MEAD: A Multi-Armed Approach for Evaluation of Adversarial Examples Detectors

Federica Granese (✉)[1⋆], Marine Picot[2,3⋆], Marco Romanelli[2], Francesco Messina[5], and Pablo Piantanida[4]

[1] Lix, Inria, Institute Polytechnique de Paris, Sapienza University of Rome
federica.granese@inria.fr
[2] Laboratoire des signaux et systèmes (L2S),
Université Paris-Saclay, CNRS, CentraleSupélec, France
{marine.picot,marco.romanelli,pablo.piantanida}@centralesupelec.fr
[3] McGill University, Canada
[4] International Laboratory on Learning Systems (ILLS),
McGill - ETS - MILA - CNRS - Université Paris-Saclay - CentraleSupélec, Canada
[5] Universidad de Buenos Aires, Argentina
fmessina@fi.uba.ar

Abstract. Detection of adversarial examples has been a hot topic in the last years due to its importance for safely deploying machine learning algorithms in critical applications. However, the detection methods are generally validated by assuming a single implicitly known attack strategy, which does not necessarily account for real-life threats. Indeed, this can lead to an overoptimistic assessment of the detectors' performance and may induce some bias in the comparison between competing detection schemes. We propose a novel multi-armed framework, called MEAD, for evaluating detectors based on several attack strategies to overcome this limitation. Among them, we make use of three new objectives to generate attacks. The proposed performance metric is based on the worst-case scenario: detection is successful if and only if all different attacks are correctly recognized. Empirically, we show the effectiveness of our approach. Moreover, the poor performance obtained for state-of-the-art detectors opens a new exciting line of research.

Keywords: Adversarial Examples · Detection · Security.

1 Introduction

Despite recent advances in the application of machine learning, the vulnerability of deep learning models to maliciously crafted examples [34] is still an open problem of great interest for safety-critical applications [2,4,10,39]. Over time, a large body of literature has been produced on the topic of defense methods against adversarial examples. On the one hand, interest in detecting adversarial examples given a pre-trained model is gaining momentum [17,24,27]. On the other hand,

⋆ These authors contributed equally to this work.

several techniques have been proposed to train models with improved robustness to future attacks [26,31,38]. Interestingly, Croce et al. have recently pointed out that, due to the large number of proposed methods, the problem of crafting an objective approach to evaluate the quality of methods to train robust models is not trivial. To this end, they have presented *RobustBench* [9], a standardized benchmark to assess adversarial robustness. To the best of our knowledge, we claim that an equivalent benchmark does not exist in the case of methods to detect adversarial examples given a pre-trained model. Therefore, in this work, we provide a general framework to evaluate the performance of adversarial detection methods. Our idea stems from the following key observation. Generally, the performance of current state-of-the-art (SOTA) adversarial examples detection methods is evaluated assuming a unique and thus implicitly known attack strategy, which does not necessarily correspond to real-life threats. We further argue that this type of evaluation has two main flaws: the performance of detection methods may be overestimated, and the comparison between detection schemes may be biased. We propose a two-fold solution to overcome the aforementioned limitations, leading to a less biased evaluation of different approaches. This is accomplished by evaluating the detection methods on simultaneous attacks on the target classification model using different adversarial strategies, considering the most popular attack techniques in the literature, and incorporating three new attack objectives to extend the generality of the proposed framework. Indeed, we argue that additional attack objectives result in new types of adversarial examples that cannot be constructed otherwise. In particular, we translate such an evaluation scheme in MEAD.

MEAD is a novel evaluation framework that uses a simple but still effective "multi-armed" attack to remove the implicit assumption that detectors know the attacker's strategy. More specifically, for each natural sample, we consider the detection to be successful *if and only if* the detector is able to identify all the different attacks perpetrated by perturbing the testing sample at hand. We deploy the proposed framework to evaluate the performance of SOTA adversarial examples detection methods over multiple benchmarks of visual datasets. Overall, the collected results are consistent throughout the experiments. The main takeaway is that considering a multi-armed evaluation criterion exposes the weakness of SOTA detection methods, yielding, in some cases, relatively poor performances. The proposed framework, although not exhaustive, sheds light on the fact that evaluations so far presented in the literature are highly biased and unrealistic. Indeed, the same detector achieves very different performances when it is informed about the current attack as opposed to when it is not. Not surprisingly, supervised and unsupervised methods achieve comparable performances with the multi-armed framework, meaning that training the detectors knowing a specific attack used at testing time does not generalize to other attacks enough. Indeed the goal of MEAD is not to show that new attacks can always fool robust classifiers but to show that the detectors that may work well when evaluated with a unique attack strategy end up being defeated by new attacks.

1.1 Summary of contributions

We propose MEAD, a novel multi-armed evaluation framework for adversarial examples detectors involving several attackers to ensure that the detector is not overfitted to a particular attack strategy. The proposed metric is based on the following criterion. Each adversarial sample is correctly detected if and only if all the possible attacks on it are successfully detected. We show that this approach is less biased and yields a more effective metric than the one obtained by assuming only a single attack at evaluation time (see Sec. 4).

We make use of three new objective functions which, to the best of our knowledge, have never been used for the purpose of generating adversarial examples at testing time. These are *KL divergence*, *Gini Impurity* and *Fisher-Rao distance*. Moreover, we argue that each of them contributes to jointly creating competitive attacks that cannot be created by a single function (see Sec. 3.2).

We perform an extensive numerical evaluation of SOTA and uncover their limitations, suggesting new research perspectives in this research line (see Sec. 5).

The remaining paper is organized as follows. First, in Sec. 2, we present a detailed overview of the recent related works. In Sec. 3, we describe the adversarial problem along with the new objectives we introduce within the proposed evaluation framework, MEAD, which is further explained in Sec. 4. We extensively experimentally validate MEAD in Sec. 5. Finally, in Sec. 6, we provide the summary together with concluding remarks.

