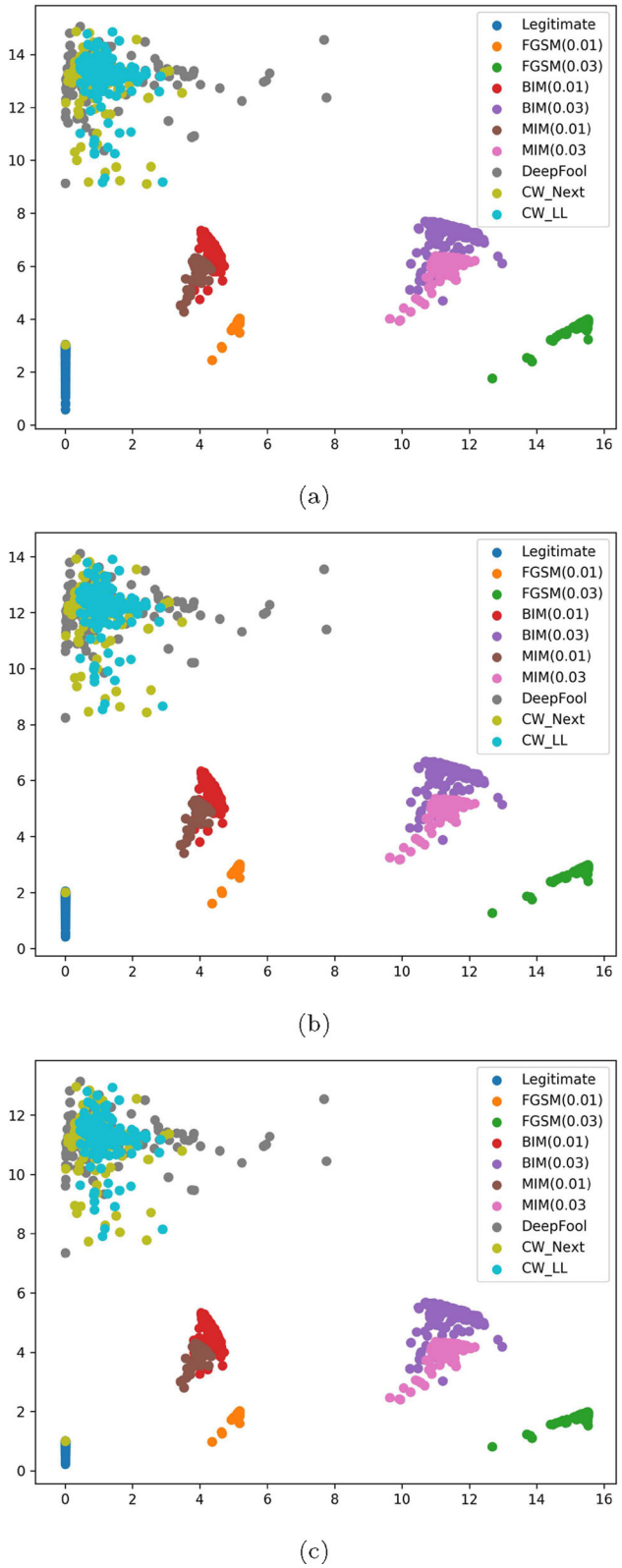


**Fig. 5** Difference in the entropy of legitimate samples and adversarial examples generated by the FGSM, BIM, MIM, and CW attacks for Inception-ResNetv2 on the ImageNet dataset before and after reducing the bit depth to **a** five bits, **b** six bits, and **c** seven bits

**Table 3** Comparison of the results of feature squeezing and trapdoor for ResNet20 on the CIFAR-10 dataset

| Attack method | Defense method | | |
|---|---|---|---|
| | Feature squeezing | Trapdoor | EBD |
| FGSM (0.01) | 2.50% | 80.13% | 99.33% |
| FGSM(0.03) | 0.89% | 89.01% | 99.15% |
| BIM (0.01) | 4.0% | 100.0% | 100.0% |
| BIM (0.03) | 4.67% | 99.80% | 100.0% |
| MIM (0.01) | 2.89% | 99.60% | 100.0% |
| MIM (0.03) | 4.0% | 99.80% | 100.0% |
| DeepFool | 0.0% | 86.92% | 100.0% |
| CW_Next | 0.0% | 85.37% | 100.0% |
| CW_LL | 0.0% | 98.80% | 100.0% |

image, where the center pixel is replaced by the median value of the neighboring pixels within the window. Therefore, the distribution of the intensity values after applying median smoothing is not much different from that before applying median smoothing. Therefore, it is not suitable for EBD.

