# On the Security of Randomized Defenses Against Adversarial Samples

Kumar Sharad
NEC Labs Europe
Heidelberg, Germany
k.sharad@gmail.com

Giorgia Azzurra Marson
NEC Labs Europe
Heidelberg, Germany
giorgia.marson@neclab.eu

Hien Thi Thu Truong
NEC Labs Europe
Heidelberg, Germany
hien.truong@neclab.eu

Ghassan Karame
NEC Labs Europe
Heidelberg, Germany
ghassan@karame.org

## ABSTRACT

Deep Learning has been shown to be particularly vulnerable to adversarial samples. To combat adversarial strategies, numerous defensive techniques have been proposed. Among these, a promising approach is to use randomness in order to make the classification process unpredictable and presumably harder for the adversary to control. In this paper, we study the effectiveness of randomized defenses against adversarial samples. To this end, we categorize existing state-of-the-art adversarial strategies into three attacker models of increasing strength, namely blackbox, graybox, and whitebox (a.k.a. adaptive) attackers. We also devise a lightweight randomization strategy for image classification based on feature squeezing, that consists of pre-processing the classifier input by embedding randomness within each feature, before applying feature squeezing. We evaluate the proposed defense and compare it to other randomized techniques in the literature via thorough experiments. Our results indeed show that careful integration of randomness can be effective against both graybox and blackbox attacks without significantly degrading the accuracy of the underlying classifier. However, our experimental results offer strong evidence that in the present form such randomization techniques cannot deter a whitebox adversary that has access to all classifier parameters and has full knowledge of the defense. Our work thoroughly and empirically analyzes the impact of randomization techniques against all classes of adversarial strategies.

## CCS CONCEPTS

• **Security and privacy → Software and application security**;
• **Computing methodologies → Machine learning**; *Computer vision*;

## KEYWORDS

ML security; robustness to adversarial samples; feature squeezing; randomization.

**ACM Reference Format:**
Kumar Sharad, Giorgia Azzurra Marson, Hien Thi Thu Truong, and Ghassan Karame. 2020. On the Security of Randomized Defenses Against Adversarial Samples. In *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security (ASIA CCS '20), June 1–5, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3320269.3384751

## 1 INTRODUCTION

Deep learning (DL) has advanced rapidly in recent years fueled by big data and readily available cheap computation power. Beyond standard machine learning applications, DL has been found extremely useful in numerous security-critical applications such as handwriting recognition, face recognition [40], and malware classification [3, 15, 45]. When used in such applications, recent studies show that DL is particularly vulnerable to adversarial samples, which are obtained from correctly classified samples by adding carefully selected perturbations to fool classifiers [7, 13, 24, 32]. These perturbations are so chosen that they are large enough to affect the model prediction but small enough to go unnoticed (e.g., through a visual check in image recognition applications). Since DL was shown vulnerable to adversarial samples, numerous attacks and defenses have been developed back and forth [28, 30, 34, 44].

While these back-and-forth attacks and defenses have clearly advanced state of the art, it is essential to analyze their robustness in different adversarial models to understand how beneficial they are in making DL more robust. Here, we distinguish between three classes of attacker models depending on the adversary's knowledge with regards to the classifier's details: *blackbox* (a.k.a. non-adaptive), meaning that the adversary only knows public information, *whitebox* (a.k.a. fully adaptive), i.e., the adversary knows full details of the classifier including any defense in place, and *graybox* (a.k.a. semi-adaptive), reflecting partial knowledge of the classifier's internals.

A popular defensive technique utilizes randomness in the classification process, with the hope to enlarge the search space of successful adversarial perturbations. The use of randomness to enhance robustness of DL classifiers has been proposed in many different flavors, both at training and classification time, ranging from randomizing the input to modifying the neural network itself in a randomized fashion. Although many works hint at the potential of such a technique [5, 16, 43, 46], there is still lack of analysis within the community on the robustness of this strategy against state-of-the-art attacks.

**Contributions.** In this work, we study the effectiveness of pre-processing randomized defenses against a wide variety of adversarial strategies, including the strongest whitebox attacks to date. Specifically, we develop a security model to formally define robustness of machine learning algorithms under the various adversarial strategies that populate the literature (cf. Section 3). Our model, inspired by cryptographic definitions of security, is generic and captures a broad variety of machine learning classifiers. To investigate the effectiveness of randomization on the classifier's robustness, we present a light-weight defensive strategy, *Randomized Squeezing*, that combines the prominent pre-processing defense Feature Squeezing [44] with input randomization (cf. Section 4). We also compare Randomized Squeezing with two other instantiations of randomized pre-processing techniques: the *Cropping-Rescaling* defense of Guo *et al.* [16], and *Region-Based Classification* by Cao and Gong [5]. We empirically compare the effectiveness of Randomized Squeezing, Cropping-Rescaling, and Region-Based Classification, against state of the art attack strategies to generate adversarial samples from MNIST, CIFAR-10, and ImageNet datasets (cf. Section 5).

Our proposal embeds randomness within the input to the classifier, operating on every pixel of the image independently by adding randomly chosen perturbations to all pixels, prior to applying the squeezing function. The combination of input randomization and squeezing instantiates a specific pre-processing transformation, similarly to Cropping-Rescaling and Region-Based Classification. For all three randomized techniques, our empirical evaluation shows that introducing an appropriate amount of randomness at pixel level does not significantly hamper accuracy and, in case of Randomized Squeezing, it also improves robustness against graybox adversaries [7, 13, 18, 24, 29, 33] compared to deterministic Feature Squeezing. Our results further highlight that, despite the perturbation induced by randomizing test images, Region-Based Classification and Randomized Squeezing can achieve high accuracy and robustness without transforming training samples. This is in contrast to prior findings [16] hinting that input transformation can be effective against adversarial samples, provided that the same transformation is also applied at training time, and opens the possibility to leverage randomness to realize *online* defenses—which can enhance the robustness of pre-trained classifiers in a flexible and efficient manner.

To further evaluate the three defenses in the whitebox model, we consider the strongest currently known attacks, tuned for each defense: the Backward Pass Differentiable Approximation (BPDA) [1] and the Expectation Over Transformation (EOT) [2]. Our results indicate that while these adaptive attacks defeat all three defenses, increasing the amount of randomness used by the defense results in a higher number of iterations, respectively, larger perturbations, necessary for the attacks to succeed. This suggests that even in the case of whitebox attacks, randomness may have a positive, although limited, impact on the classifier's robustness—in the sense of forcing the BPDA and EOT attacks to invest greater effort to craft high-confidence adversarial samples. Our results also support the intuition that introducing unpredictability to the classification process makes it more challenging, for state-of-the-art adaptive attacks, to find adversarial perturbations which are successful regardless of the randomness.

It is therefore plausible that some DL applications, which can reasonably constrain the attacker by limiting the distortion and/or requiring that adversarial samples be generated in real time, may safely employ randomized classifiers even against (properly constrained) whitebox attacks.

As far as we are aware, this is the first work that comprehensively analyses the impact of randomness on DL classifiers under all state-of the art adversarial strategies, covering also the whitebox attacker model.

## 2 BACKGROUND

In this section, we introduce notation and relevant concepts for the subsequent sections.

**Notation & Conventions.** Let $X$ be a (finite) set, and let $\mathcal{D}\colon X \to [0,1]$ be a probability distribution. We denote by $x \leftarrow_{\mathcal{D}} X$ the random sampling of an element $x$ according to distribution $\mathcal{D}$; we write $x \leftarrow_{\$} X$ for sampling $x$ uniformly at random. We denote by $f\colon X \to Y$ the function defining the classification problem of interest (a.k.a. "ground truth"), where $X$ and $Y$ are the sets of instances and of corresponding classes (or labels), respectively. A machine-learning classifier $\mathcal{C}$ is an algorithm that aims at emulating function $f$. Typically, the classifier is deterministic and can be thus thought of as a function itself. In this work, we cover a broader class of classifiers and let $\mathcal{C}$ be any, possibly randomized algorithm. If $\mathcal{C}$ is randomized, we write $y \leftarrow_{\$} \mathcal{C}(x)$ to denote that on input $x$ the classifier, run on freshly sampled randomness, outputs label $y$. Let $\mathcal{C}\colon X \to Y$ be a deterministic classifier. For $X' \subseteq X$ we denote by $X'_{\checkmark}(\mathcal{C}) = \{x \in X' : \mathcal{C}(x) = f(x)\}$ the set of instances in $X'$ where $\mathcal{C}$ agrees with $f$. Similarly, we denote by $X'_{\text{✗}}(\mathcal{C}) = \{x \in X' : \mathcal{C}(x) \neq f(x)\}$ the set of misclassified instances. Using this notation, we measure a classifier's *(empirical) accuracy*, respectively, *(empirical) error* w.r.t. a given dataset $D = \{(x, f(x)) : x \in X^D\}$, for some $X^D \subset X$ as $\mathrm{acc}_{X^D}(\mathcal{C}) := |X^D_{\checkmark}(\mathcal{C})|/|X^D|$ and $\mathit{err}_{X^D}(\mathcal{C}) := |X^D_{\text{✗}}(\mathcal{C})|/|X^D|$, respectively. For randomized classifiers, the definitions of accuracy and error need to be augmented for incorporating the randomness of the classifier. Note that accuracy and error can also be used to capture the performance of a classifier w.r.t. an *adversarially chosen* distribution $\mathcal{D}^{\mathcal{A}}$, respectively, input set $X^{\mathcal{A}}$.

### 2.1 Adversarial Samples

The accuracy of a classifier is measured w.r.t. samples drawn from a 'natural' distribution $\mathcal{D}\colon X \to [0,1]$ over the input space. This approach is grounded in results from computational learning theory [41], which guarantee a low classification error as long as samples used at test time originate from the *same distribution* of the training samples. While this assumption may be realistic in a pure machine-learning setting, it is hard to justify in general. In cybersecurity, e.g., the "test samples" are generated by an adversary $\mathcal{A}$ attempting to bypass an ML protected system, and may thus be specifically crafted to deviate from the training samples. This state of affair has been confirmed by the recent advances in attacking ML systems through *adversarial samples* [39].