2 Related Works

State-of-the-art methods to detect adversarial examples can be separated in two main groups [2]: supervised and unsupervised methods. In the supervised setting, detectors can make use of the knowledge of the attacker's procedure. The *network invariant model approach* extracts natural and adversarial features from the activation values of the network's layers [7,23,28], while the *statistical approach* extract features using statistical tools (e.g. maximum mean discrepancy [16], PCA [21], kernel density estimation [13], local intrinsic dimensionality [25], model uncertainty [13] or natural scene statistics [17]) to separate in-training and out-of-training data distribution/manifolds. To overcome the intrinsic limitation of the necessity to have prior knowledge of attacks, unsupervised detection methods consider only clean data at training time. The features extraction can rely on different techniques (e.g., *feature squeezing* [22,36], *denoiser approach* [27], *network invariant* [24], *auxiliary model* [1,32,40]). Moreover, detection methods of adversarial examples can also act on the underlying classifier by considering *a novel training procedure* (e.g., reverse cross-entropy [30]; the rejection option [1,32]) and a thresholding test strategy towards robust detection of adversarial examples. Finally, detection methods can also be impacted by the learning task of the underlying network (e.g., for human recognition tasks [35]).

2.1 Considered detection methods

Supervised methods. Supervised methods can make use of the knowledge of how adversarial examples are crafted. They often use statistical properties of either the input samples or the output of hidden layers. *NSS* [17] extract the *Natural Scene Statistics* of the natural and adversarial examples, while *LID* [25] extract the *local intrinsic dimensionality* features of the output of hidden layers for natural, noisy and adversarial inputs. *KD-BU* [13] estimates the *kernel density* of the last hidden layer in the feature space, then estimates the *bayesian uncertainty* of the input sample, following the intuition that the adversarial examples lie off the data manifold. Once those features are extracted, all methods train a detector to discriminate between natural and adversarial samples.

Unsupervised methods. Unsupervised method can only rely on features of the natural samples. *FS* [36] is an unsupervised method that uses *feature squeezing* to compare the model's predictions. Following the idea of estimating the distance between the test examples and the boundary of the manifold of normal examples, *MagNet* [27] comprises detectors based on *reconstruction error* and detectors based on *probability divergence*.

2.2 Considered attack mechanisms

The attack mechanisms can be divided into two categories: whitebox attacks, where the adversary has complete knowledge about the targeted classifier (its architecture and weights), and blackbox attacks where the adversary has no access to the internals of the target classifier.

Whitebox attacks. One of the first introduced attack mechanisms is what we call the Fast Gradient Sign Method (**FGSM**) [14]. It relies on computing the direction gradient of a given objective function with respect to (w.r.t.) the input of the targeted classifier and modifying the original sample following it. This method has been improved multiple times. Basic Iterative Method (**BIM**) [19] and Projected Gradient Descent (**PGD**) [26] are two iteration extensions of **FGSM**. They were introduced at the same time, and the main difference between the two is that **BIM** initializes the algorithm to the original sample while **PGD** initializes it to the original sample plus a random noise. Despite that **PGD** was introduced under the L_∞-norm constraint, it can be extended to any L_p-norm constraint. Deepfool (**DF**) [29] was later introduced. It is an iterative method based on a local linearization of the targeted classifier and the resolution of this simplified problem. Finally, the Carlini&Wagner method (**CW**) aims at finding the smaller noise to solve the adversarial problem. To do so, they present a relaxation based on the minimization of specific objectives that can be chosen depending on the attacker's goal.

Blackbox attacks. Blackbox attacks can only rely on queries to attack specific models. Square Attack (**SA**) [3] is an iterative method that randomly searches for a perturbation that will increase the attacker's objective at each step, Hop Skip Jump (**HOP**) [8] tries to estimate the gradient direction to perturb, and Spatial Transformation Attack (**STA**) [12] applies small translations and rotations to the original sample to fool the targeted classifier.

3 Adversarial Examples and Novel Objectives

Let $\mathcal{X} \subseteq \mathbb{R}^d$ be the input space and let $\mathcal{Y} = \{1, \dots, C\}$ be the label space related to some task of interest. We denote by P_{XY} the unknown data distribution over $\mathcal{X} \times \mathcal{Y}$. Throughout the paper we refer to the classifier $q_{\widehat{Y}|X}(y|\mathbf{x};\theta)$ to be the parametric soft-probability model, where $\theta \in \Theta$ are the parameters, $y \in \mathcal{Y}$ the label and $f_\theta : \mathcal{X} \to \mathcal{Y}$ s.t. $f_\theta(\mathbf{x}) = \arg\max_{y \in \mathcal{Y}} q_{\widehat{Y}|X}(y|\mathbf{x};\theta)$ to be its induced hard decision. Finally, we denote by $\mathbf{x}' \in \mathbb{R}^d$ an adversarial example, by $\ell(\mathbf{x}, \mathbf{x}';\theta)$ the objective function used by the attacker to generate that sample, and $a_\ell(\cdot;\varepsilon,p)$ the attack mechanism according to a objective function ℓ, with ε the maximal perturbation allow and p the L_p-norm constraint.

3.1 Generating adversarial examples

Adversarial examples are slightly modified inputs that can fool a target classifier. Concretely, Szegedy *et al.* [33] define the adversarial generation problem as:

$$\mathbf{x}' = \underset{\mathbf{x}' \in \mathbb{R}^d \,:\, \|\mathbf{x}'-\mathbf{x}\|_p < \varepsilon}{\arg\min} \|\mathbf{x}' - \mathbf{x}\| \quad \text{s.t.} \quad f_\theta(\mathbf{x}') \neq y, \tag{1}$$

where y is the true label (supervision) associated to the sample $\mathbf{x}$. Since this problem is difficult to tackle, it is commonly relaxed as follows [6]:

$$a_\ell(\mathbf{x};\varepsilon,p)^{1} \equiv \mathbf{x}'_\ell = \underset{\mathbf{x}'_\ell \in \mathbb{R}^d \,:\, \|\mathbf{x}'_\ell-\mathbf{x}\|_p < \varepsilon}{\arg\max} \ell(\mathbf{x}, \mathbf{x}'_\ell;\theta). \tag{2}$$

It is worth to emphasize that the choice of the objective $\ell(\mathbf{x}, \mathbf{x}'_\ell;\theta)$ plays a crucial role in generating powerful adversarial examples $\mathbf{x}'_\ell$. The objective function ℓ traditionally used is the Adversarial Cross-Entropy (ACE) [26]:

$$\ell_{\mathrm{ACE}}(\mathbf{x}, \mathbf{x}'_\ell;\theta) = \mathbb{E}_{Y|\mathbf{x}}\big[-\log q_{\widehat{Y}|X}(Y|\mathbf{x}'_\ell;\theta)\big], \tag{3}$$

