An Efficient Preprocessing-based Approach to Mitigate Advanced Adversarial Attacks

Han Qiu, Yi Zeng, Qinkai Zheng, Shangwei Guo, Tianwei Zhang, and Hewu Li

Abstract—Deep Neural Networks are well-known to be vulnerable to Adversarial Examples. Recently, advanced gradient-based attacks were proposed (e.g., BPDA and EOT), which can significantly increase the difficulty and complexity of designing effective defenses. In this paper, we present a study towards the opportunity of mitigating those powerful attacks with only pre-processing operations. We make the following two contributions. First, we perform an in-depth analysis of those attacks and summarize three fundamental properties that a good defense solution should have. Second, we design a lightweight preprocessing function with these properties and the capability of preserving the model's usability and robustness against these threats. Extensive evaluations indicate that our solutions can effectively mitigate all existing standard and advanced attack techniques, and beat 11 state-of-the-art defense solutions published in top-tier conferences over the past 2 years.

Index Terms—Adversarial Examples, Deep Learning, Adversarial Attacks, BPDA.

1 Introduction

Artificial Intelligence (AI), especially Deep Learning (DL) has become the most important class of technologies in the past decade. Recently, with the rapidly increasing requirements for experimenting the DL applications such as training or inference, cloud computing providers start to provide DL-related computing as services. This Deep Learning as a Service (DLaaS) can significantly improve the usage of the end-users or small enterprises to train a DL model or use existing DL models to inference. For instance, Microsoft started to provide the inference as a service [1] that can help users to use DL models with low latency and cost. However, novel security issues are found in such DLaaS scenarios that threaten the usage of the DLaaS.

Szegedy et al. [2] proposed the concept of Adversarial Examples (AEs): with imperceptible modifications to the input, the Deep Neural Network (DNN) model will be fooled to give wrong prediction results. Since then, a huge amount of research effort has been spent to enhance the powers of the attacks, or mitigate the new attacks (Fig. 1). This leads to an arms race between adversarial attacks and defenses. Basically, the generation of AEs can be converted into an optimization problem: searching for the minimal perturbations that can cause the model to predict a wrong label. Attackers used the gradient-based approaches to identify the optimal perturbations (e.g., FGSM [3], I-FGSM [4], LBFGS [2], C&W [5]). To defeat those attacks, a lot of defenses were proposed to obfuscate the gradients such as making them shattered or stochastic [6], [7], [8], [9], [10].

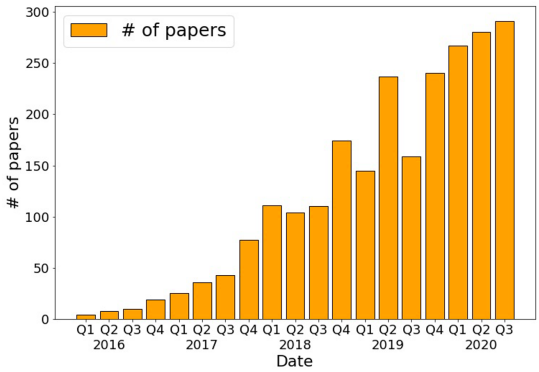

Fig. 1: The number of research papers published on Arxiv.org about adversarial examples. Data source: [11].

Unfortunately, those gradient obfuscation-based defenses were further broken by advanced attacks [12], [13]. *Backward Pass Differentiable Approximation* (BPDA) was introduced to handle the shattered gradients by approximating the gradients of non-differentiable functions. *Expectation over Transformation* (EOT) was designed to deal with the stochastic gradient by calculating the expectation of gradients of random functions. These two attacks have successfully defeated the previous defenses [12], and even new defenses published after their disclosure was still proven to be vulnerable to BPDA, EOT, or their combination [14].

The question we want to address is: *is it possible to continue the arms race by mitigating the aforementioned advanced attacks with more robust defense solutions?* This is a challenging task. First, these attacks assume very high adversarial capabilities [14]: the attacker knows every detail of the DL model and the potential defenses. This significantly increases the difficulty of defense designs and invalidates existing solutions that require hiding the model details or

- *H. Qiu and H. Li are with Institute for Network Sciences and Cyberspace, Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing, China, 100084. Email: qiuhan@tsinghua.edu.cn, lihewu@cernet.edu.cn.*
- *Y. Zeng is with the University of California San Diego, CA, USA, 92122. Email: y4zeng@eng.ucsd.edu*
- *Q. Zheng is with the Shanghai Jiao Tong University, Shanghai, China, 200240. Email: paristech-hill@sjtu.edu.cn*
- *S. Guo is with College of Computer Science, Chongqing University, Chongqing, China, 400044. Email: swguo@cqu.edu.cn.*
- *T. Zhang (corresponding author) is with the Nanyang Technological University, Singapore, 639798. Email: tianwei.zhang@ntu.edu.sg*

Yi Zeng and Han Qiu have the equal contribution.

defense mechanisms. Second, BPDA and EOT target the root causes of gradient obfuscation: the non-differentiable operation can always be approximated, and the random operation can be estimated by its expectation. It is indeed difficult for the defender to bypass these assumptions while still preserving model usability.

One possible defense strategy is adversarial training [4]: we can keep generating adversarial examples from the training-in-progress model using the Projected Gradient Descent (PGD) attack technique, and augmenting them into the training set to improve the model's robustness. This strategy is shown to be effective against different types of adversarial attacks including BPDA and EOT. However, it can bring a significant cost to perform adversarial training with large-scale DNN models and datasets. So we are more interested in an efficient method, which can be directly applied to a given model without altering it. [15] proposed a preprocessing-based solution: they tested 25 existing preprocessing functions and placed them into 10 groups. For each inference, an ensemble of 5∼10 functions is randomly selected to transform the input before feeding it to the target model. This strategy can mitigate a more sophisticated BPDA, where the adversary attempts to use a neural network to approximate the non-differentiable operations.

In this paper, we also focus on the preprocessing-based defense to enhance the model's robustness against all existing adversarial attacks. Different from [15], we aim to utilize a single lightweight transformation function to preprocess the input images. This is expected to significantly reduce the computation cost and logic complexity for model inference, which is critical when the task is deployed in resource-constrained edge and IoT devices. To achieve this goal, we make the following contributions.

First, we analyze the features and assumptions of different attacks and identify three properties for designing a qualified preprocessing function $g(\cdot)$. The first one is *usability-preserving*, which is to guarantee $g(\cdot)$ will not affect the model performance on clean samples. The next two properties are *non-differentiability* and *non-approximation*, to enhance the model robustness against both standard and advanced gradient-based attacks.

Second, we introduce a novel preprocessing function that can meet the above properties. Our function consists of two steps: (1) a DCT-based quantization is used to compress the input images, which can achieve *non-differentiability*; (2) a dropping-pixel strategy is further introduced to distort the image via random pixel dropping and displacement. This step can increase the difficulty and fidelity of *approximation*. Both steps are *usability-preserving*, thus their integration will cause a negligible impact on the model performance.

We conduct extensive experiments to show the effectiveness of our solutions. It can constrain the attack success rate under 7% even with 10000 rounds of BPDA+EOT attack (dozens of GPU hours for 100 samples), which significantly outperform 11 state-of-the-art gradient obfuscation defenses published recently in top-tier conferences. To better promote this research direction, we release a toolkit online[1], including the implementation of our defense techniques, a summary of other defense methods as well as various adversarial attacks.