An adversarial sample $x'$ is derived from a labeled sample $(x, y)$ by slightly perturbing $x$, so that $x'$ still belongs to the original class $y$, yet it is classified wrongly. Formally: $x$ and $x'$ have a relatively small distance $d(x, x') \leq \epsilon$, $f(x') = y$, and $\mathcal{C}(x') \neq y$. The tolerated amount of perturbation $\epsilon$ is called *distortion* (a.k.a. adversarial budget). The three most common metrics to measure the distance between an adversarial sample $x'$ and its legitimate counterpart $x$ are based on the $L^p$-norms ($L^0$, $L^2$, and $L^\infty$): (i) $d^0(x, x') = |\{i : x_i - x'_i \neq 0\}|$, based on the number of modified pixels; (ii) $d^2(x, x') = \left(\sum_i (x_i - x'_i)^2\right)^{\frac{1}{2}}$, based on the Euclidean distance; (iii) $d^\infty(x, x') = \max_i(x_i - x'_i)$, based on the maximum difference between pixels at corresponding positions, where $x_i - x'_i$ is the difference between pixels at position $i$ of images $x$ and $x'$, respectively. For a distance metric $d^p$, we denote by $|| \cdot ||_p$ the corresponding norm. Depending on the attacker's goal, adversarial samples can be categorized as *targeted* and *untargeted*. A targeted adversarial sample $x'$ is successful if the classifier's prediction matches an attacker-chosen label $y_t \neq y$, where $y$ is the true label. An untargeted adversarial sample instead succeeds if the classifier predicts any label other than $y$.

Prominent techniques to generate adversarial examples against DL classifiers are the Fast Gradient Sign Method (FGSM) [13], the Basic Iterative Method (BIM) [24], the Jacobian Saliency Map Approach (JSMA) [24], the Carlini-Wagner (CW) attacks [7], and DeepFool [29]. Among others, we will consider these attack strategies in our evaluation (cf. Section 5).

### 2.2 Defensive Techniques

Here we discuss the defenses against adversarial samples which are most relevant to our work. We survey more defensive techniques in Section 6.

**Feature Squeezing.** This technique, introduced by Xu *et al.* [44], transforms the input by reducing unnecessary features while keeping the DL model intact. Feature Squeezing is a generic transformation technique to reduce feature input space such that it can limit opportunities for an adversary to generate adversarial samples. The approach assumes that legitimate samples have same output on original and transformed form while adversarial samples have larger difference on outputs, the discrepancy of outputs helps to reject adversarial samples. In this paper, we study the two proposed squeezing techniques, *squeezing color bit depths* and *spatial smoothing*.

Squeezing color bits relies on the assumption that large color bit depth is not necessary for a classifier to interpret an image. The authors consider 8-bit gray scale images of size $28 \times 28$ pixels (MNIST dataset) and 24-bit color images of size $32 \times 32$ pixels (CIFAR-10 and ImageNet datasets) in their experiments. The 8-bit gray scale images are squeezed to 1-bit monochrome images by using a binary filter with cut-off set to 0.5, while each channel of the 24-bit color images (8-bit per color channel) is squeezed to 4 or 5 bits. Each channel can be reduced to $i$-bit depth by multiplying the input value with $2^i - 1$, rounded up to integers and then divided by $2^i - 1$ to scale back to [0,1].

The local smoothing is a type of spatial smoothing technique that adjusts the value of each pixel based on aggregated values, e.g., by taking median of its neighborhood pixels. The median smoothing technique follows Gaussian smoothing design. The values of neighborhood pixels are decided by a configurable sliding window of which size ranges from 1 to entire image. Experiments in [44] show that median smoothing with $2 \times 2$ and $3 \times 3$ sliding window is effective. Another way to perform spatial smoothing is non-local smoothing. Non-local smoothing considers a large area to compute replacement value for each pixel. Given an image patch, non-local smoothing searches for all similar patches and replaces the center patch with the average of similar patches. We use the notation proposed by Xu *et al.* [44] to denote a filter as "non-local means (a-b-c)", where $a$ is the search window $a \times a$, $b$ the patch size $b \times b$ and $c$ the filter strength.

**Randomness-Based Defenses.** The literature features a number of strategies to use randomness for enhancing DL classifiers against adversarial samples. Zhou *et al.* [46] propose two ways to use randomness for strengthening deep neural-network (DNN) models: to add random noise to the weights of a trained DNN model, and to select a model at random from a pool of train DNN models for each test input. Xie *et al.* [43] use randomness in a different way: to resize the image to a random size, or to add padding zeroes in a randomized fashion. We discuss two other existing randomization strategies which will be later considered in our evaluation, namely *region-based classification* [5] and *cropping-rescaling* [16], along with our proposal in Section 4.

## 3 SECURITY MODEL

In this section, we present a security model for evasion attacks that allows us to formalize *robustness* to adversarial samples.

**Game-based Modeling of Evasion Attacks.** Our model considers an adversary $\mathcal{A}$ that aims at defeating a classifier $\mathcal{C}$ by generating adversarial samples starting from "natural" samples. Following the approach of modern cryptography, our security model reproduces the above scenario through a security game between $\mathcal{A}$ and $\mathcal{C}$, that we name *evasion under chosen-sample attacks* (EV-CSA), as illustrated in Figure 1.

The adversary's goal is to present a number of adversarial samples generated from a set $X^D \subset X$ of "naturally occurring" (labeled) samples. [1] The number $N$ of adversarial samples,

[1]In practice, $X^D$ represents a set of available images used for testing, e.g., MNIST.

```
Game EV-CSA$_{\mathcal{A},\epsilon,N}(\mathcal{C}, X^D)$:
  $q \leftarrow 0,\ n \leftarrow 0$
  $X^{\mathcal{A}} \leftarrow \emptyset$
  $\mathcal{A}(\epsilon, N, \langle\mathcal{C}\rangle, X^D)^{\mathsf{Classify},\mathsf{Attack}}$
  Return $n/N$

Oracle Classify$(x)$:
  $\hat{y} \leftarrow \mathcal{C}(x)$
  Give $\hat{y}$ to $\mathcal{A}$

Oracle Attack$(x, x', y_t)$:
  If $q \geq N$: Go to line 4
  Enforce $(x, *) \notin X^{\mathcal{A}}$
  $q \leftarrow q + 1$
  $X^{\mathcal{A}} \overset{\cup}{\leftarrow} (x, x')$
  If $\mathcal{C}(x') = y_t$ and $d(x', x) \leq \epsilon$:
      $n \leftarrow n + 1$
  Return
```

**Figure 1: Security game for targeted evasion under chosen-sample attacks (EV-CSA), involving adversary $\mathcal{A}$ against classifier $\mathcal{C}$. Untargeted attacks are captured by replacing the inputs to the** Attack **oracle with pairs $(x, x')$ and, the first condition of line 11 with $\mathcal{C}(x') \neq \mathcal{C}(x)$.**

$1 \leq N \leq |X^D|$, is a game parameter and can be adapted to capture different security goals.

We specify the amount of information that $\mathcal{A}$ has about the adversarial task by passing the relevant inputs: the allowed adversarial perturbation (a.k.a. distortion) $\epsilon$, the number $N$ of adversarial samples, the classifier's code $\langle\mathcal{C}\rangle$, and the set $X^D$ of benign samples. Further, by limiting the amount of information encoded in $\langle\mathcal{C}\rangle$, our game can cover different adversarial models such as "whitebox" (a.k.a. fully adaptive), meaning that $\mathcal{A}$ knows every detail about the classifier, including neural-network weights and any defense mechanism in place, "blackbox" (a.k.a. non adaptive), i.e., $\mathcal{A}$ knows only public information about $\mathcal{C}$, and intermediate attacker's models, so-called "graybox" (a.k.a. semi-adaptive), in which $\mathcal{A}$ has only partial information about the classifier's internals and/or defensive layers. Graybox attacks include those agnostic of a defense mechanism. In this case, the adversary knows the original classifier fully, hence it is not blackbox, but it does not know the defense, hence it is not whitebox either (cf. Section 4.4).

We further let the adversary interact with the classifier through an oracle Classify, i.e., $\mathcal{A}$ can query $\mathcal{C}$ on any input $x$ of their choosing and obtain the corresponding label $\hat{y} = \mathcal{C}(x)$. Observe that the Classify oracle provides no extra power to whitebox adversaries, as having full knowledge of the classifier allows $\mathcal{A}$ to emulate the oracle. It is, however, necessary to cover weaker attacks, such as transferability attacks [32] (which are blackbox), and attacks oblivious of the defense (which are graybox).

The game also provides the adversary with a second oracle, denoted by Attack, which lets $\mathcal{A}$ submit candidate adversarial samples. This oracle allows us to describe $\mathcal{A}$'s goal formally and to define *robustness* to adversarial samples, as we see next. The adversary can present an adversarial sample by submitting a query $(x, x', y_t)$ to the Attack oracle, where $x$ is the starting sample, $x'$ is the candidate adversarial sample, and $y_t$ is the target label. Upon being queried, the oracle then checks whether the adversary reached the query limit $q \geq N$, terminating the game in such a case (cf. line 4). Otherwise, it checks whether the adversarial sample $x'$ is "fresh", in the sense that no other adversarial sample $x''$ has been already proposed for the same starting image $x$ (cf. line 8), which is necessary to invalidate trivial attacks that artificially achieve high success rate, e.g., by presenting "the same" adversarial sample over and over, in a trivially modified version.[2] If the query gets through the checks, the oracle adds the fresh pair $(x, x')$ to the adversarial set $X^{\mathcal{A}}$, and further checks whether the classifier errs on $x'$ as desired, by predicting its class as $y^t$, and whether $x'$ is sufficiently close to $x$, i.e., $d(x, x') \leq \epsilon$ according to some pre-established distance metric $d$. In case of success, the game rewards the adversary by increasing the counter $n$ which records the number of successful samples (cf. line 12).