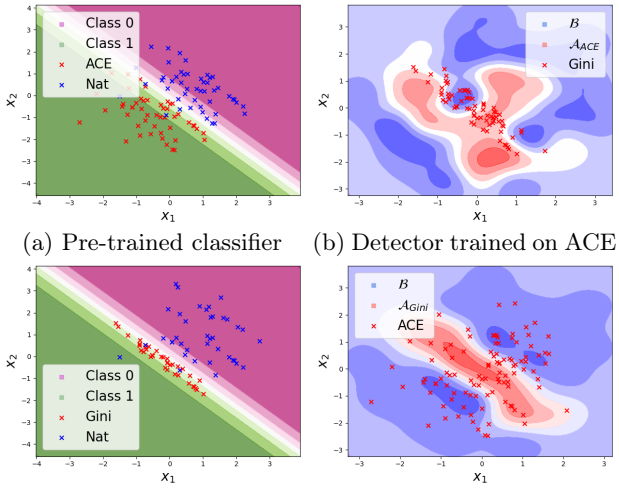
It is possible to use any objective function ℓ to craft adversarial samples. We present the three losses that we use to generate adversarial examples in the following. While these losses have already been considered in detection/robustness cases, to the best of our knowledge, they have never been used to craft attacks to test the performances of detection methods.

3.2 Three New Objective Functions

The Kullback-Leibler divergence. The Kullback-Leibler (KL) divergence between the natural and the adversarial probability distributions has been widely used in different learning problems, as building training losses for robust models [37]. KL is defined as follows:

$$\ell_{\mathrm{KL}}\left(\mathbf{x}, \mathbf{x}'_\ell;\theta\right) = \mathbb{E}_{\widehat{Y}|\mathbf{x};\theta}\left[\log\left(\frac{q_{\widehat{Y}|X}(\widehat{Y}|\mathbf{x};\theta)}{q_{\widehat{Y}|X}(\widehat{Y}|\mathbf{x}'_\ell;\theta)}\right)\right]. \tag{4}$$

[1] Throughout the paper, when the values of ε and p are clear from the context, we denote the attack mechanism as $a_\ell(\cdot)$.



(a) Pre-trained classifier (b) Detector trained on ACE

(c) Pre-trained classifier (d) Detector trained on Gini

Fig. 1: Decision boundary for the binary classifier 1a-1c: the decision region for class 1 is green, the decision region of class 0 is pink. The natural testing samples belonging to class 0 are reported in blue, the corresponding adversarial examples crafted using ACE (1a) and Gini Impurity (1c) in red. Decision boundary of the detectors 1b-1d: $\mathcal{B}$, the decision region of the natural examples; $\mathcal{A}_\ell$, reported in red shades, the decision region of the adversarial examples when the detector is trained on data points crafted via $\ell \in \{\text{ACE}, \text{Gini}\}$ as objective. The darker shades stand for higher confidence. The red points represent the adversarial examples created with the opposite loss (respectively $\ell \in \{\text{Gini}, \text{ACE}\}$).

The Fisher-Rao objective. The Fisher-Rao (FR) distance is an information-geometric measure of dissimilarity between soft-predictions [5]. It has been recently used to craft a new regularizer for robust classifiers [31]. FR can be computed as follows:

$$\ell_{\text{FR}}(\mathbf{x}, \mathbf{x}'_\ell; \theta) = 2 \arccos \left(\sum_{y \in \mathcal{Y}} \sqrt{q_{\widehat{Y}|X}(y|\mathbf{x};\theta) q_{\widehat{Y}|X}(y|\mathbf{x}'_\ell;\theta)} \right). \tag{5}$$

The Gini Impurity score. The Gini Impurity score approximates the probability of incorrectly classifying the input $\mathbf{x}$ if it was randomly labeled according to the model's output distribution $q_{\widehat{Y}|X}(y|\mathbf{x}'_\ell;\theta)$. It was recently used in [15] to determine whether a sample is correctly or incorrectly classified.

$$\ell_{\text{Gini}}(\cdot, \mathbf{x}'_\ell; \theta) = 1 - \sqrt{\sum_{y \in \mathcal{Y}} q^2_{\widehat{Y}|X}(y|\mathbf{x}'_\ell;\theta)}. \tag{6}$$

3.3 A case study: ACE vs. Gini Impurity

In Figure 1 we provide insights on why we need to evaluate the detectors on attacks crafted through different objectives. We create a synthetic dataset that consists of 300 data points drawn from $\mathcal{N}_0 = \mathcal{N}(\mu_0, \sigma^2\mathbf{I})$ and 300 data points drawn from $\mathcal{N}_1 = \mathcal{N}(\mu_1, \sigma^2\mathbf{I})$, where $\mu_0 = [1\ 1]$, $\mu_1 = [-1\ -1]$ and $\sigma = 1$. To each data point $\mathbf{x}$ is assigned true label 0 or 1 depending on whether $\mathbf{x} \sim \mathcal{N}_0$ or $\mathbf{x} \sim \mathcal{N}_1$, respectively. The data points have been split between the training set (70%) and the testing set (30%). We finally train a simple binary classifier with one single hidden layer and a learning rate of 0.01 for 20 epochs. We attack the classifier by generating adversarial examples with PGD under the L_∞-norm constraints with $\varepsilon = 1.2$ for the ACE attacks and $\varepsilon = 5$ for the Gini Impurity attacks to have a classification accuracy (classifier performance) of 50% on the corrupted data points. In Figure 1a-1c we plot the decision boundary of the binary classifier together with the adversarial and natural examples belonging to class 0. As can be seen, ACE creates points that lie in the opposite decision region with respect to the original points (Fig. 1a). Conversely, Gini Impurity tends to create new data points in the region of maximal uncertainty of the classifier (Fig. 1c). Consider the scenario where we train a simple Radial Basis Function (RBF) kernel SVM on a subset of the testing set of the natural points together with the attacked examples, generated with the ACE or the Gini Impurity score depending on the case (Fig. 1b-1d). We then test the detector on the data points originated with the opposite loss, Figure 1b and Figure 1d respectively. The decision region of the detector for natural examples is in blue, and the one for the adversarial examples is in red. The intensity of the color corresponds to the level of certainty of the detector. The accuracy of the detector on natural and adversarial data points decreases from 71% to 62% when changing to the opposite loss in Fig. 1b, and from 87% to 63% in Fig. 1d. Hence, testing on samples crafted using a different loss in Eq. (2) means changing the attack and, consequently, evaluating detectors without taking into consideration this possibility leads to a more biased and unrealistic estimation of their performance. When the detector is trained on the adversarial examples created with both the losses, the accuracy is 79.8% when testing on Gini and 66.3% when testing on ACE, which is a better trade-off in adversarial detection performances.