The roadmap of this paper is given as follows. In Section 2, the research background including the basic attack concept and development history is given. In Section 3, we define the threat model and defense requirements. In Section 4, the methodology insights are illustrated with three specific properties. In Section 5, the practical defense solution is presented. In Section 6, the extensive evaluation is listed to show the effectiveness of our method. In Section 8, we conclude our work.

2 BACKGROUNDS

2.1 Attack Concept and Scenarios

An adversary can add human-unnoticeable perturbations on the original input to fool a DNN classifier. Formally, the target DNN model is a mapping function $f(\cdot)$. Given a clean input sample x, the corresponding AE is denoted as $\widetilde{x} = x + \delta$ where δ is the adversarial perturbation. Then AE generation can be formulated as the optimization problem in Equation 1a (targeted attack where $l' \neq f(x)$ is the desired label set by the attacker) or Equation 1b (untargeted attack).

$$min\|\delta\|, s.t.\ f(\widetilde{x}) = l' \tag{1a}$$

$$min\|\delta\|, s.t.\ f(\widetilde{x}) \neq f(x) \tag{1b}$$

Generally, there are two attack scenarios [16], determined by the adversary's knowledge about the target system. (1) *White-box scenario*: the adversary knows every detail about the neural network model including the architecture and all the parameters. He is also aware of the defense mechanism and the corresponding parameters. (2) *Black-box scenario*: the adversary does not have any knowledge about the victim system. In addition to these two scenarios, there are also some works [7] assuming the adversary knows all details about the model but not the defense mechanism. It is not quite realistic and reasonable to hold the defense secret, as "this widely held principle is known in the field of security as Kerckhoffs' principle." [16]. So we exclude this scenario in this paper.

2.2 Development History

Round 1: attack. As the first study, Szefedt et al. [2] adopted the L-BFGS algorithm to solve the optimization problem of AE generation. Shortly after this work, a couple of gradient-based methods were introduced to enhance the attack techniques: the gradient descent evasion attack [17] calculated the gradients of neural networks to generate AEs; *Fast Gradient Sign Method* (FGSM) [3] calculated the adversarial perturbation based on the sign of gradients, which was further improved by its iterative versions (I-FGSM [4] and MI-FGSM [18]). Deepfool [19] is another iterative method that outperforms previous attacks by searching for the optimal perturbation across the decision boundary. Meanwhile, some other techniques were proposed to increase the attack efficiency: *Jacobian-based Saliency Map Attack* (JSMA) [20] estimated the saliency map of pixels w.r.t the classification output, and only modified the most salient pixels. One pixel

1. https://github.com/YiZeng623/Advanced-Gradient-Obfuscating

attack [21] is an extreme-case attack where only one pixel can be modified to fool the classifier.

Round 2: defense. With the advance of adversarial attacks, defense solutions were proposed to increase the robustness of DNN models. They can be classified into three categories. The first direction is adversarial training [4], [22], [23], where AEs are used with normal examples together to train DNN models to recognize and correct malicious samples. The second direction is to train other models to assist the target one. Magnet [24] used detector networks to identify AEs by approximating the manifold of normal examples. Generative Adversarial Trainer [25] utilized training target networks along with a generative network to generate adversarial perturbation for the target model to distinguish. The third direction is to design AE-aware network architecture or loss function. Deep Contractive Networks [26] added a contractive penalty to alleviate the effects of AEs. Input Gradient Regularization [27] countered AEs by penalizing the degree of variations of input perturbations on the output. Defensive distillation [28] generated soft training labels from one network and retrained a second network with higher robustness. This method claimed to have very high resistance against AEs and was one of the strongest defenses at that time.

Round 3: attack. A more powerful attack, C&W [5], was proposed by updating the objective function to minimize l_p distance between AEs and normal examples. C&W can effectively defeat Defensive Distillation [5] and other defenses with assisted models [29] with high attack success rates.

Round 4: defense. Since then, new defense strategies were introduced to increase the difficulty of AE generations by obfuscating the gradients. Five input transformations were tested to counter AEs in [6], including image cropping and rescaling, bit-depth reduction, JPEG compression, total variance minimization (TV), and image quilting. Prakash et al. [7] designed *Pixel Deflection* (PD), which randomly redistributes a small number of pixels as artificial perturbation and applies wavelet-based denoising to remove both artificial and adversarial perturbation. Xie et al. [8] proposed to use a randomization layer to randomly rescale the input image with zero-paddings. Buckman et al. [9] introduced Thermometer encoding, which encodes input images with discrete values to prevent the direct calculation of gradient descent during AE generation. Das et al. [30] proposed SHIELD that compresses different regions of an image with random compression levels to mitigate AE perturbations. Those solutions are effective against all prior attacks.

Round 5: attack. To particularly target the gradient obfuscation-based defenses, two more advanced attacks were introduced. BPDA [12] copes with the non-differentiable obfuscation operation by approximating the gradients during back-propagation. EOT [31] deals with the randomization obfuscation operation by averaging the gradients of multiple sessions. More detailed descriptions about BPDA and EOT can be found in Section 4. After the disclosure of these two attacks, a large number of defense works have been published. Unfortunately, most of them did not consider or incorrectly evaluate these two attacks, and some representative solutions have been analyzed and proved to be incapable of defeating BPDA and EOT attacks [14]. Up to now, there are still no effective preprocessing-based defenses. This is what we aim to address in this paper.

3 Threat Model and Defense Requirements

It is necessary to specify the adversarial capabilities and defense requirements in our consideration as follows.

3.1 Threat Model

Adversarial Goals. There are two main types of adversarial attacks: untargeted attacks that try to mislead the DNN models to an arbitrary label different from the correct one, and targeted attacks which succeed only when the DNN model predicts the input as one specific label set by the adversary [5]. In this paper, we focus on evaluating the targeted attacks. The untargeted attacks can be mitigated in a similar way.

Adversary's Knowledge. We consider a white-box scenario, where the adversary has full knowledge of the DNN model, including the network architecture, exact values of parameters, and hyper-parameters. We further assume that the adversary has full knowledge of the proposed defense, including the algorithms and parameters. For the defenses employing randomization techniques, we assume the random numbers generated in real-time are perfect with a large entropy such that the adversary cannot obtain or guess the correct values.

It is worth noting that this white-box scenario represents the strongest adversaries. Under such a scenario, a big number of existing state-of-the-art defenses are invalidated as shown in [14]. This also significantly increases the difficulty of defense designs.

Adversarial Capabilities. The adversary is outside of the DNN classification system, and he is not able to compromise the inference computation or the DNN model parameters (e.g., via fault injection to cause bit-flips [32] or backdoor attacks [33]). All he can do is to manipulate the input data with imperceptible perturbations. In the context of computer vision tasks, he can directly modify the input image pixel values within a certain range. We use l_∞ and l_2 distortion metrics to measure the scale of added perturbations: we only allow the generated AEs to have either a maximum l_∞ distance of $8/255$ or a maximum l_2 distance of 0.05 [12].

3.2 Defense Requirements

Based on the above threat model, we list a couple of requirements for a good defense solution:

First, there should be no modifications to the original DNN model, e.g., retraining a model with different structures [28] or datasets [34]. We set this requirement for two reasons. (1) Model retraining can significantly increase the computation cost, especially for large-scale DNN models (e.g. ImageNet scale [35]). (2) Those defense methods lack generality to cover various types of attacks. They "explicitly set out to be robust against one specific threat model" [16].