As soon as the $N$ "chosen samples" are submitted, the EV-CSA game ends outputting the success rate of the adversary, that we denote by $\mathbf{succ}^{\text{ev-csa}}_{\mathcal{A},\epsilon,N}(\mathcal{C}, X^D) = n/N$. An execution of the EV-CSA game depends on the adversarial strategy $\mathcal{A}$ and the classifier $\mathcal{C}$—both of which may be randomized. In particular, if the game depends on any randomness (used by the adversary, by the classifier, or both), the outcome is determined by the value of the randomness, and the success rate is a random variable.

**Deterministic vs. Randomized Classifiers.** We stress that oracle Attack does not reflect an actual capability of the attacker, however, it provides a natural abstraction for determining $\mathcal{A}$'s success rate. In particular, if only deterministic classifiers were considered, having $\mathcal{A}$ submit their samples through the oracle is equivalent to letting $\mathcal{A}$ present a set $X^{\mathcal{A}}$ of $N$ samples directly. That is, the usual notion of success rate against deterministic classifiers is a special case of our notion. The reason for introducing the Attack oracle is precisely that, when *randomized* classifiers are considered, it is no longer meaningful to talk about a *set* of adversarial examples (a given sample $x'$ may be correctly labeled for some choices of $\mathcal{C}$'s randomness while being misclassified for a different randomness).

**Defining Robustness.** Our security game provides a formal language to express the effectiveness of a defense in making a given classifier "more robust" to attacks. For a classifier $\mathcal{C}$, let $\mathcal{C}^d$ denote the classifier obtained from $\mathcal{C}$ by applying a defense $d$. Intuitively, a defense is effective against an attack $\mathcal{A}$ if either $\mathcal{A}$'s success rate after applying the defense is significantly smaller than that with no defense, or a larger distortion is necessary to achieve that success rate. Formally, we say that

[2]Changing a few pixels of a successful adversarial sample $x'$ yields a new sample $x'' \neq x'$ which is very likely to also be successful, thus $\mathcal{A}$ should only get credit for one of them.

a defense $d$ for classifier $\mathcal{C}$ is *effective* against attack $\mathcal{A}$, or equivalently that $\mathcal{C}^d$ is *more robust* than $\mathcal{C}$, if either $\mathbf{succ}^{\text{ev-csa}}_{\mathcal{A},\epsilon,N}(\mathcal{C}^d) \ll \mathbf{succ}^{\text{ev-csa}}_{\mathcal{A},\epsilon,N}(\mathcal{C})$, or $\mathbf{succ}^{\text{ev-csa}}_{\mathcal{A},\epsilon^d,N}(\mathcal{C}^d) = \mathbf{succ}^{\text{ev-csa}}_{\mathcal{A},\epsilon,N}(\mathcal{C})$ for $\epsilon^d \gg \epsilon$.

## 4 RANDOMNESS-BASED DEFENSES

In this section, we present the three randomness-based defensive techniques which we consider in our empirical evaluation from Section 5. While they all apply a randomized pre-processing layer at test time, the first defense, Cropping-Rescaling [16], also operates on the training phase, thereby leading to an “offline” defense. It further uses an ensembling technique, meaning that classification is based on the model predictions over an ensemble of samples generated from the original input. The second defense, Region-Based Classification [5], does not modify the training phase—i.e., it is “online”—but uses ensembling, too. The third defense, Randomized Squeezing (our design), neither alters training nor relies on ensembling. Instead, it combines the randomness layer with a subsequent image-denoising operator based on Feature Squeezing [44].

### 4.1 Cropping-Rescaling

The *Cropping-Rescaling* defense by Guo *et al.* [16] applies a randomized transformation that crops and rescales the image prior to feeding it to the classifier. The intuition behind the defense is to alter the spatial positioning of the adversarial perturbation, so that it no longer causes the desired effect and, therefore, it makes the corresponding adversarial sample less likely to succeed. More precisely, cropping-rescaling operates in two steps, at training and at test time. Training is performed on cropped and rescaled images, following the data augmentation paradigm of He *et al.* [17]. Then, to predict the label of each test image, the classifier randomly samples 30 crops of the input, rescales them, and averages the model predictions over all crops. Applying the input transformation also at training yields higher classification accuracy on adversarial samples [16].

### 4.2 Region-Based Classification

The *Region-Based Classification* defense proposed by Cao and Gong [5] computes each prediction over an ensemble generated from the input sample in a randomized fashion. Specifically, this approach samples 10,000 images uniformly at random from an appropriately sized hypercube centered at the testing image, invokes a DNN to compute predictions over the sampled images, and returns the label predicted for the majority of the images—therefore, classification is no longer “point-based” but “region-based”. Here, an “appropriate size” of the hypercube is chosen so that the region-based classifier maintains the accuracy of the underlying DNN over a (benign) test set. Taking the “majority vote” over the ensemble predictions is based on the assumption that while for benign images most neighboring samples yield the same predicted label, adversarial samples are close to the DNN’s classification boundary.

### 4.3 Randomized Squeezing

The defensive strategy that we propose, *Randomized Squeezing*, combines input randomization with a deterministic image-denoising technique, namely the Feature Squeezing defense by Xu *et al.* [44] (cf. Section 2.2). We introduce randomness at feature level, for each feature component and within a predefined threshold, so that it does not bias the prediction excessively in any particular direction. Concretely, let $\mathcal{C}$ denote a DL classifier enhanced with Feature Squeezing. Our proposal preprocesses $\mathcal{C}$’s input by adding a perturbation *rand*, chosen uniformly at random from the real interval $[-\delta, +\delta]$, $\delta \in [0, 1]$, to each pixel. The intuition here is that adding a small, carefully crafted perturbation preserves the classifier’s output on genuine images, and it significantly affects predictions on adversarial images. While the randomness added at individual feature level does not destroy the patterns of the pixels, which is critical for correct classification, it does introduce a source of unpredictability in the defense mechanism which enlarges the search space of the adversary. Indeed, to craft a successful adversarial sample, the adversary now has to search for a perturbation that yields the desired prediction for (most of) the various possible randomness values, which is a considerably harder task than fooling (deterministic) Feature Squeezing. The increased robustness achieved by randomized classifiers clearly depends on the quality of the randomness, which should be unpredictable from the adversary’s perspective. Thus, it is crucial for security that the random noise be generated from a high-entropy key to seed the underlying cryptographic pseudo-random generator. More specifically, Randomized Squeezing comprises of the following subroutines:

**Setup:** This procedure performs any instruction needed to initialize the original system. In addition, it sets the value $\delta \in [0, 1]$ for the magnitude of the randomness (setting $\delta = 0$ leaves the input unchanged, while $\delta = 1$ is almost equivalent to generating a fresh input uniformly at random), and initializes the random number generator. The perturbation magnitude $\delta$ should be sufficiently large to be effective against adversarial samples, and at the same time be sufficiently small to preserve the classifier’s accuracy on normal samples. In Section 5, we analyze in details how to choose $\delta$ in order to establish a good tradeoff between the achieved accuracy and robustness.

**Training:** Since our defense mechanism does not affect the training phase, this step is the same as for the original system. Upon completion of this phase, we can assume a trained (deterministic) classifier $\mathcal{C}$, based on Feature Squeezing, which we will use as a basis for our randomized classifier $\mathcal{C}_\$$, as we see next.

**Classification:** Upon receiving an input image $x$, the randomized classifier $\mathcal{C}_\$$ selects a uniformly random key $k_s$ to seed the underlying pseudo-random generator and expands $k_s$ until a sufficient amount of (pseudo)randomness has been generated to randomize all pixels of $x$. The randomization of each pixel consists in adding a random value $rand \in [-\delta, +\delta]$. When the value of the pixel goes outside the allowed intensity threshold (normalized to [0,1] in our experiments), we clip them at the edges instead of taking a modulo and wrapping around. This is performed to bias the randomness for pixels that are close to the intensity thresholds, which helps to preserve accuracy of the classifier for legitimate samples. The process is repeated for every color channel. Hence, for grayscale images, we add randomization just once as there is only one channel, while for color RGB images randomness is added three times, once for each channel (i.e., "R", "G", and "B", respectively), individually per pixel. The pre-processing routine of Randomized Squeezing is depicted in Figure 2. Finally, the resulting image $x'$ is fed to the (deterministic) classifier $\mathcal{C}$, and the resulting prediction $\hat{y} = \mathcal{C}(x')$ is returned as label for $x$.

```
$Randomize_{pixels}(Pixels, \delta)$
    $k_s \leftarrow_\$ \{0,1\}^{\mathsf{keylen}}$
    for $i \leftarrow 1$ to $length[Pixels]$
        do Generate $nonce_i$
           $rand = G(k_s, nonce_i)$
           ▷ Choose $rand$ randomly from $[-\delta, \delta]$
           $Pixels[i] \leftarrow Pixels[i] + rand$
           if $Pixels[i] > 1$
               do $Pixels[i] = 1$
           if $Pixels[i] < 0$
               do $Pixels[i] = 0$
           ▷ Clip pixel values to allowed threshold
           $i \leftarrow i + 1$
```

**Figure 2: Randomizing image pixels via Randomized Squeezing.** $Pixels$ **represents a vector comprising all pixels of the input image,** $G$ **denotes a pseudo-random number generator, and** $nonce_i$ **is a fresh nonce for every** $i$**.**

Note that we introduce randomness to the input only while testing and not while training: this makes our technique particularly lightweight and versatile, as it does not increase training costs and can be applied directly to any pre-trained classifier. In addition, Randomized Squeezing invokes the underlying model only once per prediction, without relying on ensembling, which also improves efficiency compared to Cropping-Rescaling and Region-Based Classification.

### 4.4 Analysis: Attacks' Categorization

We briefly discuss the attacks considered in our evaluation of randomized defenses (cf. Section 5) in the context of the attacker models from the previous section.

**Graybox Attacks.** Prominent techniques to generate adversarial examples against DL classifiers are the Fast Gradient Sign Method (FGSM) [13], the Basic Iterative Method (BIM) [24], the Jacobian Saliency Map Approach (JSMA) [24], the Carlini-Wagner (CW) attacks [7], and DeepFool [29]. These attacks were specifically designed to fool neural networks in a whitebox setting, hence they assume that architecture and parameters are known to the attacker. Generating adversarial samples according to any of the aforementioned attacks, and then using these samples against an enhanced version of the neural network via some defense mechanism,[3] results in a *graybox* attack—because the neural network's internals are available to the attacker, but the defense is not.