Non-local smoothing takes the mean of all pixels in the image, weighted by how similar those pixels are to the target pixel. Non-local smoothing is an effective way to remove perturbations in adversarial examples, but it also changes normal images substantially. In addition, the entropy after applying non-local smoothing increases because the distribution of the intensity values after applying non-local smoothing approaches a uniform distribution.

As for pixel deflection and NRP, the entropies of the input sample after applying these methods are higher than those of the original input sample. The distribution of the intensity values approaches a uniform distribution because the input sample is represented by many more intensity values. Therefore, applying non-local smoothing, pixel deflection, and NRP instead of bit-depth reduction in EBD enables it to detect adversarial examples if the entropy difference before and after applying non-local smoothing is lower than the predefined threshold; otherwise, it is a legitimate sample. We conducted experiments with EBD in which we applied non-local smoothing, pixel deflection, and NRP instead of bit-depth reduction, but they resulted in lower detection performances of adversarial examples than when bit-depth reduction is applied in EBD.

By contrast, when the bit depth is reduced, a normal sample and an adversarial example are both represented by a fixed number of intensity values in all color channels. Figure 7 shows histograms of a normal sample and those of adversarial examples before and after the bit depth has been reduced to seven bits. The distributions of the adversarial examples generated by the FGSM attack look similar to those of the normal sample. However, new intensity values are created

