Detecting Adversarial Examples from Sensitivity Inconsistency of Spatial-Transform Domain

Jinyu Tian,[1] **Jiantao Zhou,**[1,*] **Yuanman Li,**[2] **Jia Duan**[1]
[1]State Key Laboratory of Internet of Things for Smart City,
Department of Computer and Information Science, University of Macau
[2]Guangdong Key Laboratory of Intelligent Information Processing,
College of Electronics and Information Engineering, Shenzhen University
{yb77405, jtzhou}@um.edu.mo, yuanmanli@szu.edu.cn, xuelandj@gmail.com

Abstract

Deep neural networks (DNNs) have been shown to be vulnerable against adversarial examples (AEs), which are maliciously designed to cause dramatic model output errors. In this work, we reveal that normal examples (NEs) are insensitive to the fluctuations occurring at the highly-curved region of the decision boundary, while AEs typically designed over one single domain (mostly spatial domain) exhibit exorbitant sensitivity on such fluctuations. This phenomenon motivates us to design another classifier (called dual classifier) with transformed decision boundary, which can be collaboratively used with the original classifier (called primal classifier) to detect AEs, by virtue of the sensitivity inconsistency. When comparing with the state-of-the-art algorithms based on Local Intrinsic Dimensionality (LID), Mahalanobis Distance (MD), and Feature Squeezing (FS), our proposed Sensitivity Inconsistency Detector (SID) achieves improved AE detection performance and superior generalization capabilities, especially in the challenging cases where the adversarial perturbation levels are small. Intensive experimental results on ResNet and VGG validate the superiority of the proposed SID.

Introduction

Deep neural networks (DNNs) have achieved the state-of-the-art performance on a wide range of tasks including image classification (Krizhevsky, Sutskever, and Hinton 2012), speech recognition (Karpathy et al. 2014), etc. However, recent studies have shown that DNNs are vulnerable to crafted adversarial examples (AEs), which are generally imperceptible to the sense of humanity while being able to cause severe model output errors. Such vulnerability leads to serious security risks when deploying DNNs on critical scenarios, *e.g.*, self-driving cars. The attempts for defending against the threat of AEs can be roughly categorized into two types: robust classification and AE detection. The former type aims to eliminate the impact of the adversarial noise and make correct classification (*e.g.*, adversarial training (Schmidt et al. 2018), denoising, (Liao et al. 2018; Akhtar, Liu, and Mian 2018), defensive distillation (Papernot et al. 2016b), etc.). Though many robust classification strategies have been proposed, most of them are still not powerful enough to defeat the secondary attack or AEs generated by some advanced attacks *e.g.*, C&W (Carlini and Wagner 2017). In fact, in (Tsipras et al. 2019), it was pointed out that the true robustness would lead to depreciation in accuracy. Alternatively, a weaker version of the defense is just to detect the AEs, while not ambiguously rectifying the classification results. In many practical applications, such detection is still quite meaningful, generating alarming signals to potential threats. Further, the AE detection could guide a better defense strategy (Sun et al. 2020).

An AE can be regarded as the translation of a normal example (NE) along the adversarial direction. Geometrically, the adversarial direction usually points toward the highly-curved region of the decision boundary, such that the decision boundary can be crossed with the minimized perturbation magnitude (Fawzi et al. 2018). A symmetric phenomenon we would like to point out is that if we **can intentionally cause fluctuations at the highly-curved regions of the decision boundary**, then those AEs could easily lead to different classification results, while NEs would exhibit quite consistent behavior. Such phenomenon reveals that, for NEs and AEs, there exists an inconsistency of sensitivity to boundary fluctuation at highly-curved regions. This motivates us to design another classifier with transformed decision boundary, which can be collaboratively used with the original classifier (called **primal classifier**) to detect AEs, by virtue of the sensitivity inconsistency. Ideally, the designed classifier (called **dual classifier**) should have dissimilar structures at the highly-curved regions with the primal classifier, while maintaining similar structures at the other regions.

To design the dual classifier satisfying the above requirements, we resort to the transform domain techniques, i.e., design the classifier over the transform domain, rather than the traditional spatial domain. In fact, some existing works (Liu et al. 2016; Tramèr et al. 2017) pointed out that models trained on the same feature domain of a given dataset tend to produce similar decision boundaries, especially their curved regions. This would explain why AE has the ability of transferable attack on models with different architectures. More specifically, we construct the dual classifier

[*]The corresponding author.

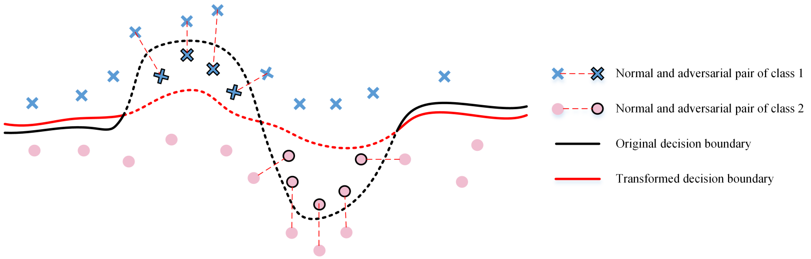

Normal and adversarial pair of class 1
Normal and adversarial pair of class 2
Original decision boundary
Transformed decision boundary

Figure 1: An illustrative example of the sensitivity of AEs against decision boundary fluctuation.

by using the *Weighted Average Wavelet Transform* (WAWT) and then propose a simple yet effective method for detecting AEs based on the sensitivity inconsistency of NEs and AEs. Our major contributions can be summarized as follows: 1) We reveal the existence of sensitivity inconsistency between NEs and AEs against the decision boundary transformation which is fulfilled by constructing the dual classifier. We theoretically prove the effectiveness of the designed dual classifier in affine and quadratic cases, and empirically demonstrate it for general cases; 2) Motivated by the sensitivity inconsistency, we define a feature to evaluate the sensitivity of an unknown example to the boundary transformation, and then propose a method called *Sensitivity Inconsistency Detector* (SID) to effectively detect AEs; and 3) When comparing with the state-of-the-art algorithms based on Local Intrinsic Dimensionality (LID) (Ma et al. 2018), Mahalanobis Distance (MD) (Lee et al. 2018), and Feature Squeezing (FS) (Xu, Evans, and Qi 2018), we observe improved detection performance and superior generalization capabilities, especially in the challenging cases where the perturbation levels are small. Experimental results on ResNet and VGG validate the superiority of the SID.

Notations: Without loss of generality, we restrict our attention to a K-class DNN classifier $\mathcal{F}$ trained on dataset $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$. For a NE $\mathbf{x}$, $\mathcal{F}$ calculates K logit values, i.e., $\mathcal{F}(\mathbf{x}) = \{f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_K(\mathbf{x})\}$, where $f_i(\mathbf{x})$ is the prediction confidence of $\mathbf{x}$ to the i-th class. The predicted label of $\mathbf{x}$ is denoted by $k(\mathbf{x}) = \texttt{argmax}_i\{f_i(\mathbf{x})\}$, for $i = 1, 2, \cdots, K$. The adversarial counterpart of $\mathbf{x}_i$ is written as $\hat{\mathbf{x}}_i = \mathbf{x}_i + \mathbf{r}_i$, where $\mathbf{r}_i$ is the adversarial perturbation. The set of all adversarial samples is expressed as $\hat{\mathbf{X}} = \{\hat{\mathbf{x}}_i\}_{i=1}^N$.