**Whitebox Attacks.** Athalye *et al.* [2] proposed the Expectation over Transformation (EOT), a generic method to generate adversarial samples that remain adversarial over a chosen distribution of transformations—in particular over randomized ones. EOT can handle only differentiable transformations, hence it is not applicable to image-denoising defenses such as Feature Squeezing. A second technique, Backward-Pass Differentiable Approximation (BPDA) [1], was introduced to cope with non-differentiable transformations and later employed to defeat, among others, a generalized version of Feature Squeezing [8]. We evaluate Cropping-Rescaling, Region-Based Classification, and Randomized Squeezing against the BPDA and EOT attacks, individually and in combination, appropriately tuned to each defense. Due to exploiting knowledge of the underlying neural-network parameters and of the defense fully, both BPDA and the combination BPDA+EOT yield whitebox (i.e., "fully adaptive") attacks.

## 5 EVALUATION AND RESULTS

In this section, we evaluate the randomness-based defenses presented in Section 4 against graybox and whitebox attacks. Specifically, we compare Feature Squeezing [44] and Randomized Squeezing by testing them against 11 state-of-the-art graybox attacks, showing that randomness hardens Feature Squeezing. We further evaluate all three randomness-based defenses against the whitebox attacks proposed by Athalye *et al.* [1], and explore how increasing the amounts of randomness affects their success.

### 5.1 Setup

**Attacks.** As proposed by Xu *et al.* [44], we analyze two variations of each targeted attack: (i) next: targets the class next to the ground truth class modulo number of classes (ii) least-likely (LL): targets the class which the image is least-likely to be classified as. Specifically, we consider the following attacks on Feature Squeezing and Randomized Squeezing: Fast Gradient Sign Method (FGSM) [13], Basic Iterative Method (BIM) [24], Carlini and Wagner $L^0$, $L^2$ and $L^\infty$ attacks (CW) [7] (Next & LL), DeepFool [29], Jacobian Saliency Map Approach (JSMA) [33] (Next & LL). We further evaluate Randomized Squeezing, Cropping-Rescaling, and Region-Based Classification, against fully adaptive attacks (BPDA and EOT) proposed by Athalye *et al.* [1].

**Datasets.** We use ImageNet, CIFAR-10, and MNIST datasets to conduct our experiments. The ImageNet dataset contains 1.2 million images for training and 50 000 images for validation. They are of various sizes and hand-labeled with 1000 classes. The images are preprocessed to $224 \times 224$ pixels and encoded

[3]Here: Cropping-Rescaling, Region-Based Classification, or Randomized Squeezing.

with 24-bit color per pixel. CIFAR-10 is a dataset of $32 \times 32$ pixel images with 24-bit color per pixel (three color channels per pixel) and 10 classes. MNIST is a dataset of hand-written digits (0-9) encoded as 8-bits grayscale images of size $28 \times 28$ pixels (one color channel per pixel).

**Target Models.** We tested the aforementioned attacks on the same pre-trained models as in Feature Squeezing [44]. Namely, we use MobileNet [20] for ImageNet, a 7-layer CNN for MNIST[4], and a DenseNet model for CIFAR-10[5] [21]. These models achieve a top-1 accuracy of 99.43%, 94.84%, and 68.36%, respectively. The prediction performance of these models is at par with best models[6]. To study the effects of introducing randomness, we use the same data samples as used by Xu *et al.* [44]. We use 100 adversarial samples for each of the 11 attacks for all datasets. Each color channel of the pixel is normalized to be in the range $[0, 1]$. We use 10 000 legitimate samples for CIFAR-10 and MNIST, and 200 samples for ImageNet (due to high computation cost) to study the effect of adding randomness to the defenses.

**Experimental Setup.** We evaluate the efficacy of the 3 defenses proposed by Xu *et al.* [44]—bit depth reduction, median smoothing and non-local smoothing, when combined with randomness. We study each of the 11 attacks against the 3 defenses with varying parameters for 3 datasets. The experiment is repeated 200 times for each randomness level to compute the statistics. In our evaluation, the accuracy over adversarial samples is averaged over 200 runs. We note that for the deterministic case when no randomness is added ($\delta = 0$) the results do not change.

**Implementation.** We implemented Randomized Squeezing in Python and executed it using CPython. Namely, we adapted the open source code released by Xu *et al.*[7] to implement our solution. We use a machine with 3.5 GHz processor and 32 GB RAM for our experiments.

## 5.2 Graybox Adversaries

This section presents the results of our evaluation of Randomized Squeezing, compared to the deterministic Feature Squeezing, against graybox attacks.

**Choosing $\delta$.** Choosing the randomness magnitude $\delta$ appropriately is vital to designing an effective defense. The choice depends upon the nature of dataset and defense used. As we show in the paragraphs ahead the behavior of defenses can vary for grayscale and color images. We study these variations extensively by running experiments for changing $\delta$. We present our evaluation of ImageNet, CIFAR-10 and MNIST datasets next.

**ImageNet.** Figure 3 shows the behavior of accuracy of the classifier for both adversarial and legitimate samples as input randomization ($\delta$) is increased. The accuracy decreases as $\delta$ is increased. Squeezing via reducing bit-depth shows a drastic drop in accuracy beyond a certain randomness for most attacks; as $\delta$ increases, a larger fraction of pixels breach the quantization threshold which results in them being flipped to 0 or 1 despite being very distant earlier, this results in accuracy dropping sharply. The CW0 adversarial samples show an improvement in accuracy for high $\delta$, this is due to large $L^0$ perturbations being undone due to noise. Median smoothing methods are less affected by large randomness values due to the randomness being averaged out. Hence we observe a gradual decline in accuracy with increasing randomness. Note that CIFAR-10 has color and each pixel has three color channels each of which are normalized to $[0, 1]$. We introduce randomness individually to each channel for each pixel, hence the effect of randomness becomes significant even at low $\delta$ values.

We conclude that an appropriately chosen $\delta$ provides desirable security properties, we found that $\delta = 0.1$ provides the best trade-off between accuracy on legitimate and adversarial samples. We present the accuracy values in Table 2. We note that the accuracy decreases slightly over legitimate samples. For comparison, Table 1 shows the accuracy of defenses when no randomness is added [44].

We further compare the robustness of Randomized Squeezing with that of the other two randomized defenses, Cropping-Rescaling and Region-Based Classification. As advocated in [16], Cropping-Rescaling achieves an accuracy of 45-65% against FGSM, while Region-Based Classification and Randomized Squeezing achieve an accuracy of 34.7% and 53.44% (with median smoothing), respectively. Similarly, for CW2 Next (Next and LL), Cropping-Rescaling achieves an accuracy of 40-65% (adapted from [16]), while Region-Based Classification and Randomized Squeezing achieve an accuracy of 79% and 70% (with median smoothing), respectively. This shows that online randomization defenses offer decent robustness to gray-box attacks even when compared to offline defenses.

**CIFAR-10.** The results for running Randomized Squeezing within the CIFAR-10 dataset are similar to ImageNet: adding randomness helps make misclassified samples unpredictable. Figure 4 shows that accuracy over both legitimate and adversarial samples drops sharply on increasing $\delta$. Identical to our evaluation in the ImageNet dataset, we note that Randomized Squeezing introduces small randomness for each color channel. The accuracy over adversarial samples improves significantly at $\delta = 0.05$ for almost all defenses with just a small drop over legitimate samples (cf. Tables 1 & 2). Large values of $\delta$ make the classifier unusable as accuracy drops.

**MNIST.** As seen in Figure 5, the results of running Randomized Squeezing with the MNIST dataset are similar to ImageNet and CIFAR-10. We present the accuracy values in Tables 1 & 2. The behavior of accuracy for the MNIST dataset is similar to that of ImageNet and CIFAR-10 (cf. Figures 4).

Figure 6 shows the probabilities of the prediction errors when used with $3 \times 3$ median smoothing with and without randomness. Each row represents a specific attack strategy $\mathcal{A}$; the $x$-axis represents the adversarial sample set $X^{\mathcal{A}}$ (in particular $|X^{\mathcal{A}}| = 100$); the color intensity of each cell indicates

[4]https://github.com/carlini/nn_robust_attacks/
[5]https://github.com/titu1994/DenseNet/
[6]http://rodrigob.github.io/are_we_there_yet/build/classification_datasets_results.html
[7]https://github.com/mzweilin/EvadeML-Zoo

