Distinguishability of Adversarial Examples

Yi Qin

Colorado School of Mines
Golden, Colorado, USA
yiqin@mines.edu

Ryan Hunt

Colorado School of Mines
Golden, Colorado, USA
ryhunt@mines.edu

Chuan Yue

Colorado School of Mines
Golden, Colorado, USA
chuan Yue@mines.edu

ABSTRACT

Machine learning models can be easily fooled by adversarial examples which are generated from clean examples with small perturbations. This poses a critical challenge to machine learning security, and impedes the wide application of machine learning in many important domains such as computer vision and malware detection. From a unique angle, we propose to investigate two important research questions in this paper: Are adversarial examples generated by different methods distinguishable from clean examples? Are adversarial examples generated by different methods distinguishable from each other? These two questions concern the *distinguishability* of adversarial examples. Answering them will potentially lead to a simple yet effective approach, termed as *defensive distinction* in this paper under the formulation of multi-label classification, for protecting against adversarial examples. We design and perform experiments using the MNIST and CIFAR-10 datasets to investigate these two questions, and obtain positive results demonstrating the strong distinguishability of adversarial examples.

KEYWORDS

Adversarial Examples, Adversarial Machine Learning, Neural Networks, Deep Learning, Defense

ACM Reference Format:

Yi Qin, Ryan Hunt, and Chuan Yue. 2020. Distinguishability of Adversarial Examples. In *The 15th International Conference on Availability, Reliability and Security (ARES 2020)*, August 25–28, 2020, Virtual Event, Ireland. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3407023.3407040>

1 INTRODUCTION

Machine learning models including SVMs [1] and especially deep neural networks [29] can be easily fooled by adversarial examples which are generated from the clean examples with small perturbations. Quite often, both machine learning models and humans can classify the clean examples such as the images of pandas with high accuracy, and humans can still classify the adversarial examples as pandas with high accuracy because the small perturbations are imperceptible; however, machine learning models are fooled to misclassify adversarial examples as some targets such as gibbons [7] desired by attackers.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ARES 2020, August 25–28, 2020, Virtual Event, Ireland

© 2020 Association for Computing Machinery.

ACM ISBN 978-1-4503-8833-7/20/08...\$15.00

<https://doi.org/10.1145/3407023.3407040>

This intriguing property or vulnerability of machine learning models poses a critical challenge to machine learning security, and it impedes the wide application of machine learning in many important domains such as computer vision (e.g., for self driving cars), and even in malware detection [1, 32] and anomaly detection [10]. Furthermore, the discovery of new and powerful adversarial example generation methods such as [3–5, 7, 11, 18, 21, 24, 28–30] goes on without cessation, indicating to a certain extent the unlimited capabilities for attackers to continuously and easily fool machine learning models. On the other hand, even state-of-the-art defense approaches such as adversarial training [7, 29] and defensive distillation [25] still suffer from major limitations and can be circumvented (Section 2). Researchers also experimented with different adversarial example detection mechanisms [8, 15–17, 19, 20, 31], and identified some limitations such as their weak capability on detecting examples generated by advanced white-box attacks [2]. Therefore, the unfortunate status quo is that attackers prevail over defenders. The defense by detection approach is intuitive and promising, but a deeper understanding of the strengths and weaknesses of this approach is needed for improving its effectiveness.

In this paper, from a unique angle, we propose to investigate two important research questions that concern the *distinguishability* of adversarial examples. Question 1: are adversarial examples generated by different methods distinguishable from clean examples? Question 2: are adversarial examples generated by different methods distinguishable from each other? If the answer to Question 1 will be positive, i.e., given a certain classification task such as image classification, generated adversarial examples (regardless of the objects they represent) largely belong to one class while clean examples belong to the other class, then defenders can simply discard those adversarial examples to protect the machine learning models. If the answer to Question 2 will be positive, i.e., adversarial examples generated by different methods clearly belong to different classes, defenders can better protect the machine learning models, for example, by incorporating the corresponding examples into the adversarial training process to enhance the robustness of the models. Besides such practical benefits, answering these two questions may also help researchers further identify the nature of adversarial examples.

Formally, we consider a classification problem in adversarial environments as a multi-label classification problem. That is, upon seeing a new input such as an image, while the original task such as classifying the image as a certain object is important, it is also important to classify the image as a generated adversarial vs. a clean example in the first place. We formulate this multi-label classification problem in Section 3 to guide us in answering the two questions, and term the corresponding defense approach as *defensive distinction*, which distinguishes adversarial vs. clean examples

and distinguishes adversarial examples generated by different methods to protect against the attacks.

We design and perform experiments using the MNIST [14] and CIFAR-10 [9] datasets to investigate the two research questions and evaluate the effectiveness of our defensive distinction approach. In our experiments, we consider multiple scenario-case combinations that defenders either know or do not know the neural network, source images, and methods as well as parameters used by attackers for generating adversarial examples. We obtain highly positive answers to both research questions. For example, in some typical cases for MNIST and CIFAR-10, adversarial vs. clean examples can be distinguished perfectly with 100% accuracy, while adversarial examples generated by different methods can be distinguished with over 90% accuracy. Our experimental results demonstrate the strong distinguishability of adversarial examples, and demonstrate the value of the defensive distinction approach.

We make four main contributions in this paper: (1) we propose to investigate two important research questions that concern the distinguishability of adversarial examples; (2) we formulate a classification problem in adversarial environments as a multi-label classification problem to answer the two questions; (3) we propose and explore a unique defense approach termed as defensive distinction; (4) we design and perform experiments to empirically demonstrate the strong distinguishability of adversarial examples and the value of our defensive distinction approach. We recommend that this unique defensive distinction approach should be considered to complement other defense approaches.

The rest of the paper is structured as follows: Section 2 introduces the related work on adversarial example generation methods and existing defense approaches, Section 3 describes our defensive distinction protection approach, Section 4 presents our experiments and results, and Section 5 discusses the limitations of this work and potential future work, and concludes the paper.