Second, we consider adding a preprocessing function over the input samples before feeding them into the DNN models. Such preprocessing operation can either remove the effects of adversarial perturbations on the inference or make it infeasible for the adversary to generate AEs adaptively,

even he knows every detail of the operation. This function should be general-purpose and applicable to various types of data and DNN models of similar tasks.

Third, this preprocessing function should be lightweight with negligible computation cost to the inference pipeline. Besides, it should also preserve the usability of the original model without decreasing its prediction accuracy. Input preprocessing can introduce a trade-off between security and usability: the side effect of correcting the adversarial examples can also alter the prediction results of clean samples. A qualified operation should balance this trade-off with maximum impact on the adversarial samples and minimal impact on the clean ones.

4 Methodology Insights

We aim to design a preprocessing function $g(\cdot)$, which transforms an input image $x \in \mathcal{X}$ to an output with the same dimension. Then given a DNN model $f(\cdot)$, the inference process becomes $y = f(g(x))$. This function $g(\cdot)$ needs to mitigate the adversarial attacks within the threat model and satisfy the defense requirements, as described in the previous section. We identify some properties and design philosophy of a good methodology in this section and give a specific algorithm in the next section.

4.1 Methodology Presentation

This preprocessing function must preserve the usability of the target model, i.e., exerting minimal influence on the accuracy of clean samples. This gives the first property:

Property 1. (Usability-preserving) $g(\cdot)$ cannot affect the prediction results of clean input: $f(g(x)) \approx f(x), \forall x \in \mathcal{X}$.

Second, as most of the attacks generate adversarial examples by calculating the gradients of the model parameters. When a preprocessing function is introduced, this calculation becomes: $\bigtriangledown_x f(g(x)) = \bigtriangledown_x f(x) \bigtriangledown_x g(x)$. So a common approach is shattered gradient-based defense, where the preprocessing operation $g(\cdot)$ is designed to be non-differentiable. With this property, the adversary is not able to craft AEs based on the gradient of the model using standard methods (e.g., FGSM, C&W, Deepfool, etc.).

Property 2. (Non-differentiability) $g(\cdot)$ is non-differentiable, i.e., it is hard to compute an analytical solution for $\bigtriangledown_x g(x)$.

It is interesting to note that this property can defeat the advanced EOT attack [12] as well. This attack was proposed to invalidate the defense solutions based on model input randomization, by statistically computing the gradients over the expected transformation of the input x. Formally, for a preprocessing function $g(\cdot)$ that randomly transforms x from a distribution of transformations T, EOT optimizes the expectation over the transformation with respect to the input by: $\bigtriangledown_x \mathbb{E}_{t\sim T} f(g(x)) = \mathbb{E}_{t\sim T} \bigtriangledown_x f(g(x))$. EOT can help to get a proper expectation with samples at each gradient descent step. However, if $g(\cdot)$ is non-differentiable, the adversary cannot calculate the gradient expectation to generate AEs either.

A function $g(\cdot)$ with the non-differentiability property can still be vulnerable to sophisticated attacks, e.g., BPDA [12], where the adversary can approximate $g(\cdot)$ with a differentiable function $g'(x)$. For instance, in the experimentation of the initial BPDA attack [12], the adversary used $g'(x) = x$ as an approximation to calculate the gradient of $g(x)$. He keeps $g(\cdot)$ on the forward pass and replaces it with x on the backward pass. The derivative of the $g(\cdot)$ will be approximated as the derivative of the identity function, which is 1. In [15], neural nets were further trained to approximate non-differentiable functions, which can defeat a wider range of shattered gradient-based defenses than the identity function. To mitigate such threats, the preprcessing function must meet the following property:

Property 3. (Non-approximation) It is difficult to find a differentiable $g'(x)$ that can approximate the non-differentiable preprocessing function $g(x)$ when calculating its gradients, i.e., $\bigtriangledown_x g'(x) \approx \bigtriangledown_x g(x)$.

A common strategy to reduce the possibility and fidelity of approximating a non-differentiable function is to add randomization in the operation. If the degree of randomization is large enough, then it will be difficult for the adversary to find a qualified deterministic differentiable function for replacement, even using neural networks. However, a high random transformation can also affect the model's usability (Property 1). So the key to the design of this function $g(\cdot)$ is to balance the trade-off between Properties 1 and 3 with a random non-differentiable operation. Past work [15] adopted an ensemble of dozens of weak preprocessing functions to defend against BPDA, making the entire inference system quite complex. In this paper, we aim to simplify this by designing one single function to achieve the same goal.

4.2 Methodology Summary

A preprocessing function $g(\cdot)$ that can meet the above three properties can effectively increase the DNN model's robustness against existing adversarial attacks. Specifically, for standard gradient-based attacks (FGSM, C&W, LBFGS, Deepfool), non-differentiability in Property 2 can prohibit the direct calculation of gradients, and the randomization employed in Property 3 can obfuscate the gradient values. A function with these two properties can provide higher robustness against these standard attacks.

For those advanced attacks, the gradient expectation attack (EOT) can be mitigated by Property 2. If a qualified function with Property 3 is identified, the adversary may have difficulty in discovering a replacement that can accurately approximate this function. Then gradient approximation attack (BPDA) becomes infeasible or at least requires a much higher cost. The combination of these two attacks cannot compromise the model's robustness either.

5 Our Proposed Solution

5.1 Overview

Our proposed function $g(\cdot)$ involves two critical steps to process the input images. The first step (Step 1 in Algorithm 1) adopts a DCT-based defensive quantization. Based on [36], we further improve the quantization table to better adapt to the machine's visionary behavior. This can realize the non-differentiability property while preserving the model's usability. The second step (Step 2 in Algorithm 1)

ALGORITHM 1: Defense preprocessing function

Input: original image $I \in \mathbb{R}^{h \times w}$
Output: processed image $I' \in \mathbb{R}^{h \times w}$
Parameters: defensive quantization table Q, distortion limit $\delta \in [0, 1]$, size of grid d.

1 $x_0 = 0$, $y_0 = 0$;
2 $n_w = w/d$, $n_h = h/d$;
3 $\mathcal{G}_I = \{(x_m, y_n) | (m, n) \in \{(0, ..., n_w) \times (0, ..., n_h)\}\}$;
/* Step 1: DTC-based Quantization */
4 **Set** defensive quantization table Q.
5 **for** (x_m, y_n) *in* $\mathcal{G}_I\{(x_0, y_0)\}$ **do**
6 $dct = DCT(I(x_{m-1} : x_m, y_{n-1} : y_n))$;
7 $dct_q = Quantization(dct, Q)$;
8 $dct_d = Dequantization(dct_q, Q)$;
9 $I_q(x_{m-1} : x_m, y_{n-1} : y_n) = IDCT(dct_d)$;
10 **end**
/* Step 2: Image Distortion */
11 **Set** random distortion limit $\delta \in [0, 1]$.
12 **Set** random size of grid d.
13 I'= `ImageDistortion`(I, d, δ);
14 **return** I';

ALGORITHM 2: Generating quantization table Q

Input: clean set $I^n \in \mathbb{R}^{n \times h \times w \times 3}$,
adversarial set $\dot{I}^n \in \mathbb{R}^{n \times h \times w \times 3}$,
Output: defensive quantization table Q