The aforementioned losses will be included in the following section to design MEAD, our *multi-armed evaluation framework*, a new method to evaluate the performance of adversarial detection with low bias.

4 Evaluation with a Multi-Armed Attacker

The proposed evaluation framework, MEAD, consists in testing an adversarial examples detection method on a large collection of attacks grouped w.r.t. the L_p-norm and the maximal perturbation ε they consider. Each given natural input example is perturbed according to the collection of attacks. Note that, for every attack, a perturbed example is considered adversarial *if and only if* it fools the

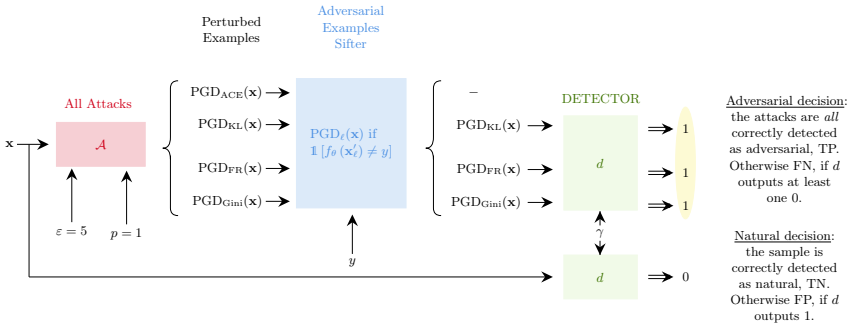


Fig. 2: MEAD: $\mathbf{x}$ is the natural example, $\varepsilon = 5$ is the perturbation magnitude, L_1 is the norm. From the set of all the possible existing attacks $\mathcal{A}$ we consider the ones using PGD. The sifter discards all the perturbed samples that do not fool the classifier f_θ. d is the detector.

classifier. Otherwise, it is discarded and will not influence the evaluation. We then feed all the natural and successful adversarial examples to the detector and gather all the predictions. Finally, based on the detection decisions, we evaluate the detector according to a worst-case scenario:
i) Adversarial decision: for each natural example, we gather all the successful adversarial examples. If the detector detects *all* of them, then the perturbed sample is considered *correctly detected* (i.e., it is a true positive). However, if the detector misses at least one of them, the noisy sample is considered *undetected* (i.e., it is a false negative).
ii) Natural decision: for each natural sample, if the detector does not detect it, then the sample is considered *correctly non-detected* (i.e., it is a true negative); otherwise it is *incorrectly detected* (i.e., it is a false positive).

Specifically, let $\mathcal{D}_m = \{(\mathbf{x}_i, y_i)\}_{i=1}^m \sim P_{XY}$ be the testing set of size m, where $\mathbf{x}_i \in \mathcal{X}$ is the natural input sample and $y_i \in \mathcal{Y}$ is its true label. Let d: $\mathcal{X} \times \mathbb{R} \to \{0, 1\}$ be the detection mechanism and $a_\ell : \mathcal{X} \times \mathbb{R} \times \{1, 2, \infty\} \to \mathcal{X}$ the attack strategy according to the objective function $\ell \in \mathcal{L}$ within a selected collection of objectives $\mathcal{L}$ as described in Sec. 3. For every considered L_p-norm, $p \in \{1, 2, \infty\}$, maximal perturbation $\varepsilon \in \mathbb{R}$, and every threshold $\gamma \in \mathbb{R}$[2]:

$$TP_{\varepsilon,p}(\gamma) = \Big\{(\mathbf{x}, y) \in \mathcal{D}_m : \forall \ell \in \mathcal{L}\ \{f_\theta\big(a_\ell(\mathbf{x})\big) \neq y\} \ \wedge\ \{d\big(a_\ell(\mathbf{x}), \gamma\big) = 1\}\Big\} \quad (7)$$

$$FN_{\varepsilon,p}(\gamma) = \Big\{(\mathbf{x}, y) \in \mathcal{D}_m : \exists \ell \in \mathcal{L}\ \{f_\theta\big(a_\ell(\mathbf{x})\big) \neq y\} \ \wedge\ \{d\big(a_\ell(\mathbf{x}), \gamma\big) = 0\}\Big\} \quad (8)$$

$$TN_{\varepsilon,p}(\gamma) = \{(\mathbf{x}, y) \in \mathcal{D}_m : d(\mathbf{x}, \gamma) = 0\} \quad (9)$$

$$FP_{\varepsilon,p}(\gamma) = \{(\mathbf{x}, y) \in \mathcal{D}_m : d(\mathbf{x}, \gamma) = 1\}. \quad (10)$$

In Fig. 2 we provide a graphical interpretation of MEAD when the perturbation magnitude and the norm are fixed.

[2] With an abuse of notation, $\forall \ell \in \mathcal{L}$ stands for all the considered attack mechanisms for specific values of ε, p within a collection of objectives $\mathcal{L}$.

5 Experiments

In this section, we assess the effectiveness of the proposed evaluation framework, MEAD. The code is available at https://github.com/meadsubmission/MEAD.

5.1 Experimental setting

Evaluation metrics. For each L_p-norm and each considered ε, we apply our multi-armed detection scheme. We gather the global result considering all the attacks and all the objectives. Moreover, we also report the results per objective. The performance is measured in terms of the AUROC↑ [11] and in terms of $\text{FPR}\downarrow_{95\%}$. The first metric is the *Area Under the Receiver Operating Characteristic curve* and represents the ability of the detector to discriminate between adversarial and natural examples (higher is better). The second metric represents the percentage of natural examples detected as adversarial when 95 % of the adversarial examples are detected, i.e., FPR at 95 % TPR (lower is better).