Related Works

In this section, we briefly review the state-of-the-art methods on adversarial attack and AE detection. As there are a large number of methods on both topics, we only mention some of the representatives.

Adversarial Attack: The adversarial attack is to craftily manipulate the normal input with imperceptible distortions to make the pre-trained model misclassify. Formally, an AE $\hat{\mathbf{x}}$ targeted on model $\mathcal{F}$ is essentially a correctly classified example $\mathbf{x}$ added with the adversarial perturbation $\mathbf{r}$ such that $k(\mathbf{x}) \neq k(\hat{\mathbf{x}})$. The added $\mathbf{r}$ is typically constrained by L_p norm (Sharif, Bauer, and Reiter 2018), i.e.,

$$\min_{\mathbf{r}} \|\mathbf{r}\|_p \quad \text{s.t.} \quad k(\mathbf{x}+\mathbf{r}) \neq k(\mathbf{x}). \tag{1}$$

Several adversarial attacks attempted to approximate the solution of this non-convex optimization problem with different searching and relaxation strategies. Goodfellow *et. al* proposed a method called Fast Gradient Sign Method (FGSM) to search for a feasible solution along the negative gradient sign direction of the cost function with an empirical step (Goodfellow, Shlens, and Szegedy 2015). To improve the attack performance, Kurakin *et. al* designed the Basic Iterative Method (BIM) (Kurakin, Goodfellow, and Bengio 2017) by adopting an iterative searching strategy. Further, Moosavi *et al.* suggested a powerful attack DeepFool (Moosavi-Dezfooli, Fawzi, and Frossard 2016), approximating the optimal solution of (1) based on Newton-Raphson method. In addition, Carlini and Wagner (C&W) (Carlini and Wagner 2017) relaxed the constraint in (1) with a margin loss, which can be integrated into the objective function and adaptively produce AEs with minimal perturbation levels. Instead of directly solving the problem (1), Jacobian Saliency Map Attack (JSMA) (Papernot et al. 2016a) constructs AEs through a greedy iterative procedure, altering only the pixels which contribute most to the correct classification. Some other recent works on adversarial attacks can be found in (Brendel, Rauber, and Bethge 2018; Eykholt et al. 2018; Brown et al. 2017) and references therein.

Detection of AEs: Most of the AE detection methods are based on the observation that AEs lie far from the distribution of NEs (Liu et al. 2019; Smith and Gal 2018; Song et al. 2018; Aigrain and Detyniecki 2019; Oberdiek, Rottmann, and Gottschalk 2018; Carrara et al. 2017; Sperl et al. 2019; Li and Li 2017). Along this line, Jan *et. al* (Metzen et al. 2017) augmented classification networks by subnetworks branching off the main network at some layers and produced a probability of the input being adversarial. Kathrin *et. al* (Grosse et al. 2017) detected AEs by using the statistical test to evaluate the confidence level of the "belonging" of an unknown example to NE distribution. Feinman *et al.* (Feinman et al. 2017) proposed two features: Kernel Density (KD) and Bayesian-Uncertainty (BU) to evaluate the proximity of an example to the NE manifold. Ma *et al.* pointed out that KD and BU have limitations of characterizing local adversarial regions. As a remedy, they proposed the feature LID (Ma et al. 2018) to describe the local adversarial regions based on the assumption that AEs are surrounded by several natural data subspaces. Alternatively, Lee *et al.* used the MD to evaluate the dissimilarity between AEs and NEs (Lee et al. 2018). Furthermore, Xu *et. al* (Xu, Evans,

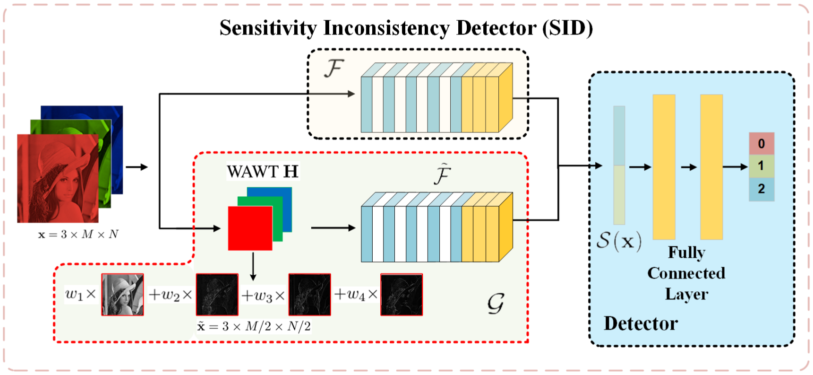

Sensitivity Inconsistency Detector (SID)
$\mathcal{F}$
WAWT $\mathbf{H}$
$\hat{\mathcal{F}}$
$\mathbf{x} = 3 \times M \times N$
$w_1\times$
$+w_2\times$
$+w_3\times$
$+w_4\times$
$\hat{\mathbf{x}} = 3 \times M/2 \times N/2$
$\mathcal{G}$
$\mathcal{S}(\mathbf{x})$
0
1
2
Fully Connected Layer
Detector

Figure 2: The schematic diagram of the proposed SID.

and Qi 2018) devised a detection method via FS, such as reducing the color depth of images or using smoothing. A similar strategy by using the non-linear dimension reducing technique was given by Crecchi *et. al* (Crecchi, Bacciu, and Biggio 2019).

Sensitivity Inconsistency and Dual Classifier Design

Fawzi *et al.* pointed out the adversarial direction usually points to the highly-curved region of the decision boundary, and many examples share the same potential adversarial directions (Fawzi et al. 2018; Moosavi-Dezfooli et al. 2017). This results in a phenomenon that AEs are more likely concentrated in some common highly-curved regions. Imagine that we can fluctuate the highly-curved region of the decision boundary. Then AEs would be very sensitive to such fluctuations in the sense of producing different classification results. In contrast, NEs would be much less sensitive and generate quite consistent results under these fluctuations (Fawzi et al. 2018). A simple yet illustrative example is given in Fig. 1, where the black line represents the decision boundary of a binary classifier and the red line shows the fluctuated (transformed) decision boundary. Here, the decision boundary transformation is conducted by flattening the highly-curved region, while keeping the remaining unchanged. In this case, under the transformed decision boundary, the classification results of AEs would change, while NEs exhibit quite consistent behaviour. This motivates us to exploit the distinct behaviour of AEs and NEs against the decision boundary transformation to tell them apart.

In the upcoming two subsections, we first formulate an optimization problem for finding an expected decision boundary transformation or equivalently the dual classifier. Then, we provide theoretical proofs and empirical justifications to demonstrate the effectiveness of the designed dual classifier.