1 $Q_0 = O_{8 \times 8}$; /* Generating a zero matrix of size 8×8. */
2 **for** I_i *in* I^n **do**
3 **for** $I_{i,channel}$ *in* I_i **do**
4 $x_0 = 0$, $y_0 = 0$;
5 $n_w = w/8$, $n_h = h/8$;
6 $\mathcal{G}_I i = \{(x_m, y_n) | (m, n) \in \{(0, ..., n_w) \times (0, ..., n_h)\}\}$;
7 **for** (x_m, y_n) *in* $\mathcal{G}_I i\{(x_0, y_0)\}$ **do**
8 $dct_I = DCT(I_{i,channel}(x_{m-1} : x_m, y_{n-1} : y_n))$;
9 $dct_{Adv} = DCT(\dot{I}_{i,channel}(x_{m-1} : x_m, y_{n-1} : y_n))$;
10 $difmat = |dct_I - dct_{Adv}|$;
$x_Q, y_Q = argmax(difmat)$;
11 $Q_0(x_Q, y_Q) += 1$;
12 **end**
13 **end**
14 **end**
15 $Q = (Q_0 / max(Q_0)) \times 80 + 20$;
16 **return** Q;

is inspired by a dropping-pixel strategy [6], [8]. We propose a novel technique to distort images by dropping randomly selected pixels of input images and displacing each pixel away from the original coordinates. Our proposed technique can generate highly randomized preprocessed images while keeping a high accuracy for DNN inference.

5.2 Step 1: DCT-based Quantization

The first step is described in Lines 4-9 in Algorithm 1. The input image is cut into grids of pixels with the size of the grid d. Pixels in each grid are transformed into the frequency space via Discrete Cosine Transform (DCT) [37] as shown. Here we use a 2D-DCT with a grid size of 8×8. A defensive quantization table Q is then used to quantize all the frequency coefficients. These DCT coefficients are further de-quantized and transformed back into the spatial space with an inverse DCT.

The critical factor in this step is the quantization table Q. [38] directly used the JPEG quantization table Q_{50} to remove the adversarial perturbations. This was proved ineffective as the JPEG quantization table was designed to compress the image based on the sensitivity of the human visual system. Later on, more effective approaches were proposed to mitigate certain adversarial attacks with randomized quantization tables [30] or a dedicated quantization table [36]. Such quantization techniques are proved to have better defense performance on AEs than directly deploying the JPEG quantization table Q_{50}. For an attacker, to defeat quantization-based defense, the adversarial perturbation on pixel values must be large enough to influence the quantization results. Therefore, the motivation of deploying quantization is to use such a non-differentiable function to increase the difficulty for generating AEs within a l_2 bound.

In our solution, we introduce a novel and more effective way to generate the quantization table, as shown in Algorithm 2. We generate our new quantization table Q in a statistical learning manner by summarizing the patterns of the AEs. Here we use the C&W attack method to generate the corresponding AE set (using different AE generation methods will lead to similar results). In the algorithm, first, all the 8×8 blocks in the spatial domain (I in Line 8) are collected from all the images' color channels for both the clean image set and the AE set ($\dot{I}$ in Line 9). By conducting DCT on all the 8×8 small blocks, we compare the difference of DCT frequency coefficients (Line 10) to statistically understand the difference brought by such AE attacks. This difference is presented as the coordinates of the particular frequency coefficients which have the most significant changes. We design our quantization table Q by such a statistical calculation.

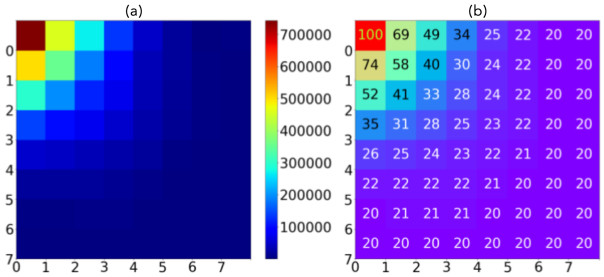

Fig. 2: Frequency space statistical results of AEs (a) and the defensive quantization table (b).

The statistical results of the spatial domain of the AEs with our DCT-based quantization are given in Fig. 2 (a). We can observe that the DC coefficients on the up-left corner are always significantly changed, and low frequencies are relatively changed more than high frequencies. The quantization table is then designed according to such statistics with the principle that the frequencies that are changed more often with larger values are sensitive to DNN models. We normalize all the values within $(0, 1)$ and remap each

value to the range of $(20, 100)$ (Line 15). The final Q table is shown in Fig. 2 (b).

5.3 Step 2: Image Distortion

The second step of the proposed defense is a novel transformation procedure by improving the image distortion method as the preprocessing function. It can provide a great random variance between the original and transformed samples without affecting the model performance.

We improved the dropping-pixels strategy [6], [8]. The general idea is to drop certain randomly selected pixels from the original image, and displace each pixel away from the original coordinates. The whole procedure of the second step consists of four stages, as illustrated in Fig. 3 in the paper and Algorithm 3.

ALGORITHM 3: Image Distortion

Input: original image $I \in \mathbb{R}^{h \times w}$
Output: distorted image $I' \in \mathbb{R}^{h \times w}$
Parameters: distortion limit $\delta \in [0, 1]$; size of grid d.

/* 1.Select a starting point, e.g., upper-left corner */
1 $x_0 = 0, y_0 = 0$;
/* 2.Random distortion over grids */
2 $n_w = w//d$, $n_h = h//d$;
3 $\mathcal{G}_I = \{(x_m, y_n) | (m, n) \in \{(0, ..., n_w) \times (0, ..., n_h)\}\}$;
4 **for** (x_m, y_n) *in* $\mathcal{G}_I \{(x_0, y_0)\}$ **do**
5 $\quad \delta_x \sim \mathcal{U}(-\delta, \delta)$;
6 $\quad \delta_y \sim \mathcal{U}(-\delta, \delta)$;
7 $\quad x_m = x_{m-1} + d \times (1 + \delta_x)$;
8 $\quad y_n = y_{n-1} + d \times (1 + \delta_y)$;
9 **end**
/* 3.Remapping grids in I to I' */
10 $\mathcal{G}_{I'} = \{(x'_m, y'_n) | x'_m = d \times m, y'_n = d \times n, (m, n) \in \{(0, ..., n_w) \times (0, ..., n_h)\}\}$;
11 **for** (x'_m, y'_n) *in* $\mathcal{G}_{I'} \{(x'_0, y'_0)\}$ **do**
12 $\quad I'(x'_{m-1} : x'_m, y'_{n-1} : y'_n) = \texttt{Remapping}(I(x_{m-1} : x_m, y_{n-1} : y_n))$;
13 **end**
/* 4.Reshape I' to the size of I */
14 $I' = \texttt{reshape}(I')$ s.t. $I' \in \mathbb{R}^{h \times w}$;
15 **return** I';

Lines 11-13 in Algorithm 1 illustrate the second step of our proposed image distortion method (in Algorithm 3). This image distortion method consists of four steps as follows. (1) One of the four corners is randomly selected as a starting point, e.g. the upper-left corner (line 1). (2) The original image is a randomly distorted grid by grid. For one grid, it will be either stretched or compressed based on a distortion level sampled from a uniform distribution $\mathcal{U}(-\delta, \delta)$ (line 5-8). (3) Distorted grids are then remapped to construct a new image (line 10-13). This step will drop pixels: the compressed grids will drop rows or columns of data; the stretched grids will cause the new image to exceed the original boundary such that the pixels mapped outside of the original boundary will be dropped (e.g., in Fig.1, the grid at the lower-right corner in stage 2 is dropped in stage 3). (4) Reshape the distorted image to the size of the original image by cropping or padding (line 14).