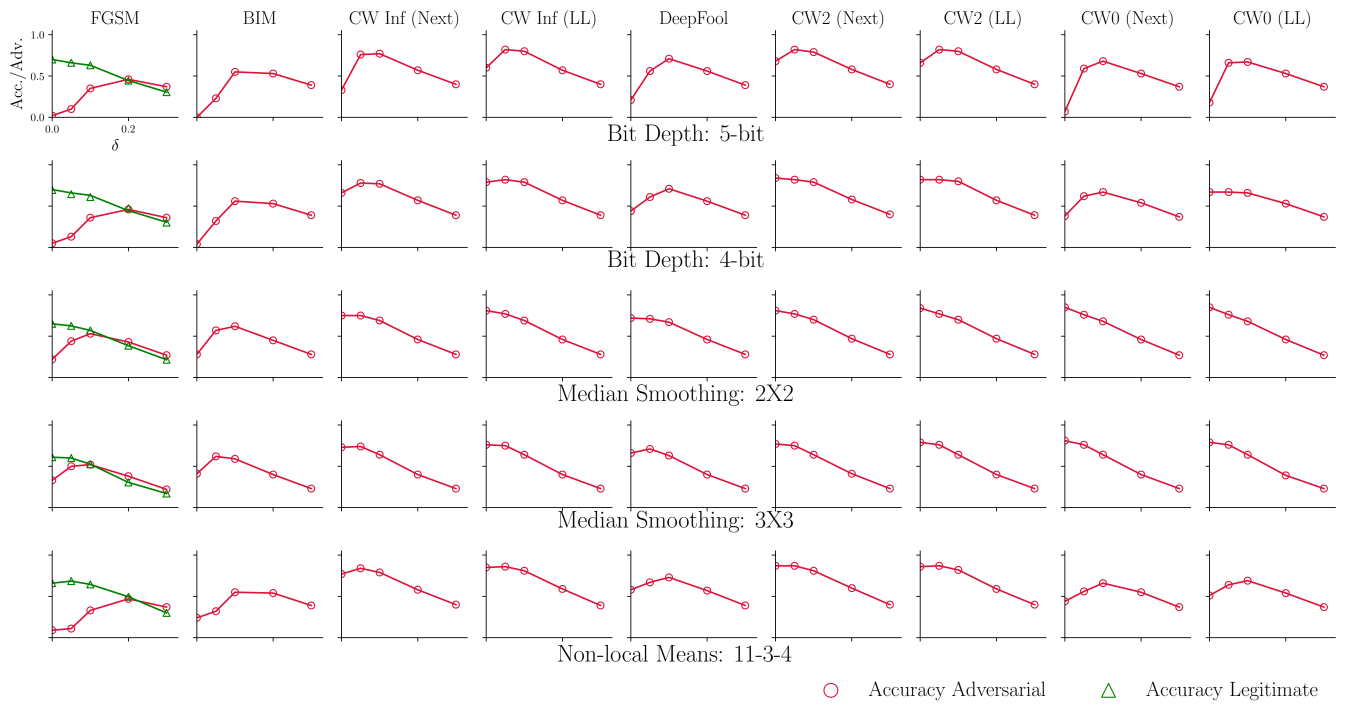

**Figure 3: ImageNet: Behavior of accuracy for magnitudes of randomness $\delta = [0, 0.05, 0.1, 0.2, 0.3]$. We also plot the accuracy of the model for legitimate samples as $\delta$ increases (shown once for each defense as the curve does not change).**

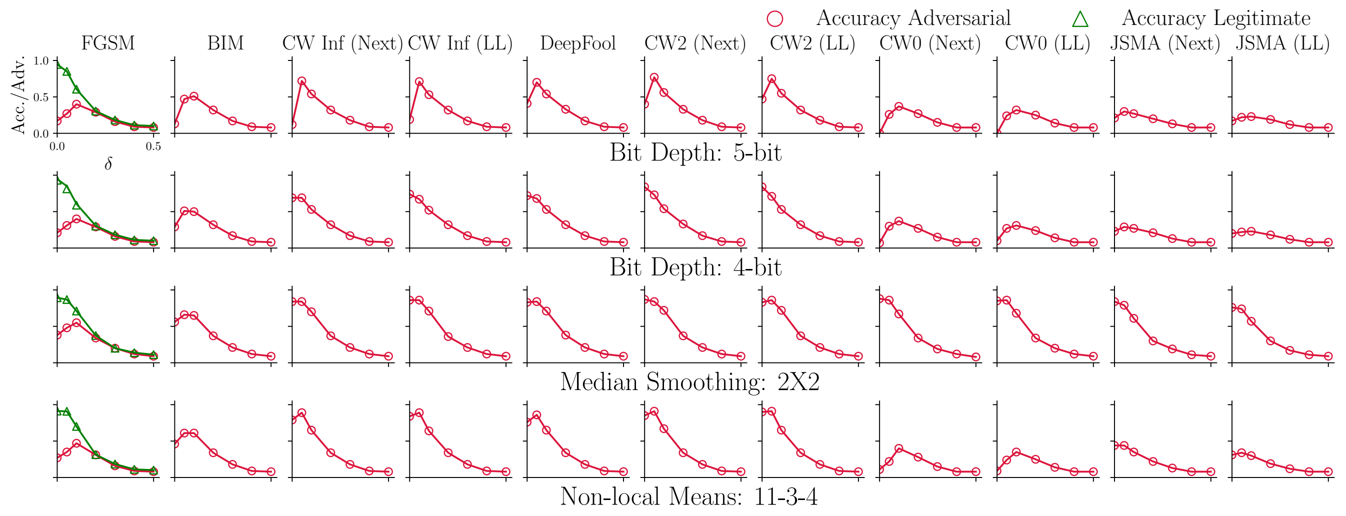

**Figure 4: CIFAR-10: Behavior of accuracy for magnitudes of randomness $\delta = [0, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5]$. We also plot the accuracy of the model for legitimate samples as $\delta$ increases (shown once for each defense as the curve does not change).**

the estimated probability (over the classifier's randomness) that the corresponding adversarial sample succeeds (note, this probability is binary for a deterministic classifier). However, when randomness is introduced the error probabilities spread out for a large number of samples as they are no longer deterministic. For each of the considered attacks $\mathcal{A}$, the empirical error over $X^{\mathcal{A}}$ can be computed by summing up the probabilities that each adversarial sample in $X^{\mathcal{A}}$ succeeds, i.e., $err_{X^{\mathcal{A}}}(\mathcal{C}_{\$}) = \sum_{i=1}^{100} \Pr\left[\mathcal{C}_{\$}(x_i) \neq f(x_i)\right]$. For ease of presentation, we only present the results applied to one defense in order to to demonstrate the effect of randomness, the results for other defenses are similar.

**Interpretation of Results.** Increase in the magnitude of randomness drives the accuracy of the classifier over legitimate

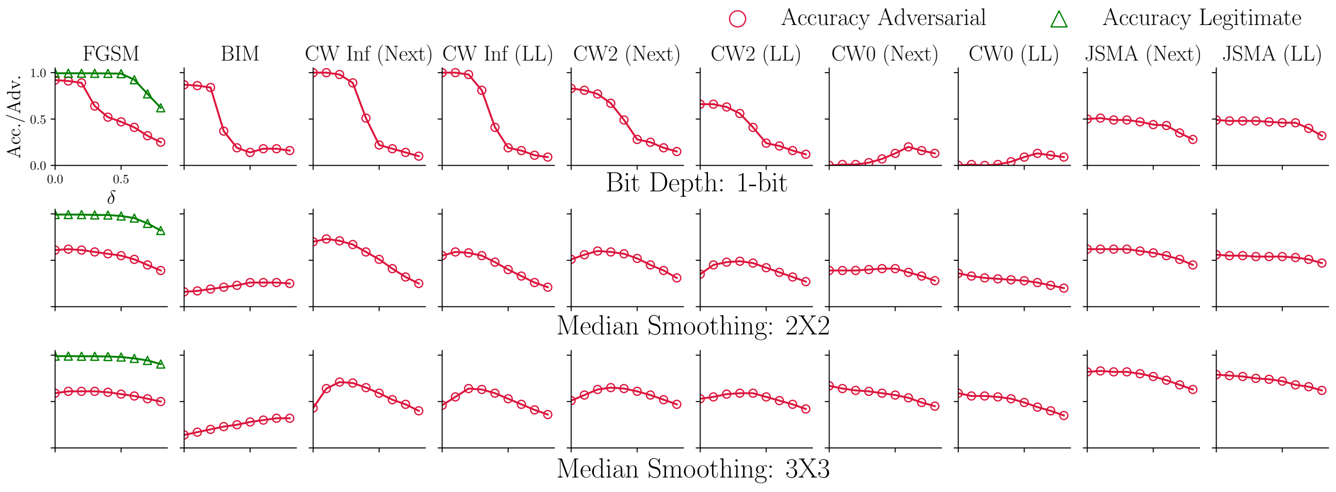

**Figure 5: MNIST: Behavior of accuracy for $\delta = [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8]$. We also plot the accuracy of the model for legitimate samples as $\delta$ increases (shown once for each defense as the curve does not change).**

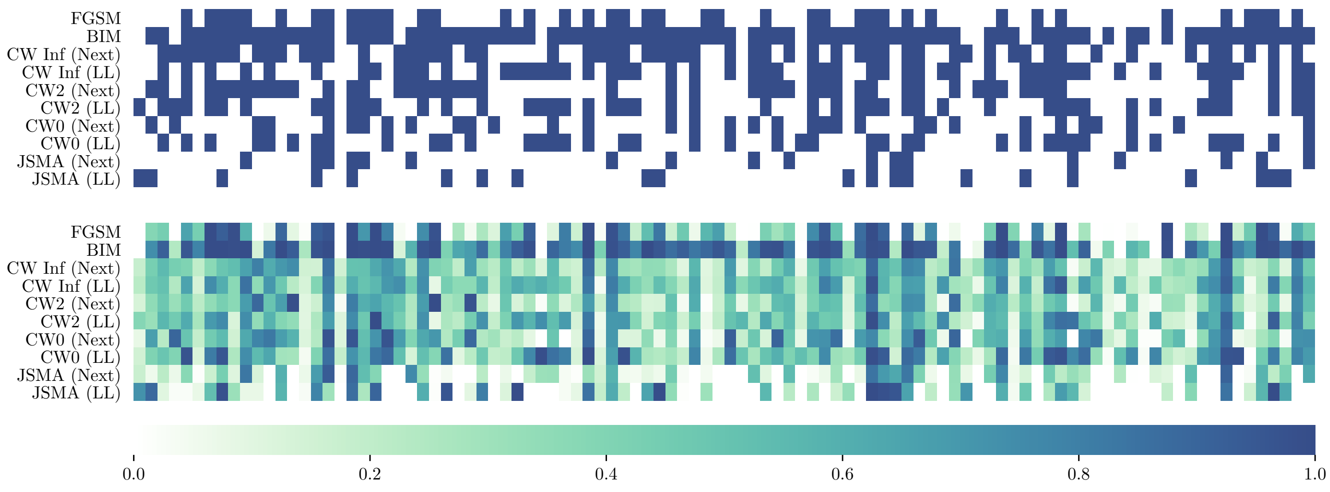

**Figure 6: MNIST: Unpredictability of errors for Median Smoothing ($3 \times 3$) defense, without randomness (top figure, $\delta = 0$) and with randomness (bottom figure, $\delta = 0.5$)**

samples towards 10% as classification becomes akin to guessing (for classification over 10 classes as in MNIST and CIFAR-10). We made a deliberate choice to clip the pixel values when they go outside the allowed bounds of $[0, 1]$ rather than wrapping around. A value of $\delta = 1$ and wrapping around the pixel values when they go out of bounds produces a truly random pixel, and hence the image. We found that at this level of randomness, accuracy over legitimate samples becomes close to 10%. Even lower values of $\delta$ produce a sharp drop in accuracy over legitimate samples, hence we choose to clip the values when they go out of bounds.