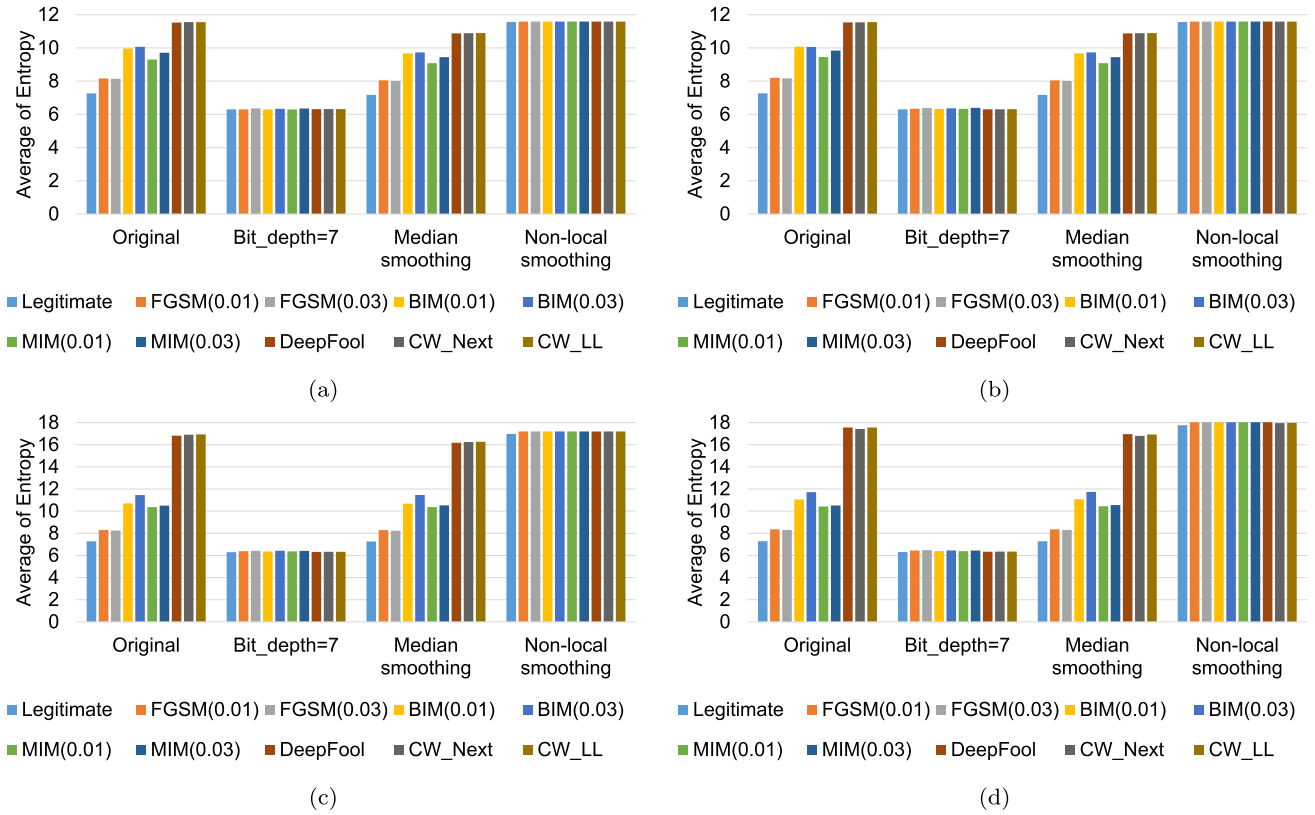


**Fig. 6** Average entropy of legitimate samples and adversarial examples generated by each adversarial attack before and after applying bit-depth reduction, median smoothing, and non-local smoothing for **a** VGG16 and **b** ResNet20 on the CIFAR-10 dataset; **c** ResNet152v2 and **d** Inception-ResNetv2 on the ImageNet dataset

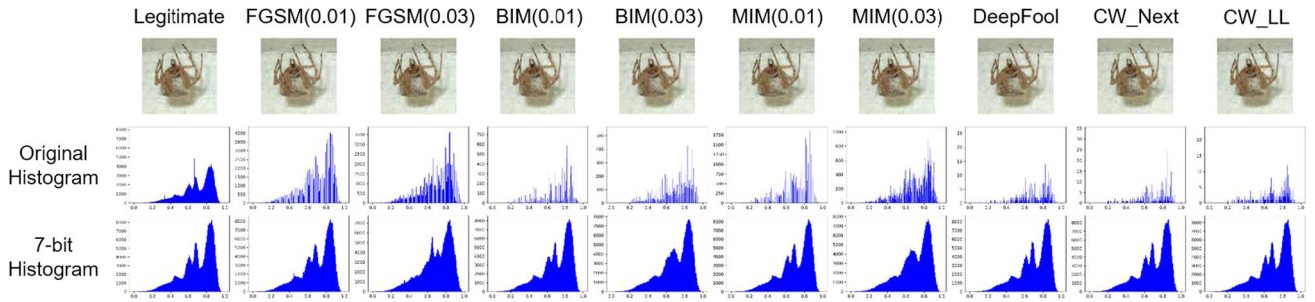


**Fig. 7** Histogram of a legitimate sample and the adversarial examples generated by each attack method for Inception-ResNetv2

by the perturbations in the adversarial examples, and they are shown as low-frequency components in the histogram. The adversarial examples generated by the BIM and MIM attacks have more new intensity values than the adversarial examples generated by the FGSM attack because the BIM and MIM attacks generate adversarial examples by adding perturbations to the original image over multiple iterations. The adversarial examples generated by the DeepFool attack have many new intensity values because it is based on the $L2$-norm. Moreover, a CW attack generates adversarial examples by adding a perturbation with very small and different values to minimize the difference between the original sample and the adversarial example. Therefore, the adversarial examples generated by the CW attack cause most of the intensity values of all pixels in the color channels to have different values. When the bit depth is reduced, the distribution of the adversarial examples becomes similar to that of a normal sample. In addition, the change in the distribution of the normal samples is small, whereas the change in those of the adversarial examples is larger. Hence, bit-depth reduction is a suitable method for EBD.