For the proposed defense, the distortion limit δ has an influence on the distortion level of each grid. It also affects the ratio of pixels that will be dropped. We apply a linear search of δ from 0.01 to 0.30, as shown in TABLE 4. The ASR becomes 0% under our defenses, which shows that the adversarial perturbation is delicate to this kind of distortion. A larger δ decreases the ACC on clean examples.

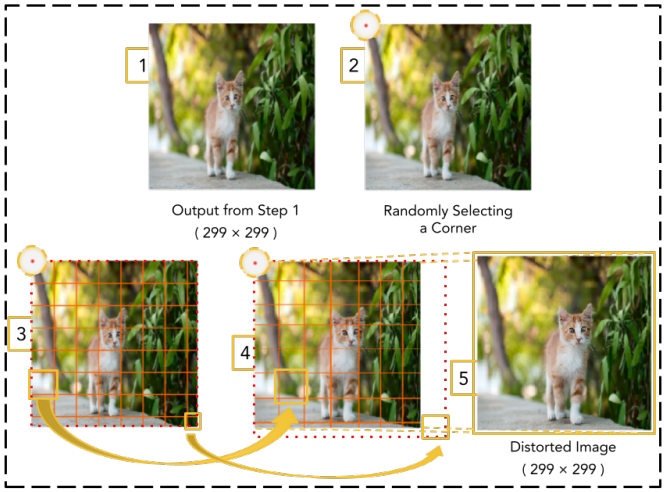

Fig. 3: Processing stages in the image distortion step.

This step can drop a certain ratio of pixels and change a huge number of pixel coordinates. In our experiments, the distortion limit δ is set as 0.15. In the ImageNet dataset, each image will have around 20%-30% pixels randomly dropped and more than 90% pixel coordinates changed each time after such preprocessing operation. This can guarantee high randomness and improve the difficulty of approximation with differentiable functions, while the model can still give correct predictions.

5.4 Security Analysis

Our preprocessing function can satisfy the three requirements, with the following quantitative justification.

For usability-preserving, we measure the prediction accuracy of clean samples for $f(g(x))$. Table 1 compares our solution with prior methods. We can observe all the methods can maintain very high model accuracy (ACC). For Property 2, our solution introduces defensive quantization, which is non-differentiable.

Defense	l_2	SSIM	ACC (top-1)
Our method	**0.22**	**0.30**	0.95
Rand [8]	0.21	0.31	0.96
FD [36]	0.00	1.00	0.97
SHIELD [30]	0.03	0.88	0.94
TV [6]	0.02	0.97	0.95
BdR [39]	0.00	1.00	0.92
PD [7]	0.02	0.98	0.97

TABLE 1: Quantitive measurement of variance of output images introduced by various kinds of defenses.

For Property 3, we measure the uncertainty of the preprocessed output to reflect the difficulty of approximation. Specifically, given one image, we use $g(\cdot)$ to preprocess it 100 times, and randomly select 2 outputs. We use l_2 norm and Structural Similarity (SSIM) score [40] to measure the variance between these two output images. Note that a larger l_2 norm or smaller SSIM score indicates a larger variance

between the two images. When l_2 norm is 0 or SSIM is 1, the output images are identical and the preprocessing function is deterministic. For each preprocessing function, we repeat the above process with 1000 randomly selected input images from the ImageNet dataset. The average SSIM score and l_2 norm are listed in Table 1. Our method can outperform other defenses with a larger l_2 norm and smaller SSIM. This indicates that our preprocessing function can introduce the highest randomness to the output, as well as the highest difficulty for the adversary to approximate it with differentiable functions.

6 Evaluation

6.1 Implementation

Configurations. We adopt Tensorflow as the DL framework for implementation. The learning rate of BPDA is 0.1 and the ensemble size[2] of EOT is 30. All experiments were conducted on a server equipped with 8 Intel I7-7700k CPUs and 4 NVIDIA GeForce GTX 1080 Ti GPUs.

Target Model and Dataset. Our methods are general-purpose and can be applied to various models as a preprocessing step for computer vision tasks. Without the loss of generality, we choose a pre-trained Inception V3 model [41] over the ImageNet dataset as the target model of attacks and defenses. This state-of-the-art model can reach 78.0% top-1 and 93.9% top-5 accuracy. We randomly select 100 images from the ImageNet Validation dataset for AE generation. These 100 images can all be predicted correctly by this Inception V3 model.

Metrics. The pixel values are normalized to $[0, 1]$. We use the l_2 norm to measure the number of perturbations generated by each attack, which is calculated by computing the total root-mean-square distortion normalized by the number of pixels ($299 \times 299 \times 3$). We only accept adversarial examples with a l_2 norm smaller than 0.05. We consider the targeted attacks where each target label different from the correct one is randomly generated [12]. The BPDA and EOT attacks are iterative processes: we stop the attack when an example is generated which is predicted as its corresponding target label and the l_2 norm is smaller than 0.05. For each attack round, we measure the prediction accuracy of the generated AEs (ACC) and the attack success rate (ASR) of the targeted attack. Note in this section, the ACCs are all top-1 accuracy. A higher accuracy or lower attack success rate indicates the defense is more resilient against the attacks.

6.2 Mitigating BPDA Attack

We first evaluate our method against BPDA. The BPDA attack evaluated in this subsection is experimented with the same as in [12]. As mentioned in Section 4, the core idea of BPDA is to approximate the obfuscated gradients of the preprocessing function g. The practical way to experiment such approximation in [12] is to assume $g(x) \approx x$ such that $\bigtriangledown_x g'(x) \approx \bigtriangledown_x g(x)$. Then, the PGD is used to calculate the approximated gradients step by step. We understand there are other possibilities (e.g. [15]) to make such approximation

2. We tested different ensemble sizes for EOT ranging from 2 to 40. The ensemble size has little influence on ASR or ACC. With a larger ensemble size, it is possible to generate AEs with smaller l_2.

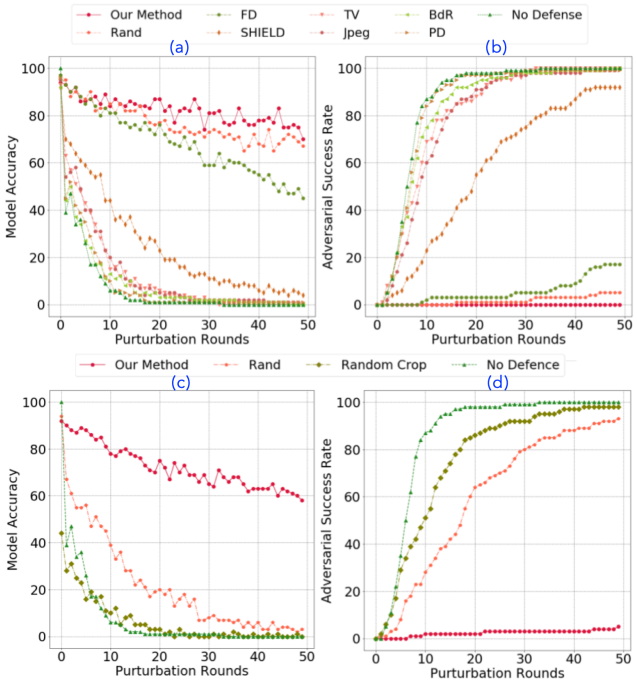

Fig. 4: Defense results on BPDA: ACC (a) and ASR (b) and defense results on BPDA+EOT: ACC (c) and ASR (d).

besides assuming $g(x) \approx x$. The defense evaluation for such case is given in Section 6.4.