2 RELATED WORK

Adversarial examples can easily fool machine learning models, and they can be generated by a number of powerful methods. One representative method is L-BFGS [29], which formulates the adversarial example generation problem as a box-constrained optimization problem and uses line-search to identify approximated optimal solutions. Another representative method is FGSM (Fast Gradient Sign Method) [7], which efficiently obtains an optimal maximum constrained perturbation for generating adversarial examples. FGSM inspired researchers to pursue a few fine-grained refinements over it, resulting in powerful methods such as BIM (Basic Iterative Method) [11], PGD (Projected Gradient Descent) [18], and MIM (Momentum Iterative Method) [5]. L-BFGS also inspired researchers to design more powerful or general methods such as C&W by [3] and EAD (Elastic Net Method) [4]. JSMA (Jacobian-based Saliency Map Approach) [24] is another representative method, which constructs adversarial saliency maps to guide the selection of input features for perturbation in multiple iterations. We refer readers to the original papers and the CleverHans report [22] for the details of these and some other methods.

On the defense side, one representative approach is adversarial training [7, 29], which basically uses both the generated adversarial examples and the clean examples to train the machine learning

models with better regularization. Another representative approach is defensive distillation [25], which basically uses the same neural network architecture to train a distilled network based on the probability vectors produced by the original network, thus reducing the models' sensitivity to small input perturbations. Both approaches in essence focus on improving the generalization and thus robustness of the machine learning models that perform the original classification task. While representing the state-of-the-art, both approaches still suffer from major limitations and can be circumvented [6]. For example, defensive distillation was found to be ineffective against C&W attacks [3] and black-box attacks [23], while adversarial training was found to be ineffective against adversarial examples generated by many iterative methods such as BIM [12], PGD [18], and MIM [5]. Qin et al. [26] found that adversarial training using adversarial examples from different attack methods is beneficial, and adversarial training should be an iterative process.

Another approach of defense is detection [8, 15–17, 19, 20, 31], which aims to determine whether an input to an original model is legitimate vs. adversarial. Magnet [19] performs a denoiser step for each input. Specifically, it trains an encoder and a decoder to remove the added adversarial perturbation. SafetyNet [16] leverages the ReLU neuron patterns and SVM to detect adversarial examples. Wang et al. [31] mutated DNN models to detect adversarial examples based on the observation that adversarial examples are more likely to have changed labels on mutated models. Ma et al. [17] proposed NIC which leverages the provenance channel and the activation value distribution channel inside the DNNs to detect adversarial examples based on the observation that adversarial examples violate either provenance invariant or value invariant. Carlini et al. [2] showed that some detection mechanisms are weak on detecting examples generated by advanced white-box attacks, and recommended that evaluations should be performed on both MNIST and CIFAR-10 datasets due to their different security properties.

We propose to take a multi-label classification approach, defensive distinction, to distinguish clean vs. adversarial examples, and distinguish adversarial examples generated from different methods. This approach can help a defender understand the in-depth behaviors of attackers by exploring the intrinsic correlations between multiple labels such as between clean vs. adversarial and the multiple classes in the original classification task. We consider eight scenario-case combinations based on the defender's knowledge of the attacks on three factors including the model, method parameters, and source examples of the adversarial example generation. We further consider the evaluation based on the defender's knowledge on adversarial example generation methods by leaving out a certain method in model training. By exploring a multi-label approach and an unique evaluation framework from the defender's perspective, our work provides new insights toward protecting against adversarial examples.

Overall, powerful adversarial example generation methods exist and will continue to be proposed, indicating to a certain extent the unlimited capabilities for attackers to easily fool machine learning models; meanwhile, while existing defense approaches are far from being perfect yet. By proposing a multi-label approach, and an unique evaluation framework from the defender's perspective, our work is likely one step toward making a change.

3 DEFENSIVE DISTINCTION PROTECTION APPROACH

We now present the problem formulation and the corresponding defensive distinction protection approach; we also define a threat model that can be used for evaluating our approach.

3.1 Multi-label classification formulation of the problem

To date, a classification problem in adversarial environments is often viewed as a single-label classification problem, where a function or model $f : \mathbb{X} \rightarrow \mathbb{Y}$ is learned to assign each input example $x_i \in \mathbb{X}$ a single label $y_i \in \mathbb{Y}$, where the range $\mathbb{Y}$ is a set of m singletons representing m possible classes such as 10 possible digits. In other words, a single concept or semantic meaning is associated with each example. State-of-the-art defense approaches such as adversarial training and defensive distillation reviewed in Section 2 do not change this basic problem formulation.

However, considering the two important *distinguishability* questions that we propose to investigate (*Question 1: are adversarial examples generated by different methods distinguishable from clean examples?* *Question 2: are adversarial examples generated by different methods distinguishable from each other?*), it is straightforward for us to formulate a classification problem in adversarial environments as a multi-label classification problem. That is, a function or model $f' : \mathbb{X} \rightarrow \mathbb{Y}'$ is learned to assign each input example x_i a pair of labels represented by $y'_i = (a_i, b_i) \in \mathbb{Y}'$. Here, the new range $\mathbb{Y}' = \mathbb{Y} \times \mathbb{Z}$ is the Cartesian product of $\mathbb{Y}$ (i.e., the range in the original classification task) and $\mathbb{Z}$, where $\mathbb{Z}$ is a set of n singletons representing n possible classes regarding if x_i is an adversarial example. In the simplest situation, $n = 2$ and b_i indicates either an example is clean or adversarial. In a more complex situation, $n > 2$ and b_i can indicate either an example is clean or generated by one of $n - 1$ different adversarial example generation methods.

Besides, more labels could be added to include more concepts or semantic meanings in our multi-label classification formulation of the problem. For example, y'_i can be extended to a triplet $(a_i, b_i, c_i) \in \mathbb{Y}'$ where $\mathbb{Y}' = \mathbb{Y} \times \mathbb{Z} \times \mathbb{X}$ is a 3-ary Cartesian product, and $c_i \in \mathbb{X}$ can indicate the source example class from which the input example x_i was created. In the training set, c_i can simply be a_i for a clean example, and is assumed to be known for an adversarial example. This more complex version of formulation has its values on further correlating to the labels of source examples, but we do not explore it in the paper.