### 5.2 Limitations and robustness of EBD against adaptive attacks

Thus far, we have only considered adversaries who do not know EBD directly. Here, we consider an adaptive attack in which the adversary not only fully knows the details of the target model but also has information on the defense method used to defend the target model against various adversarial attacks. The limitation of our detection method is that it cannot detect adversarial examples in which the adversaries determine an input sample in which the change in entropy of the input sample after reducing the bit depth is lower than the detection threshold. To evade our detection method, an adversary must generate the adversarial example by adding perturbations using only the original intensity values existing in the images, such as 1/255, 4/255, and 8/255. To evaluate EBD in this attack setting, we generated adversarial examples by setting $\epsilon$ to 4/255 and 8/255 in the FGSM, BIM, and MIM attacks and performed the BIM attack by setting the step size to 1/255. EBD was able to detect more than 90% of these adversarial examples. The adversaries could attempt to generate adversarial examples using an $\epsilon$ lower than 4/255, e.g., 1/255, 2/255, or 3/255, but the attack success rate would be low. Therefore, new intensity values are not created by FGSM attacks. The EBD cannot detect adversarial examples generated by FGSM attacks using samples that contain a lot of white or black. Because the adversarial examples contain a small number of new intensity values, there is no difference between the distribution of benign samples and that of adversarial examples.

Adversaries could attempt a CW attack by adding a penalty term to reduce the difference in entropy before and after bit-depth reduction. A CW attack should be able to compute the gradient of the objective function during the optimization process, but bit-depth reduction makes it impossible to directly compute the gradient because it is not differentiable [15]. Therefore, the adversary would have to perform a CW attack by adding a penalty term using the gradient descent process used in the BPDA attack. However, the objective function of a CW attack includes a penalty term to minimize the difference between the original sample and the adversarial example. Because of this penalty term, it would be difficult for a CW attack to determine perturbations with exact values such as 1/255 and 2/255. Even if this were possible, it would be more inefficient for a CW attack to evade our detector than an FGSM or a BIM attack because the CW attack generates adversarial examples using a very high number of attack iterations.

## 6 Conclusion

In this study, we proposed a detector that utilizes the difference in entropy of input sample after reducing the bit depth to defend DNNs against powerful and sophisticated adversarial attacks. Our proposed method successfully detected adversarial examples generated by various attack methods, including FGSM, BIM, MIM, DeepFool, and CW. Future research will be to extend the experiment to other datasets, such as voice and video domains. In addition, the concept of our EBD can be applied to detect images generated by DeepFake [47] and can be applied to medical domain [50, 51]. We could also develop a defense method that does not use the target model's output and does not train other networks.

**Author Contributions** All authors contributed to the design of our proposed method and the conceptualization of this research. GR conducted literature surveys, evaluated our detection method, and wrote the main manuscript text. DC analyzed the entropy of legitimate samples and adversarial examples. DC was responsible for overall program administration and reviewed the manuscript.

**Funding** This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government (MSIT) (No. 2021R1A4A1029650) and Institute of Information and communications Technology Planning and Evaluation (IITP) grant funded by the Korea government(MSIT) (No. 2021-0-00111, AI-based Cyber attacks and Defense for Autonomous Vehicles).

**Data availability** The data that support the findings of this study are openly available in MNIST [60] at http://yann.lecun.com/exdb/mnist/, CIFAR-10 [61] at https://www.cs.toronto.edu/~kriz/cifar.html, ImageNet [37] at https://www.image-net.org.