Datasets and classifiers. We run the experiments on MNIST [20] and CIFAR10 [18]. The underlying classifiers are a simple CNN for MNIST, consisting of two blocks of two convolutional layers, a max-pooling layer, one fully-connected layer, one dropout layer, two fully-connected layers, and ResNet-18 for CIFAR10. The training procedures involve 100 epochs with Stochastic Gradient Descent (SGD) optimizer using a learning rate of 0.01 for the simple CNN and 0.1 for ResNet-18; a momentum of 0.9 and a weight decay of 10^{-5} for ResNet-18. Once trained, these networks are fixed and never modified again.

Grouping attacks. We test the methods on the attacks presented in Sec. 2.2, and we present them based on the norm constraint used to construct the attacks.Under the L_1-norm fall PGD with ε in $\{5, 10, 15, 20, 25, 30, 40\}$. Under the L_2-norm fall PGD with ε in $\{0.125, 0.25, 0.3125, 0.5, 1, 1.5, 2\}$, CW with $\varepsilon = 0.01$, HOP with $\varepsilon = 0.1$, and DF which has no constraint on ε. Under the L_∞-norm fall FGSM, BIM and PGD with ε in $\{0.0315, 0.0625, 0.125, 0.25, 0.3125, 0.5\}$, CW with $\varepsilon = 0.3125$, and SA with $\varepsilon = 0.3125$ for MNIST and $\varepsilon = 0.125$ for CIFAR10. Finally, ST is not constrained by a norm or a maximum perturbation, as it is limited in maximum rotation (30 for CIFAR10 and 60 for MNIST) and translation (8 for CIFAR10 and 10 for MNIST).

Detection Methods. We tested dection methods introduced in Sec. 2.1. In the supervised case, we train the detectors using adversarial examples created by perturbing the samples in the original training sets with PGD under L_∞-norm and $\varepsilon = 0.03125$. In the unsupervised case, the detectors only need natural samples in the training sets. They are tested on all the previously mentioned attacks, generated on the testing sets.

Table 1: Overall performances on CIFAR10 of all the detectors per objective and in MEAD. The worst results among all the settings is in **bold**; the ones in the single-armed setting is underlined. No norm denotes the group of attacks that do not depend on the norm constraint.

CIFAR10	MEAD		ACE		KL		Gini		FR	
NSS	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%
L_1 Average	**62.9**	**81.6**	67.4	75.7	67.1	76.0	67.8	78.2	67.6	75.6
L_2 Average	**64.0**	**82.0**	68.7	71.0	68.5	70.9	65.1	82.0	68.6	71.1
L_∞ Average	**71.9**	**62.0**	76.9	40.1	77.2	39.5	73.7	59.6	74.1	57.2
No norm	**88.5**	**38.8**	**88.5**	**38.8**	**88.5**	**38.8**	**88.5**	**38.8**	**88.5**	**38.8**
KD-BU	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%
L_1 Average	**50.9**	**95.7**	70.0	88.6	70.0	88.4	74.3	92.3	69.8	88.4
L_2 Average	**59.0**	**94.1**	71.6	71.9	71.7	71.6	70.6	92.8	71.7	71.8
L_∞ Average	**36.8**	**96.9**	64.8	92.1	68.1	91.3	53.7	95.6	67.8	91.7
No norm	**65.4**	**94.2**	**65.4**	**94.2**	**65.4**	**94.2**	**65.4**	**94.2**	**65.4**	**94.2**
LID	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%
L_1 Average	**50.8**	**95.4**	69.6	82.1	69.4	82.9	88.9	49.9	69.1	83.7
L_2 Average	**63.5**	**83.1**	73.7	70.1	73.4	70.7	82.5	61.3	73.2	71.3
L_∞ Average	**53.8**	**90.8**	75.7	56.8	79.9	57.6	71.3	79.7	82.0	51.4
No norm	**88.0**	**58.1**	**88.0**	**58.1**	**88.0**	**58.1**	**88.0**	**58.1**	**88.0**	**58.1**
FS	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%
L_1 Average	75.4	64.8	92.8	25.1	92.9	24.9	**73.5**	**67.6**	92.9	24.6
L_2 Average	74.9	65.8	87.4	31.2	87.6	36.9	**73.7**	**67.2**	87.4	37.5
L_∞ Average	**52.7**	**81.1**	73.0	60.1	77.5	55.7	58.2	78.8	75.7	58.5
No norm	**62.7**	**82.5**	**62.7**	**82.5**	**62.7**	**82.5**	**62.7**	**82.5**	**62.7**	**82.5**
MagNet	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%
L_1 Average	49.6	93.7	49.8	93.5	49.7	93.3	50.1	93.2	**49.1**	**93.8**
L_2 Average	50.9	93.1	52.3	89.6	52.3	89.4	**50.5**	**93.3**	51.8	91.4
L_∞ Average	**78.0**	**46.1**	79.2	44.6	80.2	44.1	79.2	44.6	80.0	44.6
No norm	**79.9**	**45.7**	**79.9**	**45.7**	**79.9**	**45.7**	**79.9**	**45.7**	**79.9**	**45.7**

Table 2: Overall performances on MNIST of all the detectors per objective and in MEAD. The worst results among all the settings is in **bold**; the ones in the single-armed setting is underlined. No norm denotes the group of attacks that do not depend on the norm constraint.