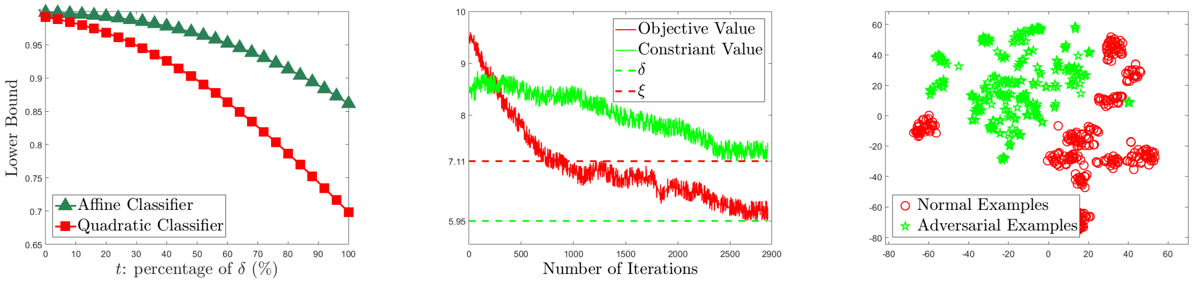
Design Dual Classifier in Transform Domain

For the primal classifier $\mathcal{F}$ trained on the dataset $\mathbf{X}$, let $\mathcal{B}_{i,j}$ be the decision boundary between class i and class j. Let also $\tilde{\mathcal{B}}_{i,j}$ be a transformed version of $\mathcal{B}_{i,j}$ corresponding to the dual classifier $\mathcal{G}$. We expect that $\mathcal{B}_{i,j}$ is similar to $\tilde{\mathcal{B}}_{i,j}$ except for those highly-curved parts. Apparently, the shape of $\mathcal{B}_{i,j}$ highly correlates with the prediction confidence of $\mathcal{F}$ on the training samples. Therefore, the problem of ensuring $\tilde{\mathcal{B}}_{i,j}$ and $\mathcal{B}_{i,j}$ to be similar except for highly-curved parts reduces to make $\mathcal{F}$ and $\mathcal{G}$ generate similar prediction confidences on NEs, but vastly different ones on AEs. Considering all the boundaries $\mathcal{B}_{i,j}$'s, we formulate the following optimization problem to search for $\mathcal{G}$:

$$\begin{aligned} &\min_{\mathcal{G}} \max_{\mathbf{x}\in\mathbf{X}} \|\mathcal{F}(\mathbf{x}) - \mathcal{G}(\mathbf{x})\|_2^2, \\ &\text{s.t. } \|\mathcal{F}(\hat{\mathbf{x}}) - \mathcal{G}(\hat{\mathbf{x}})\|_2^2 \geq \xi, \quad \forall \hat{\mathbf{x}} \in \hat{\mathbf{X}} \end{aligned} \tag{2}$$

where the cost function is designed to minimize the worst-case deviation of $\mathcal{G}$ from $\mathcal{F}$ on NEs, and the constraint guarantees their prediction confidences on AEs to be significant enough.

A straightforward way for solving this problem is to train the classifier $\mathcal{G}$ by integrating the constraint into the loss term. However, such naive solution easily causes overfitting (Raghunathan et al. 2019; Cai et al. 2018). In this work, we resort to a transform-domain technique for getting an approximated solution of (2). An intuition why we search for $\mathcal{G}$ in the transform domain is because DNN classifiers trained in the same domain tend to produce similar decision boundaries including the highly-curved parts (Charles, Rosenberg, and Papailiopoulos 2019; Fawzi, Fawzi, and Fawzi 2018; Tramèr et al. 2018). A crucial problem now is how to appropriately choose the domain transform. Fawzi *et. al* found that the decision boundary near NEs is flat along most directions, with merely few directions being highly-curved (Fawzi et al. 2018). To maintain these flat parts, so as to preserve the results on NEs, we firstly could use linear transforms. Secondly, we should simultaneously flatten the highly-curved parts, which could be achieved by introducing distortions. The theory of manifold learning indicates that the distortion of the geometrical structure is caused by the dimensionality difference between the ambient and intrinsic spaces (Erba, Gherardi, and Rotondo 2019; Tian et al. 2017). Owing to these two reasons, we propose to design a linear, dimension-reducing transform, based on which we search for the desired $\mathcal{G}$. To this end, we define the transformation WAWT



(a)

(b)

(c)

Figure 3: Illustrations of the approximation precision of the constructed dual classifier $\mathcal{G}$, and the separability of the feature of sensitivity inconsistency. (a) Lower bounds in (5) and (6), (b) Objective and constraint values of (2) during the training process, and (c) the separable AEs and NEs.

below. Let $\mathbf{x} \in \mathbf{X}$, and its WAWT transformed version $\tilde{\mathbf{x}}$ can be expressed as:

$$\tilde{\mathbf{x}} \triangleq \mathbf{H}\mathbf{x} = w_1 \mathbf{x}_{ll} + w_2 \mathbf{x}_{lh} + w_3 \mathbf{x}_{hl} + w_4 \mathbf{x}_{hh}, \qquad (3)$$

where $\mathbf{x}_{ll}$, $\mathbf{x}_{lh}$,$\mathbf{x}_{hl}$, and $\mathbf{x}_{hh}$ represent the four sub-bands of a wavelet transform. Here, w_i's are used to balance the importance of different sub-bands. Clearly, due to the lower dimension of each sub-band, $\mathbf{H}$ is a linear, dimension-reducing transform, as we expect. More details on WAWT can be found in the supplementary material.

Now, we can construct a classifier $\tilde{\mathcal{F}}(\tilde{\mathbf{x}})$ in the WAWT domain, and then the dual classifier can be naturally designed as $\mathcal{G}(\mathbf{x}) = \tilde{\mathcal{F}}(\mathbf{H}\mathbf{x})$. Training the dual classifier $\mathcal{G}$ is rather straightforward, by noticing that $\mathcal{G}$ is the composition of $\mathbf{H}$ with $\tilde{\mathcal{F}}$. As shown in Fig. 2 (the part enclosed by the red dashed line), $\mathcal{G}$ could be readily implemented by a DNN composed of the concatenation of $\tilde{\mathcal{F}}$ and WAWT layers with four trainable w_i's. To guarantee that $\mathcal{G}$ and $\mathcal{F}$ have similar classification performance on NEs, the network structure of $\tilde{\mathcal{F}}$ is exactly the same as $\mathcal{F}$, except the input dimension. Note that the primal and the dual classifiers are trained separately.

Before diving into the design of the AE detector, let us first provide some theoretical analyses on the constructed dual classifier.

Theoretical Justifications of Dual Classifier

We aim to demonstrate: **if the predication confidence of the dual classifier $\mathcal{G}$ constructed based on the WAWT $\mathbf{H}$ is consistent with that of the primal classifier $\mathcal{F}$ on NEs, then $\mathcal{G}$ can approximate the optimal solution of the problem (2) with a given ξ.** We first restrict our theoretical analysis to affine and quadratic classifiers, as most of existing theoretical works assumed (Fawzi, Moosavi-Dezfooli, and Frossard 2016; Fawzi et al. 2018). We then offer empirical justifications for the general DNN classifiers. To make the analysis tractable, we assume that the dataset $\mathbf{X}$ is composed by K different classes of samples $\mathbf{X}_k$ $(k = 1, ..., K)$, each of which is drawn from a p-dimensional Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$.