The primary motivation for our design of Randomized Squeezing is to perturb the pixels in a manner which subsumes the adversarial perturbation, and to which the adversary cannot adapt while keeping the usefulness of the classifier intact. The optimum magnitude of randomness $\delta$ to be used is contingent on the defense used. High values of $\delta$ have strong effect on the accuracy when used in conjunction with bit depth reduction, as it could change the value of a pixel drastically if the bit depth is low. In contrast, methods like local and non-local smoothing are much more resilient to high $\delta$, as they average out the noise from sections of images. The noise that we add, being additive, is filtered out.

Note that we want to use the largest value of $\delta$ possible so as to subsume the adversarial perturbations, while still maintaining high accuracy. Not all defenses are equally potent for all attacks and datasets, therefore the randomness magnitude $\delta$

**Table 1: Accuracy of original Feature Squeezing defenses without randomness ($\delta = 0$) over adversarial samples (%). Xu *et al.* [44] omit DeepFool on MNIST as the adversarial samples generated appear unrecognizable to humans; non-local smoothing is not applied to MNIST as it is hard to find similar patches on such images for smoothing a center patch. JSMA is omitted for ImageNet due to large memory requirement.**

| Dataset | Squeezer | | $L^\infty$ Attacks | | | | $L^2$ Attacks | | | $L^0$ Attacks | | | | All Attacks | Legitimate |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Name | Parameters | FGSM | BIM | $CW_\infty$ | | DeepFool | $CW_2$ | | $CW_0$ | | JSMA | | | |
| | | | | | Next | LL | | Next | LL | Next | LL | Next | LL | | |
| MNIST | None | | 54 | 9 | 0 | 0 | - | 0 | 0 | 0 | 0 | 27 | 40 | 13.00 | 99.43 |
| | Bit Depth | 1-bit | 92 | 87 | 100 | 100 | - | 83 | 66 | 0 | 0 | 50 | 49 | 62.70 | 99.33 |
| | Median Smoothing | $2 \times 2$ | 61 | 16 | 70 | 55 | - | 51 | 35 | 39 | 36 | 62 | 56 | 48.10 | 99.28 |
| | | $3 \times 3$ | 59 | 14 | 43 | 46 | - | 51 | 53 | 67 | 59 | 82 | 79 | 55.30 | 98.95 |
| CIFAR-10 | None | | 15 | 8 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 2.27 | 94.84 |
| | Bit Depth | 5-bit | 17 | 13 | 12 | 19 | 40 | 40 | 47 | 0 | 0 | 21 | 17 | 20.55 | 94.55 |
| | | 4-bit | 21 | 29 | 69 | 74 | 72 | 84 | 84 | 7 | 10 | 23 | 20 | 44.82 | 93.11 |
| | Median Smoothing | $2 \times 2$ | 38 | 56 | 84 | 86 | 83 | 87 | 83 | 88 | 85 | 84 | 76 | 77.27 | 89.29 |
| | Non-local Means | 11-3-4 | 27 | 46 | 80 | 84 | 76 | 84 | 88 | 11 | 11 | 44 | 32 | 53.00 | 91.18 |
| ImageNet | None | | 1 | 0 | 0 | 0 | 11 | 10 | 3 | 0 | 0 | - | - | 2.78 | 69.70 |
| | Bit Depth | 5-bit | 2 | 0 | 33 | 60 | 21 | 68 | 66 | 7 | 18 | - | - | 30.56 | 69.40 |
| | | 4-bit | 5 | 4 | 66 | 79 | 44 | 84 | 82 | 38 | 67 | - | - | 52.11 | 68.00 |
| | Median Smoothing | $2 \times 2$ | 22 | 28 | 75 | 81 | 72 | 81 | 84 | 85 | 85 | - | - | 68.11 | 65.40 |
| | | $3 \times 3$ | 33 | 41 | 73 | 76 | 66 | 77 | 79 | 81 | 79 | - | - | 67.22 | 62.10 |
| | Non-local Means | 11-3-4 | 10 | 25 | 77 | 82 | 57 | 87 | 86 | 43 | 47 | - | - | 57.11 | 65.40 |

**Table 2: Accuracy of Randomized Squeezing (%). Bold values indicate an improvement over corresponding values in Table 1.**

| Dataset | Squeezer | | $L^\infty$ Attacks | | | | $L^2$ Attacks | | | $L^0$ Attacks | | | | All Attacks | Legitimate |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Name | Parameters | FGSM | BIM | $CW_\infty$ | | DeepFool | $CW_2$ | | $CW_0$ | | JSMA | | | |
| | | | | | Next | LL | | Next | LL | Next | LL | Next | LL | | |
| MNIST | Bit Depth ($\delta = 0.2$) | 1-bit | 88.98 | 84.06 | 98.37 | 98.26 | - | 77.44 | 63.19 | **1.39** | **0.42** | 49.26 | 48.41 | 60.98 | 99.31 |
| | Median Smoothing ($\delta = 0.5$) | $2 \times 2$ | 54.78 | **25.53** | 50.70 | 40.22 | - | **52.20** | **42.21** | **40.65** | 28.27 | 58.15 | 54.05 | 44.68 | 97.54 |
| | | $3 \times 3$ | 58.03 | **27.61** | **58.53** | **52.87** | - | **61.27** | **55.40** | 56.86 | 48.78 | 77.23 | 71.67 | **56.83** | 97.99 |
| CIFAR-10 | Bit Depth ($\delta = 0.05$) | 5-bit | **27.33** | **47.27** | **71.75** | **70.89** | **69.91** | **76.90** | **75.06** | **26.21** | **23.89** | **29.84** | **22.06** | **49.19** | 85.03 |
| | | 4-bit | **30.70** | **51.34** | 68.89 | 67.29 | 67.55 | 72.95 | 71.08 | **30.25** | **26.60** | **29.36** | **22.14** | **48.92** | 81.27 |
| | Median Smoothing ($\delta = 0.05$) | $2 \times 2$ | **47.66** | **66.23** | 83.59 | 85.90 | **83.70** | 84.45 | **86.50** | 85.61 | **86.11** | 78.54 | 73.62 | **78.36** | 86.80 |
| | Non-local Means ($\delta = 0.05$) | 11-3-4 | **35.27** | **61.19** | **88.59** | **89.39** | **86.31** | **91.01** | **90.52** | **22.08** | **24.21** | **44.05** | **33.80** | **60.58** | 90.90 |
| ImageNet | Bit Depth ($\delta = 0.1$) | 5-bit | **34.86** | **54.97** | **76.97** | **79.50** | **70.73** | **79.45** | **80.31** | **67.52** | **67.43** | - | - | **67.97** | 63.00 |
| | | 4-bit | **35.63** | **56.11** | **76.66** | 78.92 | **70.91** | 79.11 | 79.69 | **67.40** | 66.48 | - | - | **67.88** | 61.00 |
| | Median Smoothing ($\delta = 0.1$) | $2 \times 2$ | **53.44** | **62.49** | 69.11 | 69.09 | 67.22 | 70.13 | 69.97 | 68.47 | 68.20 | - | - | 66.46 | 57.00 |
| | | $3 \times 3$ | **52.23** | **58.88** | 63.76 | 63.88 | 62.87 | 64.46 | 63.91 | 63.76 | 64.22 | - | - | 62.00 | 52.50 |
| | Non-local Means ($\delta = 0.1$) | 11-3-4 | **32.53** | **55.29** | **78.97** | 81.28 | **72.92** | 81.11 | 81.52 | **66.02** | **68.71** | - | - | **68.71** | 64.50 |
| | Pure Randomness | $\delta = 0.1$ | 34.70 | 56.33 | 76.91 | 79.58 | 68.84 | 79.05 | 80.41 | 67.06 | 67.5 | - | - | 67.82 | 64 |

must be carefully chosen. Randomized Squeezing can mitigate the limitation of a weak defense to some extent, as seen in the case of CIFAR-10 bit depth (5-bit) defense where the accuracy over adversarial samples increases by almost 2.5 times. However, efficacy of the defense is critical to have success in general.

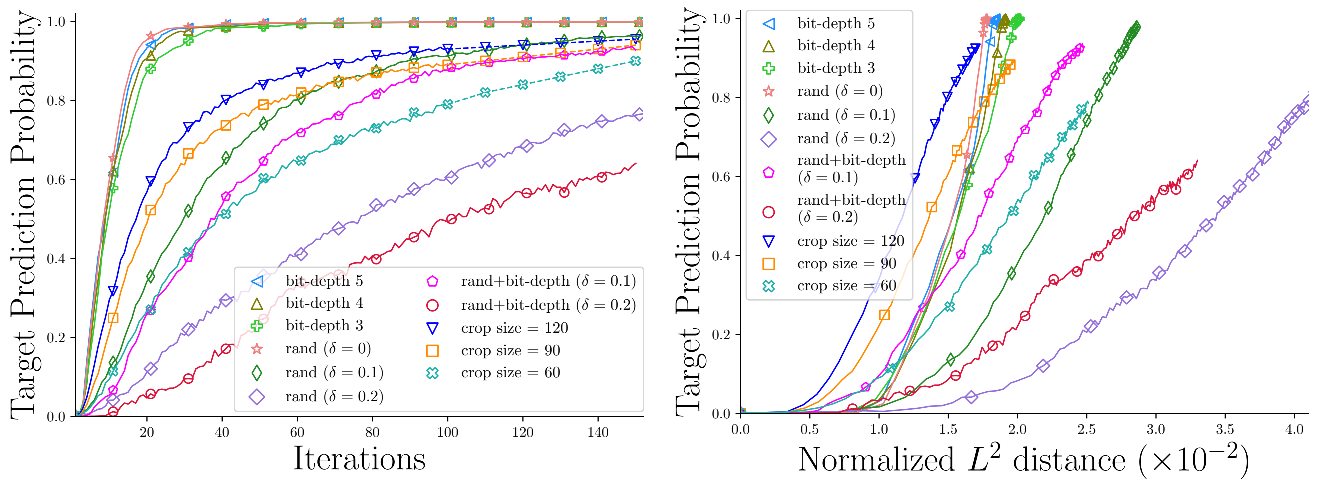

**Figure 7: Increase in target prediction probability and distortion (in normalized $L^2$ distance) as more confident adversarial samples are synthesized (dashed lines show extrapolated values).**