MNIST	MEAD		ACE		KL		Gini		FR	
NSS	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%
L_1 Average	**96.8**	**9.4**	97.0	8.2	97.1	8.6	97.4	7.0	97.1	8.1
L_2 Average	**90.3**	**26.5**	90.7	25.8	90.8	25.4	91.4	23.7	90.6	**26.5**
L_∞ Average	**88.7**	23.5	89.5	23.5	89.5	**23.6**	90.0	**23.6**	89.8	23.5
No norm	**87.1**	**57.8**	**87.1**	**57.8**	**87.1**	**57.8**	**87.1**	**57.8**	**87.1**	**57.8**
KD-BU	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%
L_1 Average	**45.6**	**95.7**	59.9	93.0	59.3	93.1	61.4	92.7	58.9	93.3
L_2 Average	**50.3**	**94.8**	59.9	93.0	59.7	93.1	59.3	93.2	59.8	93.0
L_∞ Average	**34.1**	**96.7**	42.8	96.0	44.7	95.8	48.6	95.3	44.9	95.8
No norm	**76.0**	**88.2**	**76.0**	**88.2**	**76.0**	**88.2**	**76.0**	**88.2**	**76.0**	**88.2**
LID	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%
L_1 Average	**79.9**	**54.9**	83.7	48.2	84.0	50.0	90.4	52.1	84.1	50.2
L_2 Average	**85.6**	**46.2**	87.4	44.1	87.0	45.1	87.6	44.4	86.1	45.4
L_∞ Average	**77.9**	**55.1**	83.3	46.3	83.6	47.8	88.7	38.8	83.0	49.5
No norm	**98.1**	**8.2**	**98.1**	**8.2**	**98.1**	**8.2**	**98.1**	**8.2**	**98.1**	**8.2**
FS	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%
L_1 Average	**79.8**	**66.8**	83.4	57.6	83.5	57.1	83.2	53.0	83.4	57.4
L_2 Average	**73.5**	**69.0**	75.6	65.0	75.5	65.4	74.5	67.0	74.7	65.7
L_∞ Average	**76.4**	**63.5**	80.8	54.6	80.2	54.6	79.0	58.7	80.4	58.2
No norm	**61.5**	**85.9**	**61.5**	**85.9**	**61.5**	**85.9**	**61.5**	**85.9**	**61.5**	**85.9**
MagNet	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%	AUROC↑%	$FPR_{\downarrow 95\%}$%
L_1 Average	**98.1**	5.7	98.2	5.4	98.3	**5.6**	98.3	5.2	**98.1**	**5.6**
L_2 Average	90.0	28.7	90.6	27.6	90.8	27.8	90.6	**29.1**	**89.7**	28.1
L_∞ Average	98.5	10.3	98.5	10.3	**98.4**	**10.6**	98.5	10.5	98.5	10.4
No norm	**86.9**	**74.3**	**86.9**	**74.3**	**86.9**	**74.3**	**86.9**	**74.3**	**86.9**	**74.3**

5.2 Experimental results

In this section, we refer to *single-armed setting* when we consider the setup where the adversarial examples are generated w.r.t. one of the objectives in Sec. 3. We provide the average of the performances of all the detection methods on CIFAR10 in Tab. 1 and on MNIST in Tab. 2. Due to space constraints, the detailed tables for each detection method (i.e., *NSS*, *LID*, *KD-BU*, *MagNet*, and *FS*) and for each dataset (i.e., CIFAR10 and MNIST) are reported in Appendix A.

MEAD and the single-armed setting. Table 1 shows a decrease in the performance of all the detectors when going from the single-armed setting to MEAD. *NSS* is the more robust among the supervised methods when passing from the single-armed setting to the proposed setting. Indeed, (cf Tab. 1), in terms of AUROC↑, it registers a decrease of up 4.9 percentage points under the L_1-norm constraint, 4.7 under the L_2-norm constraint, and 5.3 under the L_∞-norm constraint. This can be explained by the fact that the network in *NSS* is trained on the natural scene statistics extracted from the trained samples differently from the other detectors. In particular, these statistical properties are altered by the presence of adversarial perturbations and hence are found to be a good candidate to determine if a sample is adversarial or not. By looking closely at the results for *NSS* in Tab. 5, it comes out that it performs better when evaluated on L_∞ norm constraint. Indeed, in this case, the adversarial examples at testing time are similar to those used at training time. Not surprisingly, the performance decreases when evaluated on other kinds of attacks. Notice that, in the single-armed setting, all the supervised methods turn out to be much more inefficient than when presented in the original papers. Indeed, as already explained in Sec. 5.1, we train the detectors using adversarial examples created by perturbing the samples in the original training sets with PGD under L_∞-norm and $\varepsilon = 0.03125$, and then we test them on a variety of attacks. Hence, we do not train a different detector for each kind of attack seen at testing time. On the other side, the unsupervised detector *MagNet* appears to be more robust than *FS* when changing from the single-armed setting to MEAD. Indeed, in terms of AUROC↑, it loses at most 2.2 percentage points (L_∞ norm case). On average, *FS* is the unsupervised detector that achieves the best performance on CIFAR10, while *MagNet* is the one to achieve the best performance on MNIST.

Remark: Some single-armed setting results turn out to be worse than the corresponding results in MEAD (cf Tab. 5-9 and Tab. 11-15 in Appendix A). We provide here an explanation of this phenomenon. Given a natural input sample $\mathbf{x}$, let $\mathbf{x}_\ell$ denotes the perturbed version of $\mathbf{x}$ according to some fixed norm p, fixed perturbation magnitude ε and objective function ℓ between ACE, KL, Gini and FR. Suppose $f_\theta(\mathbf{x}_{\text{ACE}}) = y$, where y is the ground true label of $\mathbf{x}$, this means that $\mathbf{x}_{\text{ACE}}$ is a perturbed version of the natural example but not adversarial. Assume instead $f_\theta(\mathbf{x}_{\text{KL}}) \neq y$, $f_\theta(\mathbf{x}_{\text{Gini}}) \neq y$ and $f_\theta(\mathbf{x}_{\text{FR}}) \neq y$. If at testing time the detector is able to recognize all of them as being positive (i.e., adversarial), then under MEAD, $\mathbf{x}_{\text{KL}}, \mathbf{x}_{\text{Gini}}, \mathbf{x}_{\text{FR}}$ would be considered a *true positive*. This example,

counting as a true positive under MEAD, would instead be discarded under the single-armed setting of ACE, as $\mathbf{x}_{\text{ACE}}$ is neither a clean example nor an adversarial one. Then, the larger amount of true positives in MEAD can potentially lead to an increase in the global AUROC↑.