Under this new problem formulation, two concepts or semantic meanings are associated with each example. The solution to the problem will help us answer the two research questions, and help defenders simply discard or leverage the identified adversarial examples to better protect their machine learning models as described in Section 1. Moreover, one major advantage of a multi-label classification formulation and the corresponding multi-label learning is on exploring the intrinsic correlations between multiple labels. In real-world adversarial environments, this advantage is significant because it will allow researchers and defenders to perform in-depth analysis of the behaviors of attackers. For example, do attackers create most adversarial examples targeting at some specific classes such as gibbons? do attackers use different generation methods for

different targeted classes? and, do attackers use examples of some specific source classes to create adversarial examples targeting at some specific classes? These and perhaps other types of label correlation analyses will be valuable for inferring the real intentions of attackers, and even for revolutionizing the design of the original machine learning systems.

3.2 Defensive distinction approach

We simply define the *defensive distinction* protection approach as training a model for a multi-label classification problem formulated in adversarial environments to identify adversarial examples and protect against them. So a training set will include both clean examples and generated adversarial examples with their ground truth b_i values correspondingly labeled in $y'_i = (a_i, b_i) \in \mathbb{Y}'$. The ground truth a_i values for clean examples are obviously known based on some existing dataset such as MNIST. The ground truth a_i values for adversarial examples can be missing [33] or not needed (as in our solution described below) depending on specific learning algorithms.

Many multi-label learning algorithms have been proposed with different strategies or considerations on the order-of-correlations among labels [34]. In this paper, we experiment with the simplest first-order strategy that basically transforms a multi-label classification problem into multiple single-label problems. More specifically, three models will be independently trained:

- DDP-Model (defensive distinction primary model): a single-label binary model for distinguishing an input example x_i as either clean or adversarial.
- Original-Model: a single-label multi-class model for the original task on classifying the input example x_i as one of multiple possible objects such as one of 10 digits.
- DDS-Model (defensive distinction secondary model): a single-label multi-class model¹ for distinguishing known adversarial examples as generated by different specific methods.

This simple strategy provides a baseline solution to the defensive distinction approach. It is simple in model formulation, computation, and evaluation, albeit high-order strategies would be more powerful in exploring label correlations [27, 34] which we leave as future work.

3.3 Threat model

In existing defense approaches such as adversarial training and defensive distillation, a threat model only needs to define the capabilities of attackers because those approaches focus on improving the generalization and thus robustness of the machine learning models that perform the original classification task. Typically, two types of attacks are defined: (1) *white-box attacks* that have the full knowledge about the original machine learning model including its architecture, parameter values, and training data, and thus can replicate the exact model to generate adversarial examples, and (2) *black-box attacks* that only have the limited knowledge about the original machine learning model such as the labels for a given set of input examples, and thus often need to create another “substitute model” to generate adversarial examples [6, 24].

¹It can also be merged with the DD-Primary model to create a more complex model as described in Section 3.1 corresponding to the situation when $n > 2$.

Table 1: Representative scenarios and cases in the threat model from the perspective of a defender regarding if some information about the attacker is known.

	AdvGen-Model Known		AdvGen-Model Unknown	
	AdvGen-Parameters Known	AdvGen-Parameters Unknown	AdvGen-Parameters Known	AdvGen-Parameters Unknown
Adv-Examples Known	Scenario 1, Case 1 (S1C1)	Scenario 1, Case 3 (S1C3)	Scenario 2, Case 1 (S2C1)	Scenario 2, Case 3 (S2C3)
Adv-Examples Unknown	Scenario 1, Case 2 (S1C2)	Scenario 1, Case 4 (S1C4)	Scenario 2, Case 2 (S2C2)	Scenario 2, Case 4 (S2C4)

In our defensive distinction approach, a threat model needs to further consider the capabilities of defenders because the effectiveness of our approach is related to the defenders’ knowledge about the machine learning model (including its architecture, parameter values, and training data), adversarial example generation methods, parameters (e.g., maximum distortion) of the generation methods, and source examples used by attackers. We refer to these factors as AdvGen-Model, AdvGen-Methods, AdvGen-Parameters, and Adv-Examples. These considerations together with the capabilities of attackers appear to complicate the overall threat model; however, starting directly from the perspective of defenders, we can still clearly define some representative scenarios and cases as shown in Table 1.

Briefly, Scenarios 1 and 2 represent that an attacker’s AdvGen-Model is known or unknown to a defender, respectively; Cases 1 to 4 represent the four combinations of the defender’s knowledge on AdvGen-Parameters and Adv-Examples, respectively. In total, eight scenario-case combinations (from S1C1 to S2C4) or simply cases are considered in this paper. Note that we do not include AdvGen-Methods in the table because it is reasonable to assume that popular adversarial example generation methods are known to both defenders and attackers (similar to the basic assumption in cryptographic systems that algorithms should not be considered as secrets and could be known by both defenders and attackers); however, we do consider the AdvGen-Methods factor in some of our experiments in the next section by leaving out certain generation method in model training.

Later in our experiments, we also consider a scenario where an attacker has the unlimited access to the output of the DDP-Model, and has the full knowledge of the original machine learning model (i.e., the full white-box capabilities). This is a very strong type of attack since an adversary has the knowledge of both the original model as well as the detector. We refer to the attacker in this scenario as an adaptive adversary, whose goal is to simultaneously bypass the detector while misleading the original model for a targeted incorrect classification.

4 EXPERIMENTS

We now evaluate the effectiveness of our defensive distinction approach on MNIST and CIFAR-10 datasets by performing three sets of experiments with two for DDP-Model and one for DDS-Model. In this section, we introduce the design and setup of experiments, evaluation results, and adaptive adversary experiments.

4.1 Design and setup of experiments

4.1.1 Datasets. We use two popular datasets MNIST and CIFAR-10 to evaluate our defensive distinction approach. MNIST [14] consists of images for handwritten digits from 0 to 9 with a training set of 60,000 examples, and a test set of 10,000 examples. Each of the

examples is a gray-scale image with 28x28 pixels. CIFAR-10 [9] is an image recognition dataset with 10 classes including airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. In this dataset, there are 50,000 training examples and 10,000 test examples with each of the examples as a 32x32 color image.