For the affine classifier, we have the following theorem.

Theorem 1 *Let $\mathcal{F}$ be an affine classifier trained over $\mathbf{X}$ and $\mathcal{G}$ be the corresponding dual classifier based on the WAWT $\mathbf{H}$. If*

$$\max_{\mathbf{x}\in\mathbf{X}} \|\mathcal{F}(\mathbf{x}) - \mathcal{G}(\mathbf{x})\|_2^2 \leq \delta, \qquad (4)$$

then $\forall t > 0$ and $\hat{\mathbf{x}} \in \hat{\mathbf{X}}$, we have

$$\mathbb{P}\left\{\|\mathcal{F}(\hat{\mathbf{x}}) - \mathcal{G}(\hat{\mathbf{x}})\|_2^2 \geq t + \delta\right\} \geq \min_{k=1,\ldots,K} Q_{1/2}\left(|\mathbf{C}_k(\frac{1}{2}\mathbf{I} - \eta\boldsymbol{\Sigma}^{-1})\boldsymbol{\mu}_k|, \sqrt{(t+\delta)/\mathbf{C}_k\boldsymbol{\Sigma}_k\mathbf{C}_k^T}\right), \qquad (5)$$

where $Q_{1/2}(\cdot)$ is the Macrcum-Q function (Nuttall 1975), $\boldsymbol{\Sigma}$ is the sample covariance matrix of $\mathbf{X}$, $\mathbf{C}_k = \boldsymbol{\mu}_k^T(\boldsymbol{\Sigma}^{-1} - \mathbf{H}^T(\mathbf{H}\boldsymbol{\Sigma}\mathbf{H}^T)^{-1}\mathbf{H})$, and η is the perturbation magnitude.

Theorem 1 implies that, if the prediction confidence difference of the affine $\mathcal{F}$ and $\mathcal{G}$ is bounded by δ, then the approximation precision of $\mathcal{G}$ to the optimal solution of problem (2) is determined by the probability in (5). To further illustrate the approximation precision, we randomly generate 10-class mixture of Gaussian data $\mathbf{X}$, and set $\eta = 0.5$. Each class contains 100 examples of dimension 256. Fig. 3(a) shows how the lower bound in (5) decays with respect to the increasing t. As can be observed, even when t increases to $60\%\delta$, this probability bound is still larger than 0.95, implying that the constructed $\mathcal{G}$ is a precisely approximated solution of problem (2) with $\xi = 1.6\delta$. In fact, Theorem 1 also reflects the necessity of constructing $\mathcal{G}$ in the linear, dimension-reducing transform domain determined by $\mathbf{H}$. If otherwise $\mathbf{H}$ is a dimension-preserving mapping, then $\mathbf{H}$ is an invertible matrix. In this case, $\mathbf{H}^T(\mathbf{H}\boldsymbol{\Sigma}\mathbf{H}^T)^{-1}\mathbf{H} = \boldsymbol{\Sigma}^{-1}$, making $\mathbf{C}_k = \mathbf{0}, \forall k$. Hence, $\sqrt{(t+\delta)/\mathbf{C}_k\boldsymbol{\Sigma}_k\mathbf{C}_k^T} \to \infty$. Following the property of the Macrcum-Q function (Kapinas, Mihos, and Karagiannidis 2009), the lower bound in (5) vanishes as the second term goes to infinity, regardless t and σ. Besides, the Macrcum-Q function $Q_{1/2}(\cdot)$ in (5) increases as $|\mathbf{C}_k(\frac{1}{2}\mathbf{I} - \eta\boldsymbol{\Sigma}^{-1})\boldsymbol{\mu}_k|$ does, which means that a large η would result in a large lower bound in (5) and thus the designed dual classifier $\mathcal{G}$ is more precise.

For the case of quadratic classifier, we can similarly have Theorem 2.

Theorem 2 *Let $\mathcal{F}$ be a quadratic classifier and $\mathcal{G}$ be the corresponding dual classifier constructed by $\mathbf{H}$. Both $\mathcal{F}$ and $\mathcal{G}$ are trained over $\mathbf{X}$. If $\mathcal{F}$ and $\mathcal{G}$ satisfy (4), then $\forall t > 0$ and $\hat{\mathbf{x}} \in \hat{\mathbf{X}}$, we have*

$$\begin{aligned}&\mathbb{P}\left\{\|\mathcal{F}(\hat{\mathbf{x}}) - \mathcal{G}(\hat{\mathbf{x}})\|_2^2 \geq t + \delta\right\} \geq \\ &\min_{k=1,\ldots,K} \left\{1 - P(\sqrt{\delta + t}; \boldsymbol{\lambda}_k) + P(-\sqrt{\delta + t}; \boldsymbol{\lambda}_k)\right\},\end{aligned} \tag{6}$$

where $\boldsymbol{\lambda}_k$'s are the nonzero eigenvalues of the matrix

$$\hat{\mathbf{\Sigma}}_k^{\frac{1}{2}}(\mathbf{\Sigma}_k^{-1} - \mathbf{H}^T(\mathbf{H}\mathbf{\Sigma}_k\mathbf{H}^T)^{-1}\mathbf{H})\hat{\mathbf{\Sigma}}_k^{\frac{1}{2}} \tag{7}$$

with $\hat{\mathbf{\Sigma}}_k = (\mathbf{I} + \eta\mathbf{\Sigma}_k^{-1})\mathbf{\Sigma}_k(\mathbf{I} + \eta\mathbf{\Sigma}_k^{-1})^T$, $P(\cdot, \boldsymbol{\lambda}_k)$ being the cumulative distribution function of the linear combination of Chi-square random variables with $\boldsymbol{\lambda}_k$ as coefficients (Moschopoulos and Canada 1984), and η being the perturbation magnitude.

A similar curve on the lower bound in (6) can be found in Fig.3(a), as in the affine case. From Theorem 2, we can also see that if $\mathbf{H}$ is invertible as a dimension preserving mapping, then the matrix (7) becomes a zero matrix with $\boldsymbol{\lambda}_k = \mathbf{0}$, $\forall k$. Consequently, $1 - P(\sqrt{\delta + t}; \mathbf{0}) + P(-\sqrt{\delta + t}; \mathbf{0}) = 0$ for any $\delta, t > 0$, resulting in a trivial bound in (6). This again validates the usage of the linear, dimension-reducing $\mathbf{H}$. Moreover, the lower bound increases as the increase of η. It becomes even clearer when each $\mathbf{X}_k$ is sampled from an independent multivariate Gaussian. In this case, the lower bound in (6) reduces to $1 - \mathcal{T}(3p/8, \sqrt{(t + \delta)/4(1 + \eta)^4})$, where the regularized gamma function $\mathcal{T}(\cdot)$ is decreasing as the increase of η (see the Corollary 2 in the supplementary materials). The proofs of Theorems 1 and 2 are also given in the supplementary materials.