Effectiveness of the proposed objective functions. In Tab. 4 and Tab. 10, relegated to the Appendix due to space constraints, we report the averaged number of successful adversarial examples under the multi-armed setting as well as the details per single-armed settings on CIFAR10 and MNIST, respectively. The attacks are most successful when the value of the constraint ε for every L_p-norm increases. Generating adversarial examples using the ACE for each attack scheme creates more harmful (adversarial) examples for the classifier than using any other objective. However, using either the Gini Impurity score, the Fisher-Rao objective, or the Kullback-Leibler divergence seems to create examples that are either equally or more difficult to be detected by the detection methods. For this purpose, we provide two examples. First, by looking at the results in Tab. 7, we can deduce that *LID* finds it difficult to recognize the attacks based on KL and FR objective functions but not the ones created through Gini. For example, with PGD1 and $\varepsilon = 40$, we register a decrease in AUROC↑ of 9.5 percentage points when going from the single-armed setting of Gini to the one of FR. Similarly, the decrease is 8.3 percentage points in the case of KL. This behavior is even more remarkable when we look at the results in terms of FPR↓$_{95\%}$: the gap between the best FPR↓$_{95\%}$ values (obtained via Gini) and the worst (via FR) is 30.7 percentage points. On the other side, the situation is reversed if we look at the results in Tab. 8 as *FS* turns out to be highly inefficient at recognizing adversarial examples generated via the Gini Impurity score. By considering the results associated to the highest value of ε for each norm, namely $\varepsilon = 40$ for L_1-norm; $\varepsilon = 2$ for L_2-norm; $\varepsilon = 0.5$ for L_∞-norm, the gap between best FPR↓$_{95\%}$ values (obtained via KL divergence) and the worst (via Gini Impurity score), varies from a minimum of 41.7 (L_∞-norm) to a maximum of 64.4 (L_2-norm) percentage points. This example, in agreement with Sec. 3.3, testify on real data that testing the detectors without taking into consideration the possibility of creating attacks through different objective functions leads to a biased and unrealistic estimation of their performances.

Comparison between supervised and unsupervised detectors. The unsupervised methods find it challenging to recognize attacks crafted using the Gini Impurity score. Indeed, according to Sec. 3.3, that objective function creates attacks on the decision boundary of the pre-trained classifier. Consequently, the unsupervised detectors can easily associate such input samples with the cluster of naturals. Supervised methods detect Adversarial Cross-Entropy loss-based attacks more and, therefore, more volatile when it comes to other types of loss-based attacks. Overall, by looking at the results in Tab. 3 on both the datasets, most of the supervised and unsupervised methods achieve comparable performances with the multi-armed framework, meaning that the current use of the

Table 3: Performances of each detection method under the MEAD framework on CIFAR10 and MNIST averaged over the norm-based constraint. The best results among all the methods is in **bold**; the ones per type of detection method (i.e. Supervised and Unsupervised) are underlined.

	Supervised Methods						Unsupervised Methods					
	NSS		*KD-BU*		*LID*		*FS*		*MagNet*			
	AUROC↑%	FPR$_{\downarrow 95\%}$%	AUROC↑%	FPR$_{\downarrow 95\%}$%	AUROC↑%	FPR$_{\downarrow 95\%}$%	AUROC↑%	FPR$_{\downarrow 95\%}$%	AUROC↑%	FPR$_{\downarrow 95\%}$%		
MNIST	90.7	**29.3**	51.5	93.9	85.4	41.1	72.8	71.3	**93.4**	29.8		
CIFAR10	**71.8**	**66.1**	53.0	95.2	64.0	81.8	66.4	73.6	64.6	69.7		

knowledge about the specific attack is not general enough. The exception to this is *NSS*, which, as already explained, seems to be the most general detector.

On the effects of the norm and ε. The detection methods recognize attacks with a large perturbation more easily than other attacks (cf Tab. 5-9 and Tab. 11-15). L_∞-norm attacks are less easily detectable than any other L_p-norm attack. Indeed, multiple attacks are tested simultaneously for a single ε under the L_∞ norm constraint. For example, in CIFAR10 with $\varepsilon = 0.3125$ and L_∞, PGD, FGSM, BIM, and CW are tested together, whereas, with any other norm constraint, only one typology of attack is examined. Indeed the more attack we consider for a given ε, the more likely at least one attack will remain undetected. Globally, under the L_∞-norm constraint, Gini Impurity score-based attacks are the least detected attacks. However, each method has different behaviors under L_1 and L_2. *NSS* is more sensitive to Kullback-Leibler divergence-based attacks while *MagNet* is more volatile to the Fisher-Rao distance-based attacks. As already pointed out, *FS* achieves inferior performance when evaluated against attacks crafted through the Gini Impurity objective, while the sensitivity of *LID* and *KD-BU* to a specific objective depends on the L_p-norm constraint.

6 Summary and Concluding Remarks

We introduced MEAD a new framework to evaluate detection methods of adversarial examples. Contrary to what is generally assumed, the proposed setup ensures that the detector does not know the attacks at the testing time and is evaluated based on simultaneous attack strategies. Our experiments showed that the SOTA detectors for adversarial examples (both supervised and unsupervised) mostly fail when evaluated in MEAD with a remarkable deterioration in performance compared to single-armed settings. We enrich the proposed evaluation framework by involving three new objective functions to generate adversarial examples that create adversarial examples which can simultaneously fool the classifier while not being successfully identified by the investigated detectors. The poor performance of the current SOTA adversarial examples detectors should be seen as a challenge when developing novel methods. However, our evaluation framework assumes that the attackers do not know the detection method. As future work we plan to enrich the framework to a complete whitebox scenario.

Acknowledgements The work of Federica Granese was supported by the European Research Council (ERC) project HYPATIA under the European Union's Horizon 2020 research and innovation program. Grant agreement №835294. This work has been supported by the project PSPC AIDA: 2019-PSPC-09 funded by BPI-France. This work was performed using HPC resources from GENCI-IDRIS (Grant 2022-[AD011012352R1]) and thanks to the Saclay-IA computing platform.