4.1.2 Architectures. For MNIST, the first CNN is a slight variation of the original LeNet-5 [13] by adding rectified linear units to the convolutional layers and using max pooling in the sub-sampling layers. We refer to this network as LeNet-5. The second CNN network has a convolutional layer with 64 filters followed by two convolutional layers with 128 filters, one 2 by 2 max pooling layer, one fully connected layer with 64 rectified linear units, and one softmax layer. We refer to this network as Basic-CNN. When we train these two CNNs for different classification tasks in our experiments, we use learning rate $\eta=0.001$, number of epochs=6, and batch size=128. For the original 10-digit classification task, LeNet-5 and Basic-CNN achieve 99.05% and 98.99% accuracy, respectively, on the 10,000 images from the MNIST test set.

As for CIFAR-10, the first CNN network has one convolutional layer with 64 filters followed by one convolutional layer with 128 filters, one 2 by 2 average pooling layer, one convolutional layer with 128 filters, one convolutional layer with 256 filters, one 2 by 2 average pooling layer, one convolutional layer with 10 filters, and one softmax layer. We refer to this model as AllConv-CNN. The second CNN is the same as the CIFAR-10 model in [3]. It has two convolutional layers and ReLU layers with 64 filters, one 2 by 2 max pooling layer, two convolutional layers and ReLU layers with 128 filters, one 2 by 2 max pooling layer, two fully connected layers and ReLU layers with 64 filters, and one softmax layer. We refer to this model as CW-CNN. When we perform the training for the two models, we use learning rate = 0.001, number of epochs = 500, and batch_size = 128. For the original image classification task on CIFAR-10, AllConv-CNN and CW-CNN achieve 91.21% and 80% accuracy, respectively, on the 10,000 images from the CIFAR-10 test set.

4.1.3 Adversarial Example Generation. In the generation of adversarial examples for the MNIST dataset, one LeNet-5 network trained with the 60,000 images from the MNIST training set is considered as the attacker’s model (AdvGen-Model) and six generation methods including FGSM, JSMA, BIM, L-BFGS, MIM, and PGD (Section 2) are used by leveraging the v2.1.0 of the CleverHans library [22]. Based on each source example (representing one digit) chosen from the MNIST test set, nine target adversarial example generation attempts (for the rest nine digit classes) are made. In other words, we consider targeted attacks, which are more damaging than untargeted attacks that simply misclassify source examples.

We generate five adversarial example datasets using the six methods based on different method parameters and source examples. As shown in Table 2, datasets adv_tr and adv_C1 are generated by using the same source examples with the index range 0-299 for each digit class such as ‘0’, i.e., the first 300 examples from each digit class in the MNIST test set are chosen as source examples. These two datasets also share the same method parameters, but they are different due to the difference in the initial randomization of their generation. Dataset adv_C2 has the same method parameters but different source examples compared with adv_tr. Dataset

Table 2: Five adversarial example datasets for both MNIST and CIFAR-10. Source example indices are for each digit class. Method parameters notations: ϵ is input variation, θ is pixel step size, γ is maximum distortion, max_iter is the maximum number of iterations, and step is linear search step size.

Notation	Source Examples	FGSM ϵ	JSMA θ, γ	BIM ϵ	L-BFGS max_iter, step	MIM ϵ	PGD ϵ
adv_tr / adv_C1	0-299	0.3	1, 0.1	0.3	1000, 5	0.3	0.3
adv_C2	300-599	0.3	1, 0.1	0.3	1000, 5	0.3	0.3
adv_C3	0-299	0.25	0.5, 0.1	0.25	500, 5	0.25	0.25
adv_C4	300-599	0.2	-0.5, 0.1	0.2	1000, 10	0.2	0.2

adv_C3 has the same source examples but different method parameters compared with adv_tr. Dataset adv_C4 has different source examples and different parameters compared with adv_tr. The total number of adversarial examples in each dataset is approximately $300 \times 9 \times 6 = 16,200$.

Adversarial examples in adv_tr will be used for training our DDP-Model and DDS-Model defensive distinction models based on LeNet-5 and Basic-CNN. Models trained based on LeNet-5 represent the Scenario 1 in Table 1 because the AdvGen-Model also uses LeNet-5, while models trained based on Basic-CNN represent the Scenario 2 in Table 1. Adversarial examples in adv_C1 to adv_C4 will be used for testing our defensive distinction models, corresponding to the Cases C1 to C4 in Table 1.

As for the adversarial example generation for CIFAR-10, the attacker’s model (AdvGen-Model) is one AllConv-CNN network trained with the 50,000 images from the CIFAR-10 training set. Similarly, source examples are selected from the CIFAR-10 test set. We also use the same method parameters listed in Table 2 to generate the five adversarial example datasets. Models trained based on AllConv-CNN represent the Scenario 1 in Table 1, while models trained based on CW-CNN represent the Scenario 2 in Table 1.

So now, our design and setup of experiments allow us to evaluate our defensive distinction approach for all the eight scenario-case combinations from S1C1 to S2C4 (Table 1) for both datasets MNIST and CIFAR-10, respectively.

4.2 Experimental Results on MNIST

4.2.1 DDP-Model with adversarial examples generated by a single method. The simplest DDP-Model is a binary classifier for distinguishing clean examples from adversarial examples generated by a single attack method. For each method, a classifier is trained on a mixture of 1,000 clean examples from the MNIST training set and 1,000 adversarial examples from the dataset adv_tr in Table 2 corresponding to the selected method. The classifier is then evaluated on a mixture of 1,000 clean examples from the MNIST test set and 1,000 adversarial examples from one test dataset adv_C1 to adv_C4 in Table 2 corresponding to the selected method. Note that in this paper, examples are all randomly selected from the corresponding sets; each experiment including the training and testing is independently performed for 10 runs to present the average results.

Model accuracy results for the four test cases with known AdvGen-Parameters (S1C1, S1C2, S2C1, S2C2) are given in Figure 1a. Classification accuracy is very high for all the six methods, suggesting that regardless of the defender’s knowledge on AdvGen-Model (S1 vs. S2) and Adv-Examples (C1 vs. C2), a simple DDP-Model can be

highly effective as long as AdvGen-Parameters are known. Across the four test cases, the models for distinguishing JSMA examples from clean examples struggled the most but still achieved high average accuracy at 92.35%, while the models for other methods such as MIM, PGD, FGSM, and BIM classified examples perfectly. One reason for the imperfect results on the JSMA examples could be that JSMA only selects a small number such as a pair of input features (i.e., pixels) for perturbation in each iteration [24].