For comparison, we re-implemented 7 prior solutions including FD [36], Rand [8], SHIELD [30], TV [6], JPEG [6], BdR [39], and PD [7]. We select these methods because they are all preprocessing-only defense which fits our defense requirements. We give a broader comparison with the defenses that need to alter the target model in Table 5 at the end of this section. Fig. 4 (a) and (b) give the ACC and ASR versus the perturbation rounds.

After 50 attack rounds, the ACC of all the previous solutions except FD drops below 5%, and the corresponding ASR reaches higher than 90%. FD can keep the ASR lower than 20% and the ACC around 40%, which is still not very effective in defending against BPDA. However, our method is particularly effective against the BPDA attack. We can maintain an acceptable ACC (around 70% for 50 attack rounds), and restrict the ASR to almost 0. RAND can also defeat BPDA with a slightly lower ACC than ours. However, it will be broken by the EOT attack, as we will show later. These results are consistent with the l_2 norm and SSIM metrics in TABLE 1: the randomization in those operations causes large variances for one image each time during inference which significantly increase the difficulty for attackers to generate AEs.

We continue the attack until the images with perturbations reach the l_2 bound (0.05). For our method, the adversary needs 231 rounds to reach this l_2 bound with ACC of 57% and ASR of 2%. Therefore, we conclude that our solutions can effectively mitigate the BPDA attack.

6.3 Mitigating BPDA+EOT Attack

Next, we consider a more powerful attack by combining BPDA and EOT [14] which can defeat both shatter gradients

and stochastic gradients based defenses. Here we only consider defense methods that can mitigate the BPDA attack. This gives us two baselines: Rand and Random Crop[3] [6]. Fig. 4 (c) and (d) report ACC and ASR under BPDA+EOT attack. We can observe both Rand and Random Crop fail to mitigate this strong attack: ACC drops to below 20% after 20 rounds, and ASR reaches 100% after 50 rounds. In contrast, our solution can still hold ACC of around 60% and ASR of less than 10% after 50 attack rounds. These results confirm our claims and the effectiveness of our method. We continue the attacks until the images with adversarial perturbations reach the l_2 bound (0.05) and our method can maintain the ACC to 58% and keep the ASR to 7%.

6.4 Mitigating Adaptive BPDA Attack

Fig. 5: (a) Original image I_0. (b) Image produced by our method I_1, (c) Image produced by the approximated neural network I_2. $\|I_1 - I_2\|_2 = 0.22$, $\|I_1 - I_2\|_{SSIM} = 0.35$.

In previous implementation of BPDA attack, we use a naive identity function ($g(x) \approx x$) to approximate the preprocessing function following [12]. However, the adversary can improve the attacks by approximating the transformation with a neural network [15]. Thus, we adopt this adaptive BPDA attack to evaluate our defense method. We use a 6-layer DenseNet auto-encoder (same approximation attack method as [15]) to evaluate our method.

The result is that the attacker cannot find a proper approximation with such an attack. One example is shown in Fig. 5: the approximated image (c) has a large variance compared with the image preprocessed by our method (b) with l_2 norm as 0.22 and SSIM score as 0.35. Thus, such approximation cannot give a useful gradient to generate a successful AE.

We run the end-to-end attack with the trained neural network on 100 images randomly selected from ImageNet and the ASR is 0 under a maximum l_2 norm of 0.05. The average quantitative variance between the approximated image and the image processed by our method for the 100 images are: l_2 norm is 0.16 and the SSIM score is 0.36.

6.5 Mitigating Standard Attacks

We also test our method against standard attacks (I-FGSM, LBFGS, and C&W). An attack succeeds only if the prediction of the model is the targeted class. We use Cleverhans [42] to generate AE of all standard attacks. For FGSM and I-FGSM, AEs are generated under two different l_∞ constraints ($\epsilon = 0.01, 0.03$). I-FGSM is iterated ten times. For LBFGS and C&W, the optimization process is iterated until all targeted AEs are found under l_2 constraint. For LBFGS, the binary search steps are set to 5, and the maximum number of iterations is set to 1000. For C&W, the binary search steps are set to 5, the maximum number of iterations is set to 1000, and the learning rate is 0.1. We evaluate the model accuracy (ACC) and attack success rate (ASR), as well as the l_∞ norm and l_2 norm, TABLE 2 (note that FGSM is a one-step attack and it is not really effective as a targeted attack). Its iterative version I-FGSM with $\epsilon = 0.03$ can reach ASR 95%. Two optimization-based attacks, LBFGS and C&W, can even entirely break the baseline model with 100% ASR.

Attack	l_∞	l_2	Baseline ACC	Baseline ASR
Clean	0.000	0.0000	1.00	Nan
FGSM ($\epsilon = 0.01$)	0.010	0.0099	0.36	0.00
FGSM ($\epsilon = 0.03$)	0.030	0.0294	0.39	0.00
I-FGSM ($\epsilon = 0.01$)	0.010	0.0040	0.13	0.79
I-FGSM ($\epsilon = 0.03$)	0.030	0.0098	0.02	0.95
LBFGS	0.021	0.0013	0.00	1.00
C&W	0.156	0.0162	0.00	1.00

TABLE 2: Standard attacks on baseline model.

The defense results are shown in Table 3. All attacks are conducted as targeted attacks. We randomly select labels that are different from the original ones. Our solution has little influence on the ACC of benign samples. The ASR of those attacks can be kept as 0% and ACC can be maintained as around 90%.

Attack	l_2	No Defense ACC	No Defense ASR	Our method ACC	Our method ASR
No attack	0.0	100%	Nan	95%	Nan
I-FGSM	0.010	2%	95%	93%	0%
LBFGS	0.001	0%	100%	91%	0%
C&W	0.016	0%	100%	87%	0%
PGD-50	0.002	8%	86%	92%	0%
PGD-1000	0.003	0%	100%	92%	0%

TABLE 3: Results of our defenses against standard attacks.

For the distortion limit δ, it has influence on the distortion level of each grid. It also affects the ratio of pixels that will be dropped. We apply a linear search of δ from 0.01 to 0.30, as shown in TABLE 4. The ASR becomes 0% under our defenses, which shows that the adversarial perturbation is delicate to this kind of distortion. A larger δ decreases the ACC on clean examples. Thus, a moderate $\delta = 0.15$ is chosen as the optimal value.

6.6 A Broader Comparison with More Defenses

We compare our solution with a broader set of defenses against bounded attacks. These methods also adopt preprocessing while some of them require model changes, e.g., model retraining (ME-Net) or adversarial training (Crop, JPEG, TV, Quilting, and ME-Net). These methods were proved to be broken partially or entirely by BPDA or BPDA+EOT in [16].