In addition to these theoretical proofs, we give the empirical justifications on the effectiveness of the constructed $\mathcal{G}$, for general DNN classifier with complex decision boundaries. As an example, we choose the VGG classifier on CIFAR10 as the primal classifier $\mathcal{F}$, and then separately train the dual classifier $\mathcal{G}$. During the training process of $\mathcal{G}$, we calculate the values of the objective function in problem (2) on 500 test images, and the values of the constraint on their AEs produced by DeepFool (Moosavi-Dezfooli, Fawzi, and Frossard 2016). Fig. 3(b) shows how the objective function and the constraint vary with the number of iterations of minibatches. As can be observed, $\mathcal{G}$ tends to approximate the optimal solution of the problem (2) with $\xi = 7.11$, in which case the objective function is minimized to 5.95.

Detecting Adversarial Examples from Sensitivity Inconsistency

Upon the construction of the dual classifier $\mathcal{G}$, we now collaboratively use it with the primal classifier $\mathcal{F}$ to design a feature called sensitivity inconsistency. We also give the details on the proposed *Sensitivity Inconsistency Detector* (SID) for discriminating NEs and AEs.

The Feature of Sensitivity Inconsistency

For a given unknown example $\mathbf{x}_0$, a natural measure to evaluate its sensitivity against the decision boundary transformation can be defined as:

$$\mathcal{S}(\mathbf{x}_0) = \left\{f_i(\mathbf{x}_0) - g_i(\mathbf{x}_0)\right\}_{i=1}^{K}, \tag{8}$$

where $f_i(\mathbf{x}_0)$ and $g_i(\mathbf{x}_0)$ are the associated prediction conferences under $\mathcal{F}$ and $\mathcal{G}$, respectively. The vectorized $\mathcal{S}(\mathbf{x}_0)$ is then called the feature of sensitivity inconsistency of $\mathbf{x}_0$. The design process of $\mathcal{G}$ naturally endows the sensitivity inconsistency with the power to discriminate NEs and AEs. More precisely, $\forall \mathbf{x} \in \mathbf{X}, \hat{\mathbf{x}} \in \hat{\mathbf{X}}$, if $\mathcal{G}$ is the solution of problem (2), then there exists a gap between $\|\mathcal{F}(\hat{\mathbf{x}}) - \mathcal{G}(\hat{\mathbf{x}})\|_2$ and $\|\mathcal{F}(\mathbf{x}) - \mathcal{G}(\mathbf{x})\|_2$. In other words, the inequality $\|\mathcal{S}(\hat{\mathbf{x}})\|_2^2 - \|\mathcal{S}(\mathbf{x})\|_2^2 \geq \epsilon$ always holds for a certain ϵ. In this case, there would exist a sphere such that $\mathcal{S}(\mathbf{x})$ is located at the inside of it, while $\mathcal{S}(\hat{\mathbf{x}})$ is distributed on the outside. Therefore, $\mathbf{x}$ and $\hat{\mathbf{x}}$ can be well separated.

Algorithm 1: Training procedure of SID

```
Input: $\mathcal{F}$: primal classifier; $\mathcal{G}$: dual classifier; $\mathbf{X}_c$: NEs
       correctly classified by $\mathcal{F}$; $\mathcal{A}$: attack method; $E$:
       maximal training epoch.
Output: $\mathcal{D}$: trained SID.
while $t < E$ do
    for $\mathbf{B}_c$ in $\mathbf{X}_c$ do
        $\mathbf{B}_a :=$ attack $\mathbf{B}_c$ with $\mathcal{A}$;
        $\mathbf{B}_n :=$ add random noise to $\mathbf{B}_c$;
          ▷ $\mathbf{B}_c$: a minibatch of $\mathbf{X}_c$.
        $\mathbf{D}_1 := \{\mathbf{x} \in \{\mathbf{B}_n, \mathbf{B}_c\} | k_{\mathcal{F}}(\mathbf{x}) = k_{\mathcal{G}}(\mathbf{x})\}$;
        $\mathbf{D}_2 := \{\mathbf{x} \in \{\mathbf{B}_n, \mathbf{B}_c\} | k_{\mathcal{F}}(\mathbf{x}) \neq k_{\mathcal{G}}(\mathbf{x})\}$;
          ▷ $k_{\mathcal{F}}(\mathbf{x}), k_{\mathcal{G}}(\mathbf{x})$: predicted labels of $\mathcal{F}$ and $\mathcal{G}$.
        $\mathbf{D} := \{\mathbf{D}_1, \mathbf{D}_2, \mathbf{B}_a\}$;
        $\mathcal{D}^t = \min_{\mathcal{D}^t} \frac{1}{|\mathbf{D}|} \sum_{\mathbf{x} \in \mathbf{D}} \mathcal{L}\left(\mathcal{D}^t(\mathcal{S}(\mathbf{x})), y\right)$;
          ▷ $\mathcal{L}(\cdot)$: cross entropy. ▷ $\mathcal{S}(\mathbf{x})$ : calculate as (8).
          ▷ $\mathcal{D}^t(\mathcal{S}(\mathbf{x})) = \{d_0^t(\mathbf{x}), d_1^t(\mathbf{x}), d_2^t(\mathbf{x})\}$.
          ▷ $d_i^t(\mathbf{x})$: the predicted confidences of $\mathcal{D}^t$.
    end
end
```

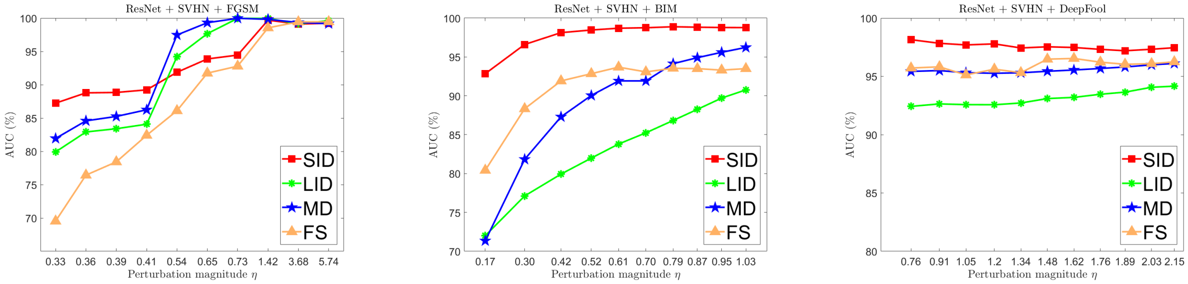
To further show the separability of NEs and AEs based on the feature of sensitivity inconsistency, we give an example in Fig. 3(c), where the considered $\mathcal{F}$ and $\mathcal{G}$ are VGG trained on CIFAR10 and its dual version. We randomly select 300 correctly classified examples from the test set and generate their adversarial counterparts using DeepFool. Then, we use the algorithm T-SNE (Maaten and Hinton 2008) to visualize the feature of sensitivity inconsistency of selected AEs and NEs. As can be observed, the feature vectors corresponding to them are largely separable. This example further motivates us to use the feature of sensitivity inconsistency for distinguishing AEs from NEs.