Model accuracy results for the remaining four test cases with unknown AdvGen-Parameters (S1C3, S1C4, S2C3, S2C4) are given in Figure 1b. Across the four test cases, classification accuracy is still very high ($> 95\%$) for L-BFGS, FGSM, and BIM, suggesting that a simple DDP-Model can still be highly effective for these three methods even when some AdvGen-Parameters are unknown to a defender. Models for the other three methods did not perform well. Notably, the models only achieved near 50% accuracy (i.e., like random guessing) for MIM, PGD, and JSMA examples under the test cases S1C4 and S2C4, and only performed slightly better than random guessing for MIM and PGD examples under the test cases S1C3 and S2C3; these results suggest that knowing the parameters of some adversarial example generation methods is important to defenders.

Overall, the results in Figure 1a and Figure 1b are highly informative in practice for defenders. Most importantly, it is always beneficial for defenders to train a DDP-Model by using adversarial examples generated based on a variety of parameters. Meanwhile, the exact model architecture and the exact clean examples used by attackers are not influential in the accuracy of the defenders’ models.

So far, we analyzed the accuracy of DDP-Model without considering whether an adversarial example successfully fooled the Original-Model for the 10-digit classification task to misclassify it as a targeted digit class. Back to our multi-label classification formulation of the problem in Section 3.1, correlating the labels of these two models on an adversarial example will be interesting and will indeed further demonstrate the value of our approach. Especially, we define a *Danger Rate* metric, which is the percentage of the successful adversarial examples (i.e., those that achieved their targeted attacks) that are not classified by our DDP-Model as adversarial and thus will continue to incur danger. Without using DDP-Model, the danger rate of successful adversarial examples is considered as 100%. Figure 1c demonstrates that our DDP-Model can significantly reduce the danger rate of successful adversarial examples by over 50% for all the eight test cases and all the six methods.

The results from this experiment are highly informative for a defender that is attempting to construct an adversarial detection system. Most importantly, this experiment emphasizes the influence of the adversarial generation parameters used by the attacker on the success of an adversarial detection system. For example, by comparing S1C1 directly with S1C3, it is apparent from Table 1 that the only difference between the test cases is that the defender is not aware of the adversarial generation parameters used by the attacker. By directly comparing the results from these two test cases, it is clear that the accuracy of the classifier falls dramatically, especially for the MIM attack where accuracy drops from near 100% to near 50%. This result does not hold for all attack methods, but is

(a) Accuracy for S1C1, S1C2, S2C1, S2C2. (b) Accuracy for S1C3, S1C4, S2C3, S2C4. (c) Danger rate of successful adv. examples.

Figure 1: DDP-Model with MNIST adversarial examples generated by a single method: model accuracy is in (a) and (b), and danger rate of successful adversarial examples is in (c).

certainly a phenomenon that should be considered in the design of a detection system.

Interestingly, the experimental results suggest that the exact model architecture used by the attacker as well as the exact clean examples used for training by the defender are not influential in the performance of the classifier. This conclusion is apparent from a direct comparison of pairs of test cases, such as S1C1 and S1C2 or S1C1 and S2C1, which isolate the variables of interest. By using a similar, though not identical, model architecture as the attacker as well as training on representative, but different, clean examples, high accuracy can still be achieved by the attack detection mechanism.

4.2.2 DDP-Model with adversarial examples generated by multiple methods. A slightly more advanced DDP-model is a binary classifier for distinguishing clean examples from a collection of adversarial examples generated by multiple attack methods. Since it is unreasonable to assume that a defender will always know in advance all the methods used by an attacker, we introduce the concept of an excluded or “left-out” method in the experiments to consider the AdvGen-Methods factor. Basically, training is on adversarial examples selected from five attack methods with one method left out, while testing is on adversarial examples selected from all the six methods. We rotate the left-out attack method to explore the differences between methods, and compare the accuracy of the left-out classifiers with that of baseline classifiers which do not leave out a method.

A left-out classifier is trained on a mixture of 5,000 clean examples from the MNIST training set and 1,000 adversarial examples by each of the five non-excluded attack methods from the dataset `adv_tr` in Table 2. A baseline classifier is trained on a mixture of 6,000 clean examples from the MNIST training set and 1,000 adversarial examples by each of the six attack methods from the dataset `adv_tr`. They are all evaluated on 6,000 clean examples from the MNIST test set and 1,000 adversarial examples by each of the six attack methods from one test dataset `adv_C1` to `adv_C4`.

Accuracy results for the left-out and baseline classifiers are shown in Figure 2a, where the x-axis identifies the left-out method (with *NONE* for baseline) in the training of a classifier, and the y-axis displays the overall accuracy of a classifier on a collection of 6,000 adversarial examples and 6,000 clean examples. In several of the test cases (S1C1, S1C2, S1C3, S2C1, S2C2, S2C3), classification accuracy is above 90% (and often even well above 95%) across all methods. In the remaining test cases S1C4 and S2C4, classification accuracy is lower but still above 80% across all methods.

Figure 2b further details the recall (or sensitivity) of a given classifier on a set of 1,000 adversarial examples generated by the left-out method. We can see that clear disparities exist between two groups of methods: L-BFGS and JSMA examples well evade the classification when the corresponding method was not explicitly considered in the training, while the examples of the other four methods are accurately classified as adversarial even when the corresponding method was not considered in the training. These disparities can be explained to a certain extent by the fact that BIM, MIM, and PGD are all fine-grained refinements over FGSM as reviewed in Section 2. Figure 2c further demonstrates that our models can significantly reduce the danger rate of successful adversarial examples by over 90% (and often over 95%) for all the eight test cases and all the six methods.