We summarize the analytic results, experimental data as well as conclusions from literature in TABLE 5. The AE

3. Random Crop are not considered in the previous subsection due to its low model usability (30%-40% ACC drop).

Attack	$\delta = 0.01$		$\delta = 0.05$		$\delta = 0.10$		$\delta = 0.15$		$\delta = 0.20$		$\delta = 0.25$		$\delta = 0.30$	
	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
Clean	0.95	Nan	0.96	Nan	0.95	Nan	0.95	Nan	**0.96**	Nan	0.93	Nan	0.91	Nan
FGSM ($\epsilon = 0.01$)	0.70	0.00	0.66	0.00	0.69	0.00	0.73	0.00	0.69	0.00	**0.75**	0.00	0.72	0.00
FGSM ($\epsilon = 0.03$)	0.51	0.00	0.51	0.00	0.51	0.00	0.53	0.00	0.55	0.00	0.55	0.00	**0.62**	0.00
I-FGSM ($\epsilon = 0.01$)	0.96	0.00	0.05	0.00	0.93	0.00	0.89	0.00	0.90	0.00	0.91	0.00	**0.93**	0.00
I-FGSM ($\epsilon = 0.03$)	0.88	0.01	0.90	0.00	0.86	0.00	**0.93**	0.00	0.92	0.00	0.89	0.00	0.89	0.00
LBFGS	0.95	0.00	**0.97**	0.00	0.93	0.00	0.91	0.00	0.94	0.00	0.94	0.00	0.88	0.00
C&W	0.86	0.00	0.87	0.00	0.85	0.00	**0.87**	0.00	0.83	0.00	0.83	0.00	0.84	0.00

TABLE 4: Impact of distortion limits on defense performance of the proposed defense

Solutions	Requirement	Attack	#1	#2	#3	$l_\infty = 0.031$	$l_2 = 0.05$
Rand [8]	◊	EOT	✓		✓	0%	-
PixelDefend [43]	◊, △	BPDA	✓	✓		9%	-
Crop [6]	◊, △	BPDA+EOT		✓		-	0%
JPEG [6]	◊, △	BPDA	✓	✓		-	0%
TV [6]	◊, △	BPDA+EOT	✓	✓		-	0%
Quilting [6]	◊, △	BPDA+EOT	✓	✓		-	0%
SHIELD [30]	◊, △	BPDA	✓	✓		-	0%
PD [7]	◊	BPDA	✓	✓		0%	-
Guided Denoiser [44]	◊	BPDA	✓	✓		-	0%
ME-Net [45]	□, ◊, △	BPDA+EOT		✓	✓	13%	-
FD [36]	◊	BPDA	✓	✓		-	10%
Our method	◊	BPDA+EOT	✓	✓	✓	-	**58%**

TABLE 5: Comparisons with a broader defenses on bounded attacks. (For defense requirements, □: target model modification; ◊: input preprocessing; and △: adversarial training).

generation is either bounded by l_∞ (0.031) or l_2 (0.05). Even combined with adversarial training, most of them cannot provide enough robustness. We can observe that our method shows much better robustness against BPDA+EOT (ACC is as high as 58% under the l_2 bound). We also reveal the satisfactory of the three properties (#1 to #3 in TABLE 5) of those methods. All the defenses in Table 5 can satisfy only part of the properties. Note that ME-Net meets properties #2 and #3 but not #1, as it retrains the model with preprocessed clean samples. We conclude that our three properties are indeed an accurate indicator to reveal the difficulty of adversarial attacks.

7 Discussion and Future Work

In Section 4, we list three properties that instruct us to give the practical defense solution. It is worth noting that some other transformations can also be used as candidates to build similar defense solutions. For instance, if we use FD [36] to make the quantization step or use transformations like Rand [8] to distort the image, the defense solution will still be effective. This is because the FD [36] can also bring defensive quantization and Rand [8] can introduce a large variance as well (see TABLE 1).

Defense	ACC	ASR
Our method	**0.72**	**0.01**
Our quantization + Rand [8]	0.70	0.01
FD [36] + Our image distortion	0.64	0.02
FD [36] + Rand [8]	0.63	0.03

TABLE 6: Compare with different combinations after 50 rounds of BPDA attack.

Such a comparison is listed in TABLE 6. This evaluation proves that our proposed solution has the best performance on maintaining the ACC. Note that replacing our quantization step by FD [36] will decrease the ACC from 0.72 to 0.64 and replacing our image distortion by Rand [8] will decrease the ACC from 0.72 to 0.70. This proves our quantization step and image distortion step have better defense performance than FD [36] and Rand [8], respectively. In summary, the core idea of this paper is to give properties that can help to build effective defense solutions to mitigate advanced adversarial attacks. Therefore, we hope in future work, better transformation functions can be discovered to build stronger defense solutions.

The second perspective worth pointing out is that in this paper we only consider the preprocessing-based defense. Therefore, approaches like adversarial training may be effective to mitigate the similar threats but are not within our scope for evaluation and comparison.

The other future direction is to enhance the attacks from the adversarial perspective. Since more advanced attacks based on BPDA were proposed in [15] and evaluated in 6.4, we hope our methodology can also inspire the invention of other advanced adversarial attacks in the future. One of the potential attacks could be more adaptive attacks by approximating the gradients by targeting partially of the preprocessing function. Such an idea may help to decrease the difficulty of the approximation-based adversarial attacks. We believe such future research directions can help to continue the arms race on the AE research.

8 Conclusion

We propose a novel and efficient preprocessing-based solution to mitigate advanced gradient-based adversarial attacks (BPDA, EOT, their combination, and adaptive attacks). Specifically, we first identify three properties to reveal possible defense opportunities. Following these properties, we

design a preprocessing transformation function to enhance the robustness of the target model. We comprehensively evaluate our solution and compare it with 11 state-of-the-art prior defenses. Empirical results indicate that our solution has the best performance in mitigating all these advanced gradient-based adversarial attacks.

We expect that our solution can heat the arms race of adversarial attacks and defenses, and contribute to the defender's side. The proposed three properties can inspire people to come up with better defenses. Meanwhile, we expect to see more sophisticated attacks that can fully tackle our defenses in the near future. All these efforts can advance the study and understanding of AEs and DL robustness.

ACKNOWLEDGEMENT

We thank the anonymous reviewers for their valuable comments. This work was supported in part by Singapore Ministry of Education AcRF Tier 1 RS02/19. This work was supported in part by National Key Research and Development Plan of China, 2018YFB1800301 and National Natural Science Foundation of China, 61832013.

REFERENCES

[1] J. Soifer, J. Li, M. Li, J. Zhu, Y. Li, Y. He, E. Zheng, A. Oltean, M. Mosyak, C. Barnes *et al.*, "Deep learning inference service at microsoft," in *2019 {USENIX} Conference on Operational Machine Learning (OpML 19)*, 2019, pp. 15–17.

[2] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[3] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[4] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," *arXiv preprint arXiv:1607.02533*, 2016.

[5] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 39–57.

[6] C. Guo, M. Rana, M. Cisse, and L. van der Maaten, "Countering adversarial images using input transformations," in *International Conference on Learning Representations*, 2018.

[7] A. Prakash, N. Moran, S. Garber, A. DiLillo, and J. Storer, "Deflecting adversarial attacks with pixel deflection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8571–8580.

[8] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, "Mitigating adversarial effects through randomization," in *International Conference on Learning Representations*, 2018.

[9] J. Buckman, A. Roy, C. Raffel, and I. Goodfellow, "Thermometer encoding: One hot way to resist adversarial examples," 2018.

[10] H. Qiu, Q. Zheng, T. Zhang, M. Qiu, G. Memmi, and J. Lu, "Towards secure and efficient deep learning inference in dependable iot systems," *IEEE Internet of Things Journal*, 2020.

[11] "A complete list of all (arxiv) adversarial example papers," https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html, 2020.