The Design of SID

We now give the details of SID, by virtue of the developed feature of sensitivity inconsistency. The schematic diagram of SID is depicted in Fig. 2, which consists of three parts: the pre-trained primal classifier $\mathcal{F}$, the dual classifier $\mathcal{G}$, and a detector trained on features of sensitivity inconsistency. For a given $\mathcal{F}$, the dual classifier $\mathcal{G}$ is the composition of a WAWT layer and a DNN classifier with the same structure as $\mathcal{F}$. The output prediction confidence of $\mathcal{F}$ and $\mathcal{G}$ is

Detector	Source	DeepFool	FGSM	BIM	C&W	Source	DeepFool	FGSM	BIM	C&W
SID	ResNet & CIFAR10	**97.08**	94.79	**99.38**	**96.56**	VGG & CIFAR10	**98.32**	84.92	**96.55**	**95.61**
MD		93.72	**99.89**	92.73	94.22		93.84	**99.76**	89.13	78.84
LID		93.49	96.54	80.52	81.21		90.02	96.85	80.04	73.68
FS		84.63	78.66	89.06	85.51		90.78	78.28	79.94	76.34
SID	ResNet & SVHN	**98.15**	93.64	**98.78**	**97.04**	VGG & SVHN	**98.78**	94.41	**99.83**	**95.55**
MD		95.43	**98.62**	96.23	91.59		87.31	**99.73**	90.79	80.71
LID		92.43	96.77	90.77	87.67		91.51	98.32	92.39	83.07
FS		95.72	90.07	93.66	91.55		93.99	82.97	95.43	87.52

Table 1: Comparison of AUC scores (%) of detecting AEs under different settings.



(a)

(b)

(c)

Figure 4: Detection performance of SID, LID, MD, and FS w.r.t the perturbation magnitude η. Attacking (a) ResNet on SVHN with FGSM, (b) ResNet on SVHN with BIM, and (c) ResNet on SVHN with DeepFool.

then passed into the detector which is composed of two fully connected layers. To improve the discrimination capability of SID, we design the detector as a three-class classifier; in addition to AEs (label 0) and NEs (label 1), those examples which are inconsistently classified by $\mathcal{F}$ and $\mathcal{G}$ are categorized into the third class (label 2).

The details of training the SID is given in Algorithm 1, which starts with the generation of adversarial and noisy counterparts to NEs. Similar to prior studies (Ma et al. 2018; Lee et al. 2018), these NEs ($\mathbf{X}_c$) are assumed to be correctly classified by $\mathcal{F}$. For one mini-batch of NEs ($\mathbf{B}_c$), the corresponding AE mini-batch ($\mathbf{B}_a$) is generated using an attack strategy $\mathcal{A}$ (line 3). The noisy mini-batch ($\mathbf{B}_n$) can be readily produced by adding random noises with the same noise magnitude as the adversarial perturbation added on $\mathbf{B}_a$ (line 4). Examples in $\mathbf{B}_c$ and $\mathbf{B}_n$ are further split into $\mathbf{D}_1$ which contains the examples consistently classified by $\mathcal{F}$ and $\mathcal{G}$, and $\mathbf{D}_2$, which corresponds to the inconsistently classified ones (line 5-6). We now have prepared a batch of examples $\mathbf{D}$ composed by three classes of examples, $\mathbf{B}_a$, $\mathbf{D}_1$, and $\mathbf{D}_2$ whose labels are 0, 1, and 2 respectively (line 7). Upon having $\mathbf{D}$, we can calculate the sensitivity inconsistency $\mathcal{S}(\mathbf{x})$ of each $\mathbf{x} \in \mathbf{D}$ via (8), and minimize the average cross entropy of SID on $\mathbf{D}$ (line 8). After exhausting all mini-batches in $\mathbf{X}_c$, and repeating the above process for E times, we finally obtain the trained SID.

At the testing stage, classes 1 and 2 are merged. That is, a given example $\mathbf{x}_0$ is identified as an AE if the predicted label from SID is 0; otherwise, it is treated as a NE.

Experimental Results

We evaluate the performance of the proposed SID on detecting AEs. We compare SID with the state-of-the-art schemes based on LID (Ma et al. 2018), MD (Lee et al. 2018), and FS (Xu, Evans, and Qi 2018). In addition, we discuss the robustness of SID under a white box attack. We consider two network structures VGG19 with batch normalization (Simonyan and Zisserman 2014) and ResNet34 (He et al. 2016) on two datasets CIFAR10 (Krizhevsky and Hinton 2009) and SVHN (Netzer et al. 2011). The metric for evaluating the performance of different detectors is the widely-used AUC score. For fair comparison, all the parameters in LID, MD, and FS are fine tuned to achieve the best performance. In the WAWT, the wavelet transform $\mathbf{sym17}$ is adopted (Lee et al. 2019).

Detection Performance Comparison and Analysis

We first show the performance comparison when AEs for training and testing are all generated by the same attack. In Table 1, we compare the AUC scores of different detectors, where the attack methods DeepFool (Moosavi-Dezfooli, Fawzi, and Frossard 2016), FGSM (Goodfellow, Shlens, and Szegedy 2015), BIM (Kurakin, Goodfellow, and Bengio 2017), and C&W (Carlini and Wagner 2017) are considered. The perturbation magnitudes η of all the AEs can be found in the supplementary file. Compared with LID, MD, and FS, our proposed SID outperforms them in most cases, especially when the attack methods are more sophisticated, *e.g.*, BIM, DeepFool, and C&W.

Model-Dataset	Source	Target (SID/ MD / LID)			
		DeepFool	FGSM	BIM	C&W
ResNet & SVHN	DeepFool	**98.15**/95.32/87.26	**94.87**/87.58/77.65	**95.73**/90.09/76.75	**96.16**/91.56/73.43
	FGSM	92.47 /**94.34**/84.26	**91.33**/85.34/78.43	**96.93**/89.55/73.03	**94.73**/90.41/80.41
	BIM	**95.62**/94.65/90.87	**96.42**/86.72/76.86	**99.47**/89.69/83.31	**98.47**/91.13/86.26
	C&W	**97.07**/95.02/82.05	**96.06**/87.54/78.57	**98.06**/89.67/74.52	**97.87**/91.52/77.68
ResNet & CIFAR10	DeepFool	**95.99**/92.63/80.94	**95.23**/85.42/83.61	**92.99**/86.89/77.52	**93.98**/91.14/82.79
	FGSM	**92.66**/91.43/83.21	**93.64**/83.21/76.54	**90.31**/82.31/75.43	**93.27**/89.43/81.34
	BIM	**94.26**/90.99/83.21	**95.27**/84.35/79.31	**99.68**/87.01/79.39	**99.31**/92.65/83.04
	C&W	**96.02**/90.91/82.51	**96.37**/84.32/79.13	**95.54**/87.25/78.53	**97.58**/93.78/87.53

Table 2: Comparison of the generalizability (AUC scores).

Source	RseNet-CIFAR10		RseNet-SVHN	
	A_O	A_W	A_O	A_W
BIM-L	99.81	98.31	98.51	98.13
BIM-S	97.17	96.31	95.65	95.13
FGSM-L	96.13	93.47	98.37	96.53
FGSM-S	87.93	87.74	83.88	82.35

Table 3: Robustness of SID against the white box attack.