Overall, Figure 2 clearly demonstrates that a DDP-Model trained on adversarial examples generated by multiple methods can be highly effective. While it is always beneficial for defenders to train a DDP-Model using adversarial examples generated by as many as possible methods, our models do have the strong capability to correctly classify the adversarial examples of an unknown method whose siblings are known as shown in the results for the four methods FGSM, BIM, MIM, and PGD that belong to the same family. In addition, similar to what we have observed in the previous experiment, knowing the parameters of some adversarial example generation methods is still important to defenders.

Most importantly, these results reveal the uniqueness of the signature that each method creates when generating an adversarial samples. Clearly, L-BFGS adds perturbations to a sample that appear very distinctly from those added by FGSM, while perturbations added by MIM appear similar to those added by BIM from the perspective of the classifier. An attacker that is able to perturb legitimate samples in a way that is much different from the expectations of a defender has a competitive advantage, and as a result the defender is tasked with considering a wide variety of methods that an attacker may use or finding a way to generalize the classifier beyond only the potential attack methods that are known to the defender.

Regardless, the binary classifier demonstrated high accuracy when evaluated against a mixed pool of adversarial samples. The results suggest that variables such as knowledge of the attacker’s model, access to the attacker’s source samples, and knowledge of the adversarial generation parameters are much less influential than knowledge of the adversarial method chosen by the attacker.

Figure 2: DDP-Model with MNIST adversarial examples generated by multiple methods: model accuracy is in (a), recall on the left-out adversarial examples is in (b), and danger rate is in (c).

Figure 3: Model accuracy of DDS-Model with MNIST adversarial examples generated by multiple methods.

4.2.3 DDS-Model with adversarial examples generated by multiple methods. DDS-Model aims to distinguish known adversarial examples as generated by different specific methods. It can inform a defender about the specific techniques used by an attacker, and further help the defender exploit the weaknesses of the attack techniques. We trained a classifier using 1,000 adversarial examples generated by each of the six methods for a total of 6,000 examples from the dataset `adv_tr` in Table 2. We then evaluated the classifier using 1,000 adversarial examples generated by each of the six methods for a total of 6,000 examples from one test dataset `adv_C1` to `adv_C4`.

Model accuracy results for the four test cases with known AdvGen-Parameters (S1C1, S1C2, S2C1, S2C2) are shown in Figure 3a. With the exception of the MIM and PGD methods, adversarial examples from other methods are all accurately ($> 90\%$) classified across the four test cases. Model accuracy results for the remaining four test cases with unknown AdvGen-Parameters (S1C3, S1C4, S2C3, S2C4) are provided in Figure 3b. Across these four test cases, PGD and MIM completely evaded our classifiers and FGSM was classified with low accuracy, but the remaining three methods (except for JSMA in case S1C4) were classified accurately.

The confusion matrices summed from the 10 runs of each experiment for all the eight cases are provided in Figure 4. Note that the accuracy scores given in Figure 3a and Figure 3b are selected from the diagonal of the confusion matrix and normalized by the number of evaluated examples. We summarize three observations from the confusion matrices. First, the confusion matrices can help explain why PGD and MIM are the most difficult methods for correct classification. On the one hand, these two methods confuse each other.

On the other hand, their examples are largely classified as BIM examples especially in the four cases S1C3, S1C4, S2C3, and S2C4; it is presumably because while PGD starts with random perturbation [18] and MIM integrates momentum to escape local maxima [5] thus both resulting in diverse examples, they still resemble their close sibling BIM [11] very much. Second, FGSM is the third most difficult method in terms of classification accuracy. FGSM examples are misclassified as BIM examples in the three cases S1C4, S2C3, and S2C4. However, BIM has the highest accuracy compared to PGD, MIM, and FGSM. One possible explanation is that BIM covers more diverse adversarial examples due to its iterative implementation. Third, examples from JSMA and L-BFGS are mostly classified correctly in the eight cases except that a significant number of JSMA examples are misclassified as L-BFGS examples in the case S1C4. This indicates that adversarial examples generated from optimal methods like JSMA and L-BFGS have strong distinguishability compared to less optimal methods.

Overall, these results demonstrate the strong distinguishability of competing adversarial example generation methods. Similar to what is observed for a DDP-Model in the two previous experiments, it is also beneficial for defenders to train a DDS-Model by using adversarial examples generated based on a variety of parameters; meanwhile, the exact model architecture and the exact clean examples used by attackers are not influential in the accuracy of the defenders' models.

4.3 Experimental Results on CIFAR-10

We perform the same three sets of experiments to evaluate two DDP-Models and one DDS-Model on CIFAR-10. We use both AllConv-CNN and CW-CNN architectures to train the three models while only using AllConv-CNN for the adversarial example generation. The training and evaluation procedures on CIFAR-10 are identical to that of MNIST in Section 4.2, so we only present the results here.

4.3.1 DDP-Model with adversarial examples generated by a single method. Figure 5 shows the model accuracy results for CIFAR-10, and the danger rate for successful adversarial examples. In general, the model accuracy results are lower for CIFAR-10 compared to the accuracy results for MNIST shown in Figure 1. One reason could be that the original image classification on CIFAR-10 is more challenging than on MNIST. Existing detection techniques tend to be weaker on CIFAR-10, as also observed in [2].

In Figure 5a and Figure 5b, classification accuracy is mostly high for MIM, PGD, FGSM, and BIM, suggesting that regardless of the

	BIM	FGSM	JSMA	L-BFGS	MIM	PGD
BIM	9933	57	0	10	0	0
FGSM	973	9027	0	0	0	0
JSMA	18	16	9288	665	0	13
L-BFGS	9	0	66	9925	0	0
MIM	0	7	0	0	7714	2279
PGD	0	9	0	0	1163	8828

(a) S1C1

	BIM	FGSM	JSMA	L-BFGS	MIM	PGD
BIM	9458	540	0	2	0	0
FGSM	223	9777	0	0	0	0
JSMA	13	0	9613	369	0	5
L-BFGS	13	0	447	9540	0	0
MIM	0	1	0	0	9095	904
PGD	0	1	0	0	1360	8639

(b) S1C2

	BIM	FGSM	JSMA	L-BFGS	MIM	PGD
BIM	9903	57	1	39	0	0
FGSM	3466	6534	0	0	0	0
JSMA	1	5	9725	248	0	21
L-BFGS	36	0	127	9837	0	0
MIM	8765	1234	1	0	0	0
PGD	9853	136	0	11	0	0