References

1. Aldahdooh, A., Hamidouche, W., Déforges, O.: Revisiting model's uncertainty and confidences for adversarial example detection (2021) 3
2. Aldahdooh, A., Hamidouche, W., Fezza, S.A., Déforges, O.: Adversarial example detection for DNN models: A review. CoRR **abs/2105.00203** (2021) 1, 3
3. Andriushchenko, M., Croce, F., Flammarion, N., Hein, M.: Square attack: a query-efficient black-box adversarial attack via random search. In: European Conference on Computer Vision. pp. 484–501. Springer (2020) 4
4. Athalye, A., Carlini, N., Wagner, D.A.: Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In: Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018. Proceedings of Machine Learning Research, vol. 80, pp. 274–283. PMLR (2018) 1
5. Atkinson, C., Mitchell, A.F.S.: Rao's distance measure. Sankhyā: The Indian Journal of Statistics, Series A (1961-2002) **43**(3), 345–365 (1981) 6
6. Carlini, N., Wagner, D.: Towards evaluating the robustness of neural networks. In: 2017 IEEE Symposium on Security and Privacy (SP). pp. 39–57. IEEE (2017) 5
7. Carrara, F., Becarelli, R., Caldelli, R., Falchi, F., Amato, G.: Adversarial examples detection in features distance spaces. In: Computer Vision - ECCV 2018 Workshops - Munich, Germany, September 8-14, 2018, Proceedings, Part II. Lecture Notes in Computer Science, vol. 11130, pp. 313–327. Springer (2018) 3
8. Chen, J., Jordan, M.I., Wainwright, M.J.: Hopskipjumpattack: A query-efficient decision-based attack. In: 2020 ieee symposium on security and privacy (sp). pp. 1277–1294. IEEE (2020) 4
9. Croce, F., Andriushchenko, M., Sehwag, V., Flammarion, N., Chiang, M., Mittal, P., Hein, M.: Robustbench: a standardized adversarial robustness benchmark. CoRR **abs/2010.09670** (2020) 2
10. Croce, F., Hein, M.: Minimally distorted adversarial examples with a fast adaptive boundary attack. In: International Conference on Machine Learning. pp. 2196–2205. PMLR (2020) 1
11. Davis, J., Goadrich, M.: The relationship between precision-recall and roc curves. In: Proceedings of the 23rd international conference on Machine learning. pp. 233–240 (2006) 9
12. Engstrom, L., Tran, B., Tsipras, D., Schmidt, L., Madry, A.: Exploring the landscape of spatial robustness. In: International Conference on Machine Learning. pp. 1802–1811. PMLR (2019) 4
13. Feinman, R., Curtin, R.R., Shintre, S., Gardner, A.B.: Detecting adversarial samples from artifacts. CoRR **abs/1703.00410** (2017) 3, 4
14. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. International Conference on Learning Representations (2015) 4
15. Granese, F., Romanelli, M., Gorla, D., Palamidessi, C., Piantanida, P.: DOCTOR: A simple method for detecting misclassification errors. In: Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual. pp. 5669–5681 (2021) 6
16. Grosse, K., Manoharan, P., Papernot, N., Backes, M., McDaniel, P.D.: On the (statistical) detection of adversarial examples. CoRR **abs/1702.06280** (2017) 3
17. Kherchouche, A., Fezza, S.A., Hamidouche, W., Déforges, O.: Natural scene statistics for detecting adversarial examples in deep neural networks. In: 22nd IEEE

International Workshop on Multimedia Signal Processing, MMSP 2020, Tampere, Finland, September 21-24, 2020. pp. 1–6. IEEE (2020) 1, 3, 4
18. Krizhevsky, A.: Learning multiple layers of features from tiny images. Tech. rep. (2009) 9
19. Kurakin, A., Goodfellow, I., Bengio, S., et al.: Adversarial examples in the physical world (2016) 4
20. LeCun, Y., Cortes, C., Burges, C.: Mnist handwritten digit database (2010) 9
21. Li, X., Li, F.: Adversarial examples detection in deep networks with convolutional filter statistics. In: IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017. pp. 5775–5783. IEEE Computer Society (2017) 3
22. Liang, B., Li, H., Su, M., Li, X., Shi, W., Wang, X.: Detecting adversarial image examples in deep neural networks with adaptive noise reduction. IEEE Trans. Dependable Secur. Comput. **18**(1), 72–85 (2021) 3
23. Lu, J., Issaranon, T., Forsyth, D.A.: Safetynet: Detecting and rejecting adversarial examples robustly. In: IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017. pp. 446–454. IEEE Computer Society (2017) 3
24. Ma, S., Liu, Y., Tao, G., Lee, W., Zhang, X.: NIC: detecting adversarial samples with neural network invariant checking. In: 26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019. The Internet Society (2019) 1, 3
25. Ma, X., Li, B., Wang, Y., Erfani, S.M., Wijewickrema, S.N.R., Schoenebeck, G., Song, D., Houle, M.E., Bailey, J.: Characterizing adversarial subspaces using local intrinsic dimensionality. In: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. OpenReview.net (2018) 3, 4
26. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. In: International Conference on Learning Representations (2018) 2, 4, 5
27. Meng, D., Chen, H.: Magnet: A two-pronged defense against adversarial examples. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017. pp. 135–147. ACM (2017) 1, 3, 4
28. Metzen, J.H., Genewein, T., Fischer, V., Bischoff, B.: On detecting adversarial perturbations. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net (2017) 3
29. Moosavi-Dezfooli, S.M., Fawzi, A., Frossard, P.: Deepfool: a simple and accurate method to fool deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2574–2582 (2016) 4
30. Pang, T., Du, C., Dong, Y., Zhu, J.: Towards robust detection of adversarial examples. In: Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada. pp. 4584–4594 (2018) 3
31. Picot, M., Messina, F., Boudiaf, M., Labeau, F., Ben Ayed, I., Piantanida, P.: Adversarial robustness via fisher-rao regularization. IEEE Transactions on Pattern Analysis and Machine Intelligence pp. 1–1 (2022) 2, 6
32. Sotgiu, A., Demontis, A., Melis, M., Biggio, B., Fumera, G., Feng, X., Roli, F.: Deep neural rejection against adversarial examples. EURASIP J. Inf. Secur. **2020**, 5 (2020) 3

33. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing properties of neural networks. International Conference on Learning Representations (2014) 5
34. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I.J., Fergus, R.: Intriguing properties of neural networks. In: 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings (2014) 1
35. Tao, G., Ma, S., Liu, Y., Zhang, X.: Attacks meet interpretability: Attribute-steered detection of adversarial samples. In: Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada. pp. 7728–7739 (2018) 3
36. Xu, W., Evans, D., Qi, Y.: 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) 3, 4
37. Zhang, H., Yu, Y., Jiao, J., Xing, E.P., Ghaoui, L.E., Jordan, M.I.: Theoretically principled trade-off between robustness and accuracy. In: International Conference on Machine Learning. pp. 1–11 (2019) 5
38. Zheng, S., Song, Y., Leung, T., Goodfellow, I.J.: Improving the robustness of deep neural networks via stability training. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016. pp. 4480–4488. IEEE Computer Society (2016) 2
39. Zheng, T., Chen, C., Ren, K.: Distributionally adversarial attack. In: The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019. pp. 2253–2260. AAAI Press (2019) 1
40. Zheng, Z., Hong, P.: Robust detection of adversarial attacks by modeling the intrinsic properties of deep neural networks. In: Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada. pp. 7924–7933 (2018) 3