To more thoroughly show the advantages of SID, we give the AUC scores of different detectors on AEs generated by FGSM, BIM and DeepFool, with respect to the varying η. As can be seen from Fig. 4, SID achieves better AUC performance than all the competing detectors on AEs generated by BIM and DeepFool, for all η's. In the cases of AEs produced by FGSM, our SID is still the best when $\eta \leq 0.41$, and gradually becomes inferior to LID and MD when η further increases. Arguably, when η is large, FGSM could be deemed as unsuccessful, as the incurred distortions would be too large. In fact, when η is significant enough (*e.g.*, $\eta > 3.68$), the AUC performance of all detectors is almost perfect in the case of FGSM. This observation, in turn, reflects that the detection challenges mainly lie in the regime of small η, in which SID is the best detector in all the tested cases.

We also compare the generalizability of different detectors in Table 2. Note that here FS is not considered, as the training of FS merely relies on NEs. It can be found that the generalizability of SID is much improved in most cases, compared with MD and LID. Such desirable generalizability is attributed to the fact that AEs in the same curved regions would have similar sensitivity to the boundary transformation. For those sophisticatedly designed attacks such as DeepFool, BIM, and C&W, the generated AEs can be regarded as good approximations to the optimal ones (in the sense of (1)). This implies that these AEs from different sources would be concentrated in the same curved regions as the underlying optimal ones, and hence, would exhibit similar sensitivity inconsistency. Hence, naturally, SID trained on one source is very likely to be able to detect AEs from other sources.

Robustness of SID under white box attack

We now evaluate the robustness of SID under the white box attack. In this scenario, an attacker knows everything including model parameters, training dataset, details on SID, etc. The purpose is to generate AEs to mislead the target classifier and simultaneously fool SID. A widely-adopted white box attack strategy is to maximize the cost function $\mathcal{L}$ of the classification model and loss l of the detector at the same time (Lee et al. 2018; Ma et al. 2018). Namely,

$$\max_{\mathbf{r}} \mathcal{L}(\hat{\mathbf{x}}, y_c) + \alpha \cdot l(\hat{\mathbf{x}}, y_d), \quad \text{s.t.} \quad \|\mathbf{r}\|_2 < \eta, \tag{9}$$

where y_c is the ground-truth label of the NE $\mathbf{x}$, $y_d = 0$ indicates that $\hat{\mathbf{x}} = \mathbf{x} + \mathbf{r}$ is an AE, η is the maximal allowable adversarial perturbation magnitude, and α is a tradeoff parameter.

To evaluate the robustness of SID, we randomly select 1000 NEs, which are correctly classified by ResNet, from CIFAR10 and SVHN respectively. We then generate their adversarial counterparts according to the white box attack strategy (9). The performance degradation of SID to the white box attack is examined in Table 3. The first 2 rows give the results for SID trained on AEs produced by BIM with two different settings of η (BIM-L and BIM-S for large and small η's). The remaining two rows report the results for the cases of FGSM (FGSM-L and FGSM-S for large and small η's). Detailed settings of η can be found in the supplementary file. Here, A_O and A_W represent the AUC scores of detecting the original AEs and the ones generated by the white box attack. For instance, the SID trained on BIM-L source achieves 99.81% AUC score on detecting original AEs targeted on ResNet-CIFAR10, while still obtaining 98.31% on detecting AEs produced by the white box attack. As the AUC scores for detecting original AEs and the ones from (9) are close, we claim that SID shows promising robustness against the white box attack.

Conclusions

In this paper, we have proposed a simple yet effective method for detecting AEs, via the sensitivity inconsistency between NEs and AEs to the decision boundary fluctuations. Experimental results have been provided to show the superiority of the proposed SID detector, compared with the state-of-the-art algorithms based on LID, MD, and FS.

Acknowledgments

This work was supported by Macau Science and Technology Development Fund under SKL-IOTSC-2018-2020, 077/2018/A2, 0015/2019/AKP, and 0060/2019/A1, by Research Committee at University of Macau under MYRG2018-00029-FST and MYRG2019-00023-FST, by Natural Science Foundation of China under 61971476 and 62001304, and by Guangdong Basic and Applied Basic Research Foundation under 2019A1515110410.

References

Aigrain, J.; and Detyniecki, M. 2019. Detecting Adversarial Examples and Other Misclassifications in Neural Networks by Introspection. In *International Conference on Machine Learning*, 1–10.

Akhtar, N.; Liu, J.; and Mian, A. 2018. Defense Against Universal Adversarial Perturbations. In *IEEE Conference on Computer Vision and Pattern Recognition*, 3389–3398.

Brendel, W.; Rauber, J.; and Bethge, M. 2018. Decision based Adversarial Attacks: Reliable Attacks against Black-Box Machine Learning Models. In *International Conference on Learning Representations*, 1–12.

Brown, T.; Mane, D.; Roy, A.; Abadi, M.; and Gilmer, J. 2017. Adversarial Patch. ArXiv preprint arXiv:1712.09665.

Cai, Q.-Z.; Du, M.; Liu, C.; and Song, D. 2018. Curriculum Adversarial Training. ArXiv preprint arXiv:1805.04807.

Carlini, N.; and Wagner, D. 2017. Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods. In *ACM Workshop on Artificial Intelligence and Security*, 3–14.

Carrara, F.; Falchi, F.; Caldelli, R.; Amato, G.; Fumarola, R.; and Becarelli, R. 2017. Detecting Adversarial Example Attacks to Deep Neural Networks. In *International Workshop on Content-Based Multimedia Indexing*, 1–7.

Charles, Z.; Rosenberg, H.; and Papailiopoulos, D. 2019. A Geometric Perspective on the Transferability of Adversarial Directions. In *International Conference on Artificial Intelligence and Statistics*, 1960–1968.

Crecchi, F.; Bacciu, D.; and Biggio, B. 2019. Detecting Adversarial Examples through Nonlinear Dimensionality Reduction. In *European Symposium on Artificial Neural Networks*, 582–597.

Erba, V.; Gherardi, M.; and Rotondo, P. 2019. Intrinsic Dimension Estimation for Locally Undersampled Data. *Scientific reports* 9(1): 1–9.

Eykholt, K.; Evtimov, I.; Fernandes, E.; Li, B.; Rahmati, A.; Xiao, C.; Prakash, A.; Kohno, T.; and Song, D. 2018. Robust Physical-World Attacks on Deep Learning Visual Classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1–10.

Fawzi, A.; Fawzi, H.; and Fawzi, O. 2018. Adversarial Vulnerability for Any Classifier. In *Advances in Neural Information Processing Systems*, 1178–1187.

Fawzi, A.; Moosavi-Dezfooli, S.-M.; and Frossard, P. 2016. Robustness of Classifiers: from Adversarial to Random Noise. In *Advances in Neural Information Processing Systems*, 1632–1640.

Fawzi, A.; Moosavi-Dezfooli, S.-M.; Frossard, P.; and Soatto, S. 2018. Empirical Study of the Topology and Geometry of Deep Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 3762–3770.

Feinman, R.; Curtin, R. R.; Shintre, S.; and Gardner, A. B. 2017. Detecting Adversarial Samples from Artifacts. ArXiv preprint arXiv:1703.00410.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and Harnessing Adversarial Examples. *International Conference on Learning Representations* 1–11.