## 5.3 Whitebox Adversaries

In this section, we study whether randomness influences the success of whitebox adversaries. We evaluate the BPDA and EOT attacks proposed by Athalye *et al.* [1] against Cropping-Rescaling [16], Region-Based Classification [5],[8] and Randomized Squeezing with bit-depth reduction as squeezing function. Beyond tuning the attacks to each defensive technique, we chose the attack strategies that are best suited for each of the defenses, as outlined in [1]. Specifically, we consider: a pure instantiation of EOT against Cropping-Rescaling, as this defense applies differentiable transformations; the BPDA attack against Region-Based Classification, and a combination of BPDA and EOT against Randomized Squeezing, so that both the non-differentiability of squeezing and the input randomization are taken into account when generating adversarial samples. For completeness, we also ran the attacks against the deterministic bit-depth reduction.

We evaluated the defenses against the aforementioned attacks on the same set of 100 images, selected at random from the ImageNet dataset. Each image was assigned a target class at random. Figure 7 summarizes the results.

Recall that every attack iteration aims to generate adversarial samples with higher confidence compared to the previous iteration. Correspondingly, in the leftmost plot we depict the (average) target prediction probability against the number of iterations. We see that the prediction probability starts at 0 for all defenses, and approaches 1 as the number of iterations increases. Notice that the prediction probability quickly reaches 1 for defenses which have no randomness whereas it grows gradually for randomness-based defenses. For instance, when randomness is not applied, less than 20 iterations are sufficient to achieve a prediction probability of 0.8. In contrast, for randomized defenses 20 iterations lead to a prediction probability of only 0.6 in the best case (i.e., Cropping-Rescaling with crop size 120), and of less than 0.05 in the case of the seemingly most robust technique (i.e., Randomized Squeezing for $\delta = 0.2$). Comparing among the randomness-based defenses, we observe that for a prediction probability of 0.6, the attacks requires about 20, 35, or 50 iterations for Cropping-Rescaling depending on the crop size, 35 or 100 iterations in the case of Region-Based Classification for randomness magnitudes $\delta = 0.1$ and $\delta = 0.2$, respectively, and 45 or 140 iterations against Randomized Squeezing for the same values of $\delta$, indicating that both online defenses outperform Cropping-Rescaling.

We also illustrate how the target prediction probability varies with the distortion, measured as normalized $L^2$ distance (see the rightmost plot in Figure 7). Again, we see that randomness-based defenses are more robust than their deterministic counterparts, as they force the attacker to introduce larger perturbation. Namely, for all deterministic defenses, high-confidence adversarial samples can be generated with a distortion below 0.02, while larger perturbations are necessary to defeat randomized defenses. In particular, high-confidence adversarial samples against Region-Based Classification and Randomized Squeezing require perturbations with $L^2$ distance above 0.025 (for $\delta = 0.1$), and above 0.035 (for $\delta = 0.2$), respectively, with Region-Based Classification presenting slightly higher robustness than Randomized Squeezing according to this metric. Our results support the intuition that introducing unpredictability to the classification process makes it computationally expensive to find adversarial perturbations which are successful regardless of the randomness.

[8]Our implementation of Region-Based Classification slightly differs from the originally proposed version [5], namely we use the same randomization strategy as Randomized Squeezing (which is equivalent to sampling at random from a hypercube, cf. Section 4.3) and classify one sample instead of ensembling. Due to averaging over 100 images, ensembling would lead to the same results as adversary can chose to defeat a majority of samples.

## 6 RELATED WORK

There has been massive effort to make ML models robust to adversarial samples, leading to a huge number of defenses proposed in the last few years.[9] Here, we discuss only a selection of these proposals which we find representative of general defensive principles. Following widely-adopted nomenclature, we differentiate between *certified* and *heuristic* defenses to distinguish between proposals that come with provable guarantees and those which do not.

**Heuristic defenses.** An early proposal is *defensive distillation* [30, 34], which extends the deep-learning concept of distillation to the adversarial setting, aiming to extract knowledge from a given neural-network architecture to improve its own resilience to gradient-based attacks. This proposal has been shown ineffective [31].

A broad class of defenses attempts to *detect* whether a given input is adversarial. Early methods derive statistical properties from large datasets of legitimate, respectively, adversarial samples, and then inspect these properties to discern the adversarial nature of new and unknown samples [11, 14]. Even though these techniques where shown to be quite robust, they are computationally expensive and require large datasets for the reliability of statistical results. A more efficient approach is to train a detector to specifically learn adversarial samples, as done by MagNet [28]. MagNet relies on the assumption that natural data lie on a manifold of significantly smaller dimension than the whole input space, while adversarial samples fall outside the manifold. Based on this, Magnet employs a detector which deems samples far from the manifold as adversarial. A different kind of detector is instantiated by the feature squeezing defense by Xu *et al.* [44] (cf. Section 2.2), which uses the discrepancy of output predictions between original samples and their squeezed versions to detect adversarial manipulations.

We note that Randomized Squeezing, while using squeezing routines, does not aim at detecting adversarial samples, rather at making it harder for the adversary to generate successful perturbations. Instead, our proposal can be seen as an instantiation of so-called *input transformation*, which applies a pre-processing step to the input in order to reduce the sensitivity of the model to small changes in input—with the hope of making the classifier more robust to adversarial perturbations [16].

The most robust among all heuristic defenses to date appears to be *adversarial training*, introduced in [13] and later extended in several works [23, 26]. Adversarial training essentially finds adversarial samples by running known attacks, and adds those samples to the training set so that the model learns to correctly classify certain adversarial inputs. Madry *et al.* [26] propose a generic training methodology targeting robustness against all low-distortion adversarial samples, i.e., samples generated by applying small perturbations to clean inputs. This methodology is based on the idea that, if the training set contains sufficiently representative adversarial samples, the resulting classifier will be able to withstand *all* low-distortion attacks. Based on this, the authors heuristically generate "sufficiently representative" adversarial samples using projected gradient descent (PDG), an attack strategy which generalizes first-order attacks. The resulting trained networks, based on MNIST and CIFAR datasets respectively, achieve different levels of robustness against FGSM, PGD, and CW attacks. Although various works demonstrated the feasibility of this technique, adversarial training requires a large number of samples that are expensive to generate. In addition, it cannot resist unknown attacks.

Except for adversarial training, all the aforementioned techniques were shown, in a way or another, vulnerable to adaptive attacks.

**Certified defenses.** The idea of ensuring robustness to *all* attacks within a certain class has been investigated further, fostering a line of work aimed at developing *certified defenses* [10, 19, 35, 36, 42]. In contrast to heuristic defenses, certified defenses provide provable guarantees against bounded adversarial perturbations. More specifically, given a classifier and an input sample, a certified defense comes with an upper bound on the worst-case loss against norm-based attackers: the bound provides a "certificate of robustness", and it guarantees that no perturbation within the allowed threshold can turn the starting input into an adversarial one, therefore ruling out all attacks which are restricted to the given threshold. Certified defenses offer a promising direction towards ending the arms race between attackers and defenders.

To improve early certification techniques, which do not scale to large dataset such as ImageNet, *PixelDP* [25] leverages differential privacy to generically increase the robustness of a based classifier, offering probabilistic certificates of robustness for several datasets, including ImageNet. This solution trades scalability with clean-data accuracy, which drops significantly as the allowed adversarial perturbation increases. In a similar vein, recent works prove the certified robustness of *randomized smoothing*, a pre-processing technique similar to Randomized Squeezing and Region-Based Classification which adds Gaussian noise to the classifier's input (instead of uniform noise) [9, 22].

On the downside, robustness certificates only hold with respect to the original input, meaning that only test inputs can be certified [6]. It is thus unclear which guarantees a robustness certificate can offer for data that has not been seen before. In addition, recent work by Ghiasi *et al.* [12] shows an attack, so-called *shadow*, to bypass certified defenses, hinting that certified robustness does not truly capture robustness to all realistic attacks. More specifically, the shadow attack generates adversarial samples which do not respect the norm-bound imposed by the certificate, and thus are technically outside the adversarial model; however, those samples are indistinguishable from the original samples, and cause the classifier to generate a "spoofed" certificate of robustness, ultimately bypassing the defense.

[9]https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html

## 7 CONCLUDING REMARKS

In this work, we investigated whether *randomness* can help in increasing the robustness of DL classifiers against adversarial samples. We thoroughly analyzed three heuristic defensive techniques that employ randomness in different ways, namely Cropping-Rescaling [16], Region-Based Classification [5], and Randomized Squeezing (which we propose), and compared their effectiveness against state of the art graybox and whitebox attack strategies.

Our results show that randomness can enhance robustness against graybox attacks. This is in line with prior work investigating, among others, randomized input transformations as a defense against (blackbox and) graybox attacks [16]. In contrast to prior findings [16], our results demonstrate that randomized transformations can be effective even when applied in a purely online fashion, i.e., without the need of augmenting training with the same transformations.

None of the analyzed defenses can deter whitebox attacks, though. In fact, designing secure solutions against fully-adaptive adversaries is an extremely challenging open problem [4, 27, 37, 38]. Although recent works investigate certified defenses [10, 19, 35, 36, 42], which provide provable guarantees against bounded adversarial perturbations, these solutions are limited by the prior knowledge of the inputs, i.e., only inputs that are included within the test set can be certified, and were recently bypassed by out-of-the-model, yet realistic, attacks.