## Declarations

**Conflict of interest** The authors declare that they have no conflict of interest.

**Ethical approval** This article does not contain any studies with animals performed by any of the authors.



## References

1. He, K., Zhang, X., Ren, S.: Deep residual learning for image recognition, In: Proceedings of the IEEE/CVF Computer Vision Pattern Recognition (CVPR), pp. 770–778, (2016)
2. Lin, T.Y., Dollár, P., Girshick, R.B., He, K., Hariharan, B., Belongie, S.J.: Feature pyramid networks for object detection, In:

Proceedings of the IEEE/CVF Computer Vision Pattern Recognition (CVPR), pp. 2117–2125, (2017)
3. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A,N., Kaiser, Ł., Polosukhin, I.: Attention is all you need, Adv. Neural Info. Process. Syst. **30**, (2017)
4. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing properties of neural networks, In: Proceedings of International Conference on Learning Representations (ICLR), (2014)
5. Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z.B., Swami, A.: The limitations of deep learning in adversarial settings, In: Proceedings of the IEEE European Symposium on Security and Privacy (SP), pp. 372–387, (2016)
6. Moosavi-Dezfooli, S.M., Fawzi, A., Frossard, P.: Deepfool: A simple and accurate method to fool deep neural networks, In: Proceedings of the IEEE/CVF Computer Vision Pattern Recognition (CVPR), pp. 2574–2582, (2016)
7. Xie, C., Zhang, Z., Zhou, Y., Bai, S., Wang, J., Ren, Z., Yuille, A.: Improving transferability of adversarial examples with input diversity, In: Proceedings of the IEEE/CVF Computer Vision Pattern Recognition (CVPR), pp.2730–2739, (2019)
8. Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A. Kohno, T., Song, D.: Robust physical-world attacks on deep learning visual classification, In: Proceedings of the IEEE/CVF Computer Vision Pattern Recognition (CVPR), pp. 1625–1634, (2018)
9. Zhao, Y. Zhu, H., Liang, R., Shen, Q., Zhang, S., Chen, K.: Seeing isn't believing: Towards more robust adversarial attack against real world object detectors, In: Proceedings of ACM Conference on Computer and Communications Security (CCS), pp. 1989–2004, (2019)
10. Sharif, M., Bhagavatula, S., Bauer, L, Reiter, M.K.: Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition, In: Proceedings of ACM Conference on Computer and Communications Security (CCS), pp. 1528–1540, (2016)
11. Ryu, G., Park, H., Choi, D.: Adversarial attacks by attaching noise markers on the face against deep face recognition. J. Inf. Secur. Appl. **60**, 102874 (2021)
12. Xie, C., Wang, J., Zhang, Z., Zhou, Y., Xie, L., Yuille, A.: Adversarial examples for semantic segmentation and object detection, In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 1369–1378, (2017)
13. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks, In: Proceedings of International Conference on Learning Representations (ICLR), (2018)
14. Papernot, N., McDaniel, P., Wu, X., Jha, S., Swami, A.: Distillation as a defense to adversarial perturbations against deep neural networks, In: Proceedings of the IEEE Symposium on Security and Privacy (SP), pp. 582–597, (2016)
15. Xu, W., Evans, D., Qi, Y.: Feature squeezing: Detecting adversarial examples in deep neural networks, In: Proceedings of Network and Distributed System Security Symposium (NDSS), (2018)
16. Ye, D., Chen, C., Liu, C., Wang, H., Jiang, S.: Detection defense against adversarial attacks with saliency map. Int. J. Intell. Syst. **37**(12), 10193–10210 (2021)
17. Prakash, A., Moran, N., Garber, S., Dilillo, A., Storer, J.: Deflecting adversarial attacks with pixel deflection, In: Proceedings of the IEEE/CVF Computer Vision Pattern Recognition (CVPR), pp. 8571–8580, (2018)