(c) S1C3

	BIM	FGSM	JSMA	L-BFGS	MIM	PGD
BIM	9478	0	1	521	0	0
FGSM	9926	65	0	9	0	0
JSMA	0	0	4046	5954	0	0
L-BFGS	28	0	285	9687	0	0
MIM	9917	9	1	73	0	0
PGD	9628	2	2	368	0	0

(d) S1C4

	BIM	FGSM	JSMA	L-BFGS	MIM	PGD
BIM	9710	290	0	0	0	0
FGSM	120	9880	0	0	0	0
JSMA	11	0	9861	124	0	4
L-BFGS	5	0	155	9840	0	0
MIM	0	0	0	0	9005	995
PGD	0	0	0	0	1129	8871

(e) S2C1

	BIM	FGSM	JSMA	L-BFGS	MIM	PGD
BIM	9942	58	0	0	0	0
FGSM	282	9718	0	0	0	0
JSMA	4	1	9811	183	0	1
L-BFGS	7	0	99	9894	0	0
MIM	0	0	0	0	9035	965
PGD	0	0	0	0	351	9649

(f) S2C2

	BIM	FGSM	JSMA	L-BFGS	MIM	PGD
BIM	9999	0	0	1	0	0
FGSM	6459	3541	0	0	0	0
JSMA	0	1	9954	44	0	1
L-BFGS	2	0	179	9819	0	0
MIM	9753	247	0	0	0	0
PGD	9996	4	0	0	0	0

(g) S2C3

	BIM	FGSM	JSMA	L-BFGS	MIM	PGD
BIM	9941	0	0	59	0	0
FGSM	9962	38	0	0	0	0
JSMA	0	0	9218	782	0	0
L-BFGS	7	0	103	9890	0	0
MIM	9992	0	0	8	0	0
PGD	9959	0	0	41	0	0

(h) S2C4

Figure 4: Confusion matrices for DDS-Model on MNIST.

defender’s knowledge on AdvGen-Model (S1 vs. S2) and AdvGen-Examples (C1 vs. C2), a simple DDP-Model can be highly effective for the four methods as long as AdvGen-Parameters are known. However, the accuracy for JSMA and L-BFGS is mostly between 50-60%. One reason for the poor performance on the JSMA and L-BFGS examples could be that they are optimized methods which add smaller perturbations compared to the other four methods.

Recall that *Danger Rate* is the percentage of the successful adversarial examples (i.e., those that achieved their targeted attacks) that are not classified by our DDP-Model as adversarial and thus will continue to incur danger. Without the DDP-Model, the danger rate for successful adversarial examples is considered as 100%. Figure 5c shows the danger rate for successful adversarial examples. The danger rate is very low for MIM, PGD, FGSM, and BIM but not JSMA and L-BFGS, suggesting that our DDP-Model can reduce the danger rate of successful adversarial examples by over 85% for MIM, PGD, FGSM, and BIM.

4.3.2 DDP-Model with adversarial examples generated by multiple methods. Figure 6 shows the accuracy results for the left-out and baseline classifiers, the recall (or sensitivity) results for the six left-out classifiers, and danger rate results of successful adversarial examples. In Figure 6a, the accuracy results are mostly high, suggesting that DDP-Models with left-out method are still effective although in some cases the accuracy can be as low as 50%. In Figure 6b, the recall results are mostly high for MIM, BIM, FGSM and PDG while the results for JSMA and L-BFGS are relatively low. Since BIM, MIM, and PGD are all fine-grained refinements over FGSM, a classifier with any one of them as the left-out method still performs well on adversarial examples generated from the remaining methods. This observation is consistent with the corresponding MNIST results shown in Figure 2b, and is beneficial for defenders to detect adversarial examples generated by new variations of some existing methods. In Figure 6c, the danger rate is mostly low which means that our models can reduce the danger rate by over 50% in most cases.

Compared to the corresponding MNIST results in Figure 2, Figure 6 further illustrates that DDP-Model trained on adversarial examples generated by multiple methods can be highly effective.

4.3.3 DDS-Model with adversarial examples generated by multiple methods. Model accuracy results for the four test cases with known AdvGen-Parameters (S1C1, S1C2, S2C1, S2C2) are shown in Figure 7a. Model accuracy results for the remaining four test cases with unknown AdvGen-Parameters (S1C3, S1C4, S2C3, S2C4) are provided in Figure 7b. The accuracy results for known AdvGen-Parameters are generally higher than accuracy results for unknown AdvGen-Parameters, suggesting that it is important for defenders to know the parameters of adversarial generation methods.

Figure 8 shows the confusion matrices summed from 10 runs for all the eight cases. First, compared to the MNIST confusion matrices in Figure 4, BIM, FGSM, MIM, and PGD are still confusing each other because of the fact that BIM, MIM, and PGD are all fine-grained refinements over FGSM and they belong to the same family. Second, FGSM examples are mostly classified as BIM examples in the three cases S2C1, S2C2, and S2C3. They are also classified as examples of MIM and PGD in the two cases S1C4 and S2C4. However, BIM has relatively higher accuracy compared to PGD, MIM, and FGSM. One possible explanation is that BIM covers more diverse adversarial examples due to its iterative implementation. Third, examples from JSMA and L-BFGS are mostly classified correctly in the three cases S1C1, S1C2, and S1C3. On the one hand, in all of the eight cases, JSMA and L-BFGS confuse each other. On the other hand, a significant number of examples from JSMA and L-BFGS are classified as BIM examples in the five cases S1C4, S2C1, S2C2, S2C3, and S2C4. Again, this is probably due to the fact that BIM explores more diverse examples in the search space.

Overall, these results demonstrate the strong distinguishability of competing adversarial example generation methods in CIFAR-10 despite less accurate results compared to MNIST. Similar to the observation in Section 4.2, it is also beneficial for defenders to train a DDS-Model by using different sets of adversarial examples generated based on a variety of method parameters.

4.4 Adaptive Adversary Experiments

Now, we evaluate our detection mechanism in the scenario when an adaptive adversary is presented, i.e., the attacker has full knowledge of the original model and is aware of the detector which is the

Figure 5: DDP-Model with CIFAR-10 adversarial examples generated by a single method: model accuracy is in (a) and (b), and danger rate of successful adversarial examples is in (c).