[12] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," in *International Conference on Machine Learning*, 2018, pp. 274–283.

[13] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, "Synthesizing robust adversarial examples," in *International Conference on Machine Learning*, 2018, pp. 284–293.

[14] F. Tramer, N. Carlini, W. Brendel, and A. Madry, "On adaptive attacks to adversarial example defenses," *arXiv preprint arXiv:2002.08347*, 2020.

[15] E. Raff, J. Sylvester, S. Forsyth, and M. McLean, "Barrage of random transforms for adversarially robust defense," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 6528–6537.

[16] N. Carlini, A. Athalye, N. Papernot, W. Brendel, J. Rauber, D. Tsipras, I. Goodfellow, A. Madry, and A. Kurakin, "On evaluating adversarial robustness," *arXiv preprint arXiv:1902.06705*, 2019.

[17] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2013, pp. 387–402.

[18] Y. Dong, F. Liao, T. Pang, X. Hu, and J. Zhu, "Discovering adversarial examples with momentum," *arXiv preprint arXiv:1710.06081*, 2017.

[19] S.-M. Moosavi-Dezfooli, A. Fawzi, and 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] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European symposium on security and privacy (EuroS&P)*. IEEE, 2016, pp. 372–387.

[21] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 828–841, 2019.

[22] R. Huang, B. Xu, D. Schuurmans, and C. Szepesvári, "Learning with a strong adversary," *arXiv preprint arXiv:1511.03034*, 2015.

[23] U. Shaham, Y. Yamada, and S. Negahban, "Understanding adversarial training: Increasing local stability of supervised models through robust optimization," *Neurocomputing*, vol. 307, pp. 195–204, 2018.

[24] D. Meng and 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.

[25] H. Lee, S. Han, and J. Lee, "Generative adversarial trainer: Defense to adversarial perturbations with gan," *arXiv preprint arXiv:1705.03387*, 2017.

[26] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," *arXiv preprint arXiv:1412.5068*, 2014.

[27] A. S. Ross and 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.

[28] N. Papernot, P. McDaniel, X. Wu, S. Jha, and 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.

[29] N. Carlini and D. Wagner, "Magnet and" efficient defenses against adversarial attacks" are not robust to adversarial examples," *arXiv preprint arXiv:1711.08478*, 2017.

[30] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, S. Li, L. Chen, M. E. Kounavis, and D. H. Chau, "Shield: Fast, practical defense and vaccination for deep learning using jpeg compression," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 196–204.

[31] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, "Synthesizing robust adversarial examples," *arXiv preprint arXiv:1707.07397*, 2017.

[32] A. S. Rakin, Z. He, and D. Fan, "Bit-flip attack: Crushing neural network with progressive bit search," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 1211–1220.

[33] Y. Gao, C. Xu, D. Wang, S. Chen, D. C. Ranasinghe, and S. Nepal, "Strip: A defence against trojan attacks on deep neural networks," in *Proceedings of the 35th Annual Computer Security Applications Conference*, 2019, pp. 113–125.

[34] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," *arXiv preprint arXiv:1705.07204*, 2017.

[35] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[36] Z. Liu, Q. Liu, T. Liu, N. Xu, X. Lin, Y. Wang, and W. Wen, "Feature distillation: DNN-oriented jpeg compression against adversarial examples," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019, pp. 860–868.

[37] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE transactions on Computers*, vol. 100, no. 1, pp. 90–93, 1974.

[38] C. Guo, M. Rana, M. Cisse, and L. Van Der Maaten, "Countering adversarial images using input transformations," *arXiv preprint arXiv:1711.00117*, 2017.
[39] W. Xu, D. Evans, and Y. Qi, "Feature squeezing: Detecting adversarial examples in deep neural networks," in *25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018*. The Internet Society, 2018.
[40] A. Hore and D. Ziou, "Image quality metrics: Psnr vs. ssim," in *2010 20th International Conference on Pattern Recognition*. IEEE, 2010, pp. 2366–2369.
[41] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.
[42] N. Papernot, F. Faghri, N. Carlini, I. Goodfellow, R. Feinman, A. Kurakin, C. Xie, Y. Sharma, T. Brown, A. Roy, A. Matyasko, V. Behzadan, K. Hambardzumyan, Z. Zhang, Y.-L. Juang, Z. Li, R. Sheatsley, A. Garg, J. Uesato, W. Gierke, Y. Dong, D. Berthelot, P. Hendricks, J. Rauber, and R. Long, "Technical report on the cleverhans v2.1.0 adversarial examples library," *arXiv preprint arXiv:1610.00768*, 2018.
[43] Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman, "Pixeldefend: Leveraging generative models to understand and defend against adversarial examples," *arXiv preprint arXiv:1710.10766*, 2017.
[44] F. Liao, M. Liang, Y. Dong, T. Pang, X. Hu, and 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.
[45] Y. Yang, G. Zhang, D. Katabi, and Z. Xu, "Me-net: Towards effective adversarial robustness with matrix estimation," in *International Conference on Machine Learning*, 2019, pp. 7025–7034.

Han Qiu received the B.E. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2011, the M.S. degree from Telecom-ParisTech (Institute Eurecom), Biot, France, in 2013, and the Ph.D. degree in computer science from the Department of Networks and Computer Science, Telecom-ParisTech, Paris, France, in 2017. He worked as a postdoc and a research engineer with Telecom Paris and LINCS Lab from 2017 to 2020. Currently, he is an assistant professor at Institute for Network Sciences and Cyberspace, Tsinghua University, China. His research interests include AI security, data security, and cloud computing.

Yi Zeng received his B.E. degree in Electronic and Information Engineering from the Xidian University, Xi'an, Shannxi, China, in 2019. Currently, he is pursuing his master's degree in Machine Learning & Data Science under the Electronic and Computer Engineering department of the University of California, San Diego, CA, USA. His research interests include AI security, Computer Vision, and Machine Learning.

Qinkai Zheng has earned a bachelor's degree in Information Engineering in 2018 from the SPEIT at Shanghai Jiao Tong University, Shanghai, China. Currently, he is a master student in a double degree program between Shanghai Jiao Tong University and Telecom Paris, Paris, France. His research subject is robust machine learning. His research interests include adversarial learning and graph learning.

Shangwei Guo received the Ph.D. degree in computer science from Chongqing University, Chongqing, China at 2017. He worked as a postdoctoral research fellow at Hong Kong Baptist University and Nanyang Technological University from 2018 to 2020. He is now an associate professor in the College of Computer Science, Chongqing University. His research interests include secure deep learning, secure cloud/edge computing, and database security.

Tianwei Zhang is an assistant professor in School of Computer Science and Engineering, at Nanyang Technological University. His research focuses on computer system security. He is particularly interested in security threats and defenses in machine learning systems, autonomous systems, computer architecture and distributed systems. He received his Bachelor's degree at Peking University in 2011, and the Ph.D degree at Princeton University in 2017.

Hewu Li received his M.S. and Ph.D. degrees in computer science from Tsinghua University, in 2001 and 2004, respectively. He is currently an associate professor and assistant to the Dean of the Institute for Network Sciences and Cyberspace at Tsinghua University. He is also the director of the Wireless and Mobile Network Technology research laboratory, 2009 AsiaFI (Asia Future Internet) Wireless Chairman of the Mobile Working Group. His research interests include mobile wireless network architecture, hybrid satellite-terrestrial networks, broadband wireless access technology, and mobility architecture in next-generation networks.