18. Naseer, M., Khan, S., Hayat, M., Khan, F.S., Porikli, F.: A self-supervised approach for adversarial robustness, In: Proceedings of the IEEE/CVF Computer Vision Pattern Recognition (CVPR), pp. 262–271, (2020)
19. Goodfellow, I., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples, In: Proceedings of International Conference on Learning Representations (ICLR), (2015)
20. Kurakin, A., Goodfellow, I., Bengio, S.: Adversarial examples in the physical world, In: Proceedings of International Conference on Learning Representations (ICLR), (2017)
21. Carlini, N., Wagner, D.: Towards evaluating the robustness of neural networks, In: Proceedings of the IEEE Symposium on Security and Privacy (SP), pp. 39–57, (2017)
22. Zheng, H., Zhang, Z., Gu, J., Lee, H., Prakash, A.: Efficient adversarial training with transferable adversarial examples, In: Proceedings of the IEEE/CVF Computer Vision Pattern Recognition (CVPR), pp. 1181–1190, (2020)
23. Xiao, C., Li, B., Zhu, L.Y., He, W., Liu, M., Song, D.: Generating adversarial examples with adversarial networks, arXive preprint arXiv:1801.02610, (2018)
24. Goodfello, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets, Adv. Neural Info. Process. Syst. 27, (2014)
25. Athalye, A., Carlini, N., Wagner, D.: Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples, In: Proceedings of International Conference on Machine Learning (ICML), pp. 274–283, (2018)
26. Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z.B., Swami, A.: Practical black-box attacks against machine learning, In: Proceedings of ACM Asia Conference on Computer and Communications Security (AsiaCCS), pp. 506–519, (2017)
27. Park, H., Ryu, G., Choi, D.: Partial retraining substitute model for query-limited black-box attacks. Appl. Sci. **10**, 7168 (2020)
28. Dong, Y., Liao, F., Pang, T., Su, H., Zhu, J., Hu, X., Li, J.: Boosting adversarial attacks with momentum, In: Proceedings of the IEEE/CVF Computer Vision Pattern Recognition (CVPR), pp. 9185–9193, (2018)
29. Inkawhich, N., Wen, W., Li, H.H., Chen, Y.: Feature space perturbations yield more transferable adversarial examples, In: Proceedings of the IEEE/CVF Computer Vision Pattern Recognition (CVPR), pp. 7066–7074, (2019)
30. Meng, D., Chen, H.: MagNet: A two-pronged defense against adversarial examples, In: Proceedings of ACM Conference on Computer and Communications Security (CCS), pp. 135–147, (2017)
31. Samangouei, P., Kabkab, M., Chellappa, R.: Defense-GAN: Protecting classifiers against adversarial attacks using generative models, In: Proceedings of International Conference on Learning Representations (ICLR), (2018)
32. Song, Y., Kim, T., Nowozin, S., Ermono, S., Kushman, N.: PixelDefend: Leveraging generative models to understand and defend against adversarial examples, In: Proceedings of International Conference on Learning Representations (ICLR), (2018)
33. Oord, A.V.D., Kalchbrenner, N., Kavukcuoglu, K.: Pixel recurrent neural networks, In: Proceedings of International Conference on Machine Learning (ICML), pp. 1747–1756, (2016)
34. Guo, C., Rana, M., Cisse, M., Maaten, L.V.D.: Countering adversarial images using input transformations, In: Proceedings of International Conference on Learning Representations (ICLR), (2018)
35. Xie, C., Wang, J., Zhang, Z., Ren, Z., Yuille, A., Mitigating adversarial effects through randomization, In: Proceedings of International Conference on Learning Representations (ICLR), (2018)
36. Jeddi, A., Shafiee, M.J., Karg, M., Scharfenberger, C., Wong, A.: Learn2Perturb: An end-to-end feature perturbation learning to improve adversarial robustness, In: Proceedings of the IEEE/CVF Computer Vision Pattern Recognition (CVPR), pp. 1241–1250, (2020)
37. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A large-scale hierarchical image database, In: Proceedings