Grosse, K.; Manoharan, P.; Papernot, N.; Backes, M.; and McDaniel, P. 2017. On the (Statistical) Detection of Adversarial Examples. ArXiv preprint arXiv:1702.06280.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Kapinas, V. M.; Mihos, S. K.; and Karagiannidis, G. K. 2009. On The Monotonicity of the Generalized Marcum and Nuttall Q-Functions. *IEEE Transactions on Information Theory* 55(8): 3701–3710.

Karpathy, A.; Toderici, G.; Shetty, S.; Leung, T.; Sukthankar, R.; and Fei-Fei, L. 2014. Large-Scale Video Classification with Convolutional Neural Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1725–1732.

Krizhevsky, A.; and Hinton, G. 2009. Learning Multiple Layers of Features from Tiny Images. Technical report, Citeseer.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, 1097–1105.

Kurakin, A.; Goodfellow, I.; and Bengio, S. 2017. Adversarial Examples in the Physical World. *International Conference on Learning Representations (Workshop)* 1–10.

Lee, G.; Gommers, R.; Waselewski, F.; Wohlfahrt, K.; and O'Leary, A. 2019. Pywavelets: A Python Package for Wavelet Analysis. *Journal of Open Source Software* 4(36): 1237–1238.

Lee, K.; Lee, K.; Lee, H.; and Shin, J. 2018. A Simple Unified Framework for Detecting Out-Of-Distribution Samples and Adversarial Attacks. In *Advances in Neural Information Processing Systems*, 7167–7177.

Li, X.; and Li, F. 2017. Adversarial Examples Detection in Deep Networks with Convolutional Filter Statistics. In *International Conference on Computer Vision*, 5764–5772.

Liao, F.; Liang, M.; Dong, Y.; Pang, T.; Hu, X.; and Zhu, J. 2018. Defense against Adversarial Attacks Using High-Level Representation Guided Denoiser. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1778–1787.

Liu, J.; Zhang, W.; Zhang, Y.; Hou, D.; Liu, Y.; Zha, H.; and Yu, N. 2019. Detection Based Defense against Adversarial Examples from the Steganalysis Point of View. In *IEEE Conference on computer vision and pattern recognition*, 4825–4834.

Liu, Y.; Chen, X.; Liu, C.; and Song, D. 2016. Delving Into Transferable Adversarial Examples and Black-Box Attacks. ArXiv preprint arXiv:1611.02770.

Ma, X.; Li, B.; Wang, Y.; Erfani, S. M.; Wijewickrema, S.; Schoenebeck, G.; Song, D.; Houle, M. E.; and Bailey, J. 2018. Characterizing Adversarial Subspaces Using Local Intrinsic Dimensionality. In *International Conference on Learning Representations*, 1–9.

Maaten, L. v. d.; and Hinton, G. 2008. Visualizing Data Using T-SNE. *Journal of Machine Learning Research* 9(11): 2579–2605.

Metzen, J. H.; Genewein, T.; Fischer, V.; and Bischoff, B. 2017. On Detecting Adversarial Perturbations. *International Conference on Learning Representations* 1–12.

Moosavi-Dezfooli, S.-M.; Fawzi, A.; Fawzi, O.; Frossard, P.; and Soatto, S. 2017. Analysis of Universal Adversarial Perturbations. ArXiv preprint arXiv:1705.09554.

Moosavi-Dezfooli, S.-M.; Fawzi, A.; and Frossard, P. 2016. Deepfool: A Simple and Accurate Method to Fool Deep Neural Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2574–2582.

Moschopoulos, P.; and Canada, W. 1984. The Distribution Function of a Linear Combination of Chi-Squares. *Computers & mathematics with applications* 10(4-5): 383–386.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; and Ng, A. Y. 2011. Reading Digits in Natural Images with Unsupervised Feature Learning. In *Advances in Neural Information Processing Systems*, 1–10.

Nuttall, A. 1975. Some Integrals Involving the Q_M Function. *IEEE Transactions on Information Theory* 21(1): 95–96.

Oberdiek, P.; Rottmann, M.; and Gottschalk, H. 2018. Classification Uncertainty of Deep Neural Networks based on Gradient Information. In *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*, 113–125. Springer.

Papernot, N.; McDaniel, P.; Jha, S.; Fredrikson, M.; Celik, Z. B.; and Swami, A. 2016a. The Limitations of Deep Learning in Adversarial Settings. In *Proceedings of IEEE European Symposium on Security and Privacy*, 372–387.

Papernot, N.; McDaniel, P.; Wu, X.; Jha, S.; and Swami, A. 2016b. Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks. In *IEEE Symposium on Security and Privacy*, 582–597.

Raghunathan, A.; Xie, S. M.; Yang, F.; Duchi, J. C.; and Liang, P. 2019. Adversarial Training Can Hurt Generalization. ArXiv preprint arXiv:1906.06032.

Schmidt, L.; Santurkar, S.; Tsipras, D.; Talwar, K.; and Madry, A. 2018. Adversarially Robust Generalization Requires More Data. In *Advances in Neural Information Processing Systems*, 5014–5026.

Sharif, M.; Bauer, L.; and Reiter, M. K. 2018. On the Suitability of Lp-Norms for Creating and Preventing Adversarial Examples. In *IEEE Conference on Computer Vision and Pattern Recognition (Workshope)*, 1–10.

Simonyan, K.; and Zisserman, A. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *International Conference on Learning Representations* 1–9.

Smith, L.; and Gal, Y. 2018. Understanding Measures of Uncertainty for Adversarial Example Detection. ArXiv preprint arXiv:1803.08533.

Song, Y.; Kim, T.; Nowozin, S.; Ermon, S.; and Kushman, N. 2018. Pixeldefend: Leveraging Generative Models to Understand and Defend Against Adversarial Examples. In *International Conference on Learning Representations*, 1–9.

Sperl, P.; Kao, C.-Y.; Chen, P.; and Böttinger, K. 2019. DLA: Dense-Layer-Analysis for Adversarial Example Detection. In *European Symposium on Security and Privacy*, 1–15.

Sun, G.; Su, Y.; Qin, C.; Xu, W.; Lu, X.; and Ceglowski, A. 2020. Complete Defense Framework to Protect Deep Neural Networks against Adversarial Examples. *Mathematical Problems in Engineering* 2020: 1–15.

Tian, J.; Zhang, T.; Qin, A.; Shang, Z.; and Tang, Y. Y. 2017. Learning the Distribution Preserving Semantic Subspace for Clustering. *IEEE Transactions on Image Processing* 26(12): 5950–5965.

Tramèr, F.; Papernot, N.; Goodfellow, I.; Boneh, D.; and McDaniel, P. 2017. The Space of Transferable Adversarial Examples. ArXiv preprint arXiv:1704.03453.

Tramèr, F.; Papernot, N.; Goodfellow, I.; Boneh, D.; and McDaniel, P. 2018. The Space of Transferable Adversarial Examples. ArXiv preprint arXiv:1802.00420.

Tsipras, D.; Santurkar, S.; Engstrom, L.; Turner, A.; and Madry, A. 2019. Robustness May Be at Odds with Accuracy. In *International Conference on Learning Representations*, 1–9.

Xu, W.; Evans, D.; and Qi, Y. 2018. Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks. In *Network and Distributed System Security Symposium*, 1–15.