A close look at our experimental results reveals that, although the strongest currently known whitebox attacks can successfully generate robust adversarial samples against the three defenses, increasing the amount of added randomness requires a higher number of iterations for the attack to succeed or, similarly, a larger distortion to achieve a certain success rate. That is, larger randomness makes the task of generating adversarial samples "moderately harder". This may be good enough for applications in which the attacker has limited time to generate an adversarial sample, or is restricted to small distortion.

An interesting future direction would be to further explore the effect of randomization on the attack's cost, in terms of lower bounds for the number of iterations, respectively, amount of perturbation needed to generate robust adversarial samples. We therefore hope that our paper motivates further research in this area.

## ACKNOWLEDGMENTS

The authors are thankful to Anish Athalye, Nicholas Carlini, and anonymous reviewers for helpful comments. This paper has (partly) received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 779852.

## REFERENCES

[1] Anish Athalye, Nicholas Carlini, and David A. Wagner. 2018. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples. In *ICML (JMLR Workshop and Conference Proceedings)*, Vol. 80. JMLR.org, 274–283.

[2] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. 2018. Synthesizing Robust Adversarial Examples. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018 (JMLR Workshop and Conference Proceedings)*, Jennifer G. Dy and Andreas Krause (Eds.), Vol. 80. JMLR.org, 284–293. http://proceedings.mlr.press/v80/athalye18b.html

[3] Michael Backes and Mohammad Nauman. 2017. LUNA: Quantifying and Leveraging Uncertainty in Android Malware Analysis through Bayesian Machine Learning. In *EuroS&P*. IEEE, 204–217.

[4] Sébastien Bubeck, Eric Price, and Ilya P. Razenshteyn. 2018. Adversarial examples from computational constraints. *CoRR* abs/1805.10204 (2018).

[5] Xiaoyu Cao and Neil Zhenqiang Gong. 2017. Mitigating Evasion Attacks to Deep Neural Networks via Region-based Classification. In *Proceedings of the 33rd Annual Computer Security Applications Conference, Orlando, FL, USA, December 4-8, 2017*. ACM, 278–287. https://doi.org/10.1145/3134600.3134606

[6] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian J. Goodfellow, Aleksander Madry, and Alexey Kurakin. 2019. On Evaluating Adversarial Robustness. *CoRR* abs/1902.06705 (2019).

[7] Nicholas Carlini and David A. Wagner. 2017. Towards Evaluating the Robustness of Neural Networks. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*. 39–57. https://doi.org/10.1109/SP.2017.49

[8] Jiefeng Chen, Xi Wu, Yingyu Liang, and Somesh Jha. 2018. Improving Adversarial Robustness by Data-Specific Discretization. *CoRR* abs/1805.07816 (2018). arXiv:1805.07816 http://arxiv.org/abs/1805.07816

[9] Jeremy M. Cohen, Elan Rosenfeld, and J. Zico Kolter. 2019. Certified Adversarial Robustness via Randomized Smoothing. In *ICML (Proceedings of Machine Learning Research)*, Vol. 97. PMLR, 1310–1320.

[10] Krishnamurthy Dvijotham, Sven Gowal, Robert Stanforth, Relja Arandjelovic, Brendan O'Donoghue, Jonathan Uesato, and Pushmeet Kohli. 2018. Training verified learners with learned verifiers. *CoRR* abs/1805.10265 (2018).

[11] R. Feinman, R. R. Curtin, S. Shintre, and A. B. Gardner. 2017. Detecting Adversarial Samples from Artifacts. *ArXiv e-prints* (March 2017). arXiv:1703.00410 [stat.ML]

[12] Amin Ghiasi, Ali Shafahi, and Tom Goldstein. 2020. Breaking Certified Defenses: Semantic Adversarial Examples with Spoofed Robustness Certificates. In *International Conference on Learning Representations*. https://openreview.net/forum?id=HJxdTxHYvB

[13] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. In *International Conference on Learning Representations*. http://arxiv.org/abs/1412.6572

[14] Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick D. McDaniel. 2017. On the (Statistical) Detection of Adversarial Examples. *CoRR* abs/1702.06280 (2017). http://arxiv.org/abs/1702.06280

[15] Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick D. McDaniel. 2016. Adversarial Perturbations Against Deep Neural Networks for Malware Classification. *CoRR* abs/1606.04435 (2016). http://arxiv.org/abs/1606.04435

[16] Chuan Guo, Mayank Rana, Moustapha Cissé, and Laurens van der Maaten. 2018. Countering Adversarial Images using Input Transformations. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=SyJ7ClWCb

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *CVPR*. IEEE Computer Society, 770–778.

[18] Warren He, James Wei, Xinyun Chen, Nicholas Carlini, and Dawn Song. 2017. Adversarial Example Defense: Ensembles of Weak Defenses are not Strong. In *11th USENIX Workshop on Offensive Technologies, WOOT 2017, Vancouver, BC, Canada, August 14-15, 2017.*, William Enck and Collin Mulliner (Eds.). USENIX Association. https://www.usenix.org/conference/woot17/workshop-program/presentation/he

[19] Matthias Hein and Maksym Andriushchenko. 2017. Formal Guarantees on the Robustness of a Classifier against Adversarial Manipulation. In *NIPS*. 2263–2273.

[20] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *CoRR* abs/1704.04861 (2017). arXiv:1704.04861 http://arxiv.org/abs/1704.04861

[21] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2017. Densely Connected Convolutional Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. 2261–2269. https://doi.org/10.1109/CVPR.2017.

243
[22] Jinyuan Jia, Xiaoyu Cao, Binghui Wang, and Neil Zhenqiang Gong. 2019. Certified Robustness for Top-k Predictions against Adversarial Perturbations via Randomized Smoothing. *CoRR* abs/1912.09899 (2019).
[23] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2016. Adversarial Machine Learning at Scale. arXiv:arXiv:1611.01236
[24] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. 2016. Adversarial examples in the physical world. *CoRR* abs/1607.02533 (2016). arXiv:1607.02533 http://arxiv.org/abs/1607.02533
[25] Mathias Lécuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. 2018. On the Connection between Differential Privacy and Adversarial Robustness in Machine Learning. *CoRR* abs/1802.03471 (2018). arXiv:1802.03471 http://arxiv.org/abs/1802.03471
[26] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *ICLR (Poster)*. OpenReview.net.
[27] Saeed Mahloujifar and Mohammad Mahmoody. 2018. Can Adversarially Robust Learning Leverage Computational Hardness? *CoRR* abs/1810.01407 (2018).
[28] Dongyu Meng and Hao Chen. 2017. MagNet: a Two-Pronged Defense against Adversarial Examples. *CoRR* abs/1705.09064 (2017). http://arxiv.org/abs/1705.09064
[29] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 2574–2582. https://doi.org/10.1109/CVPR.2016.282
[30] Nicolas Papernot and Patrick D. McDaniel. 2017. Extending Defensive Distillation. *CoRR* abs/1705.05264 (2017). http://arxiv.org/abs/1705.05264
[31] Nicolas Papernot and Patrick D. McDaniel. 2017. Extending Defensive Distillation. *CoRR* abs/1705.05264 (2017).
[32] Nicolas Papernot, Patrick D. McDaniel, and Ian J. Goodfellow. 2016. Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples. *CoRR* abs/1605.07277 (2016). http://arxiv.org/abs/1605.07277
[33] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. 2016. The Limitations of Deep Learning in Adversarial Settings. In *IEEE European Symposium on Security and Privacy, EuroS&P 2016, Saarbrücken, Germany, March 21-24, 2016*. 372–387. https://doi.org/10.1109/EuroSP.2016.36
[34] Nicolas Papernot, Patrick D. McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2016. Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks. In *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 582–597.
[35] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. 2018. Certified Defenses against Adversarial Examples. In *International Conference on Learning Representations*. https://openreview.net/forum?id=Bys4ob-Rb
[36] Aditi Raghunathan, Jacob Steinhardt, and Percy S. Liang. 2018. Semidefinite relaxations for certifying robustness to adversarial examples. In *NeurIPS*. 10900–10910.
[37] Ali Shafahi, W. Ronny Huang, Christoph Studer, Soheil Feizi, and Tom Goldstein. 2018. Are adversarial examples inevitable? *CoRR* abs/1809.02104 (2018).
[38] Adi Shamir, Itay Safran, Eyal Ronen, and Orr Dunkelman. 2019. A Simple Explanation for the Existence of Adversarial Examples with Small Hamming Distance. *CoRR* abs/1901.10861 (2019). arXiv:1901.10861 http://arxiv.org/abs/1901.10861
[39] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *International Conference on Learning Representations*. http://arxiv.org/abs/1312.6199
[40] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. 2014. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. In *CVPR*. IEEE Computer Society, 1701–1708.
[41] Leslie G. Valiant. 1984. A Theory of the Learnable. *Commun. ACM* 27, 11 (1984), 1134–1142.
[42] Eric Wong and J. Zico Kolter. 2018. Provable Defenses against Adversarial Examples via the Convex Outer Adversarial Polytope. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018 (JMLR Workshop and Conference Proceedings)*, Jennifer G. Dy and Andreas Krause (Eds.), Vol. 80. JMLR.org, 5283–5292. http://proceedings.mlr.press/v80/wong18a.html
[43] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. 2018. Mitigating Adversarial Effects Through Randomization. In *International Conference on Learning Representations*. https://openreview.net/forum?id=Sk9yuql0Z
[44] Weilin Xu, David Evans, and Yanjun Qi. 2017. Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks. *CoRR* abs/1704.01155 (2017).
[45] Zhenlong Yuan, Yongqiang Lu, Zhaoguo Wang, and Yibo Xue. 2014. Droid-Sec: deep learning in android malware detection. In *SIGCOMM*. ACM, 371–372.
[46] Yan Zhou, Murat Kantarcioglu, and Bowei Xi. 2018. Breaking Transferability of Adversarial Samples with Randomness. *CoRR* abs/1805.04613 (2018). arXiv:1805.04613 http://arxiv.org/abs/1805.04613