of the IEEE/CVF Computer Vision Pattern Recognition (CVPR), pp. 248–255, (2009)
38. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks, In: Proceedings of European Conference on Computer Vision (ECCV), pp. 630–645, (2016)
39. Szegedy, C., Ioffe, S., Vanhoucke, V.: Inception-v4, Inception-ResNet and the impact of residual connections on learning, In: Proceedings of AAAI conference on artificial intelligence, (2017)
40. Miyato, T., Dai, A.M., Goodfellow, I.: Adversarial training methods for semi-supervised text classification, arXiv preprint arXiv:1605.07725, (2016)
41. Papernot, N., McDaniel, P.: Extending defensive distillation, arXiv preprint arXiv:1705.05264, (2017)
42. Metzen, J.H., Genewein, T., Fischer, V., Bischoff, B.: On detecting adversarial perturbations, In: Proceedings of International Conference on Learning Representations (ICLR), (2017)
43. Lu, J., Issaranon, T., Forsyth, D.: SafetyNet: Detecting and rejecting adversarial examples robustly, In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 1–9, (2017)
44. Bradshaw, J., Matthews, A.G.G., Ghahramani, Z.: Adversarial examples, uncertainty, and transfer testing robustness in Gaussian process hybrid deep networks, arXiv preprint arXiv:1707.02476, (2017)
45. Buckman, J., Roy, A., Raffel, C., Goodfellow, I.: Thermometer encoding: One hot way to resist adversarial examples, In: Proceedings of International Conference on Learning Representations (ICLR), (2018)
46. Yin, Z., Wang, H., Wang, J., Tang, J., Wang, W.: Defense against adversarial attacks by low-level image transformations. Int. J. Intell. Syst. **35**, 1453–1466 (2020)
47. Zakharov, E., Shysheya, A., Burkov, E., Lempitsky, V.: Few-shot adversarial learning of realistic neural talking head models, In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 9459–9468, (2019)
48. Pang, T., Xu, K.N., Dong, Y., Du, C., Chen, N., Zhu, J.: Rethinking softmax cross-entropy loss for adversarial robustness, In: Proceedings of International Conference on Learning Representations (ICLR), (2020)
49. Xiao, C., Zhong, P., Zheng, C.: Enhancing Adversarial Defense by k-Winners-Take-All, In: Proceedings of International Conference on Learning Representations (ICLR), (2020)
50. Kim, Y.J., Ganbold, B., Kim, K.G.: Web-based spine segmentation using deep learning in computed tomography images. Healthc. Inform. Res. **26**, 61–67 (2020)
51. Yoon, D., Lim, H.S., Jung, K., Kim, T.Y., Lee, S.: Deep learning-based electrocardiogram signal noise detection and screening model. Healthc. Inform. Res. **25**, 201–211 (2019)
52. Ma, S., Liu, Y., Tao, G., Lee, W.C., Zhang, X.: NIC: Detecting adversarial samples with neural network invariant checking, In: Proceedings of Network and Distributed Systems Security (NDSS), (2019)
53. Ma, X., Li, B., Wang, Y., Erfani, S.M., Wijewickrema, S., Schoenebeck, G., Song, D., Houle, M.E., Bailey, J.: Characterizing adversarial subspaces using local intrinsic dimensionality, In: Proceedings of International Conference on Learning Representations (ICLR), (2018)
54. Cohen, G., Sapiro, G., Giryes, R.: Detecting adversarial samples using influence functions and nearest neighbors, In: Proceedings of the IEEE/CVF Computer Vision Pattern Recognition (CVPR), pp. 14453–14462, (2020)
55. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition, In: Proceedings of International Conference on Learning Representations (ICLR), (2015)
56. Ryu, G., Choi, D.: A hybrid adversarial training for deep learning model and denoising network resistant to adversarial examples. Appl. Intell. **53**(8), 9174–9187 (2023)
57. Ryu, G., Choi, D.: Feature-based adversarial training for deep learning models resistant to transferable adversarial examples, IEICE Trans. Inf. Syst. E105-D(5), 1039–1049 (2022)
58. Shan, S., Wenger, E., Wang, B., Li, B., Zheng, H., Zhao, B.Y.: Gotta catch 'em all: using honeypots to catch adversarial attacks on neural networks, In: Proceedings of ACM Conference on Computer and Communications Security, pp. 67–83, (2020)
59. Na, H., Ryu, G., Choi, D.: Adversarial attack based on perturbation of contour region to evade steganalysis-based detection. IEEE Access **9**, 122308–122321 (2021)
60. Li, D.: The mnist database of handwritten digit images for machine learning research. IEEE Signal Process. Mag. **29**, 141–142 (2012)
61. Jrizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images, Citeseer, (2009)
62. Keras Applications GitHub Website. https://github.com/keras-team/keras-applications
63. CleverHans GitHub Website. https://github.com/tensorflow/cleverhans