Figure 6: DDP-Model with CIFAR-10 adversarial examples generated by multiple methods: model accuracy is in (a), recall on the left-out adversarial examples is in (b), and danger rate is in (c).

Figure 7: Model accuracy of DDS-Model with CIFAR-10 adversarial examples generated by multiple methods.

DDP-Model. Specifically, the attacker has white-box capability of the original model, and unlimited query capability of the detector. The goal of the adaptive adversary is to simultaneously bypass the detector while misleading the original model for a targeted incorrect classification.

We design two white-box adaptive attacks including JSMA-based and MIM-based attacks to evaluate a DDP-Model trained with clean examples, and adversarial examples generated by JSMA. We use 10 clean examples as source examples to generate 9 adversarial examples for each source example with 9 target classes other than the source class. Without the detector, the success rates for JSMA and MIM using MNIST and CIFAR-10 are all 100%. The average generation time for each adversarial example is around one second.

We define *failure rate* as the percent of adversarial examples failed to bypass the detector after five minutes of continuous generation (and query on the output of the DDP-Model). With the detector, the failure rates for JSMA and MIM attacks on MNIST are 6.67% and 15.56%, respectively. On CIFAR-10, the failure rates for JSMA and MIM attacks are 36.67% and 100%, respectively. Our experiments demonstrate that it becomes harder for attackers to

generate adversarial examples with the existence of our DDP-Model as a detector.

5 CONCLUSION

We proposed two important research questions that concern the distinguishability of adversarial examples, and formulated a classification problem in adversarial environments as a multi-label classification problem. We proposed a defensive distinction protection approach to answer the two questions and address the problem. We designed and performed experiments using the MNIST and CIFAR-10 datasets and eight representative cases. Our experimental results demonstrate the strong distinguishability of adversarial examples, and the practical as well as research value of our approach. Our work provides insights of some correlations between adversarial examples generated from different methods. It also recommends the defenders to use a variety of parameters and different methods to train a detector. Our work also suggests many possibilities for the future work such as adopting high-order multi-label learning strategies to further explore the intrinsic correlations of labels as discussed in Section 3.2, and investigating the appropriate ways for integrating defensive distinction with other defense approaches.

ACKNOWLEDGMENTS

Research was sponsored by the Army Research Office and was accomplished under Grant Number W911NF-17-1-0447. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein. This work was also partially supported by a Google Cloud Platform grant.

	BIM	FGSM	JSM	L-BFGS	MM	PCD
BIM	8726	0	133	5	1	1135
FGSM	45	8551	0	0	426	978
JSM	921	15	5881	3096	4	83
L-BFGS	1	0	1888	8111	0	0
MM	741	1241	0	0	7493	525
PCD	2480	2	35	2	22	7459

(a) S1C1

	BIM	FGSM	JSM	L-BFGS	MM	PCD
BIM	7926	0	13	0	4	2057
FGSM	2	9197	0	0	792	9
JSM	124	9	6854	3003	2	8
L-BFGS	9	0	1158	8833	0	0
MM	6	1543	0	0	8280	171
PCD	893	0	3	0	411	8693

(b) S1C2

	BIM	FGSM	JSM	L-BFGS	MM	PCD
BIM	9633	0	29	0	1	337
FGSM	4	8587	0	0	1396	13
JSM	18	0	7795	2187	0	0
L-BFGS	5	0	4079	5916	0	0
MM	79	17	0	0	5702	4202
PCD	6992	0	18	0	1	2989

(c) S1C3

	BIM	FGSM	JSM	L-BFGS	MM	PCD
BIM	8213	0	1773	5	0	9
FGSM	528	0	9	2	585	8876
JSM	5084	14	3594	238	48	1022
L-BFGS	5	0	1733	8262	0	0
MM	5903	0	48	0	4	4045
PCD	8607	0	1377	2	0	14

(d) S1C4

	BIM	FGSM	JSM	L-BFGS	MM	PCD
BIM	5338	16	2209	0	609	1828
FGSM	1000	8664	2	0	334	0
JSM	2806	19	5586	1397	192	0
L-BFGS	3000	0	4055	2945	0	0
MM	1532	3114	64	0	4805	485
PCD	4667	28	716	0	1261	3328

(e) S2C1

	BIM	FGSM	JSM	L-BFGS	MM	PCD
BIM	4529	15	1637	6	60	3753
FGSM	2976	5851	28	0	1118	27
JSM	2087	76	5563	2255	7	12
L-BFGS	2119	8	1923	5950	0	0
MM	3476	2153	4	0	2826	1541
PCD	4390	15	298	0	350	4947

(f) S2C2

	BIM	FGSM	JSM	L-BFGS	MM	PCD
BIM	8208	49	866	681	194	2
FGSM	3015	6222	0	0	763	0
JSM	3005	2	2687	4306	0	0
L-BFGS	3000	0	2252	4748	0	0
MM	5599	1351	4	19	2732	295
PCD	8674	43	485	377	377	44

(g) S2C3

	BIM	FGSM	JSM	L-BFGS	MM	PCD
BIM	9941	0	0	59	0	0
FGSM	9962	38	0	0	0	0
JSM	0	0	9218	782	0	0
L-BFGS	7	0	103	9890	0	0
MM	9992	0	0	8	0	0
PCD	9959	0	0	41	0	0

(h) S2C4

Figure 8: Confusion matrices for DDS-Model on CIFAR-10.

REFERENCES

- [1] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Srđić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. 2013. Evasion Attacks against Machine Learning at Test Time. In *European Conference on Machine Learning and Knowledge Discovery in Databases*. 387–402.
- [2] Nicholas Carlini and David A. Wagner. 2017. Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods. In *ACM Workshop on Artificial Intelligence and Security (AISec@CCS)*. 3–14.
- [3] Nicholas Carlini and David A. Wagner. 2017. Towards Evaluating the Robustness of Neural Networks. In *IEEE Symposium on Security and Privacy (SP)*. 39–57.
- [4] Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinpeng Yi, and Cho-Jui Hsieh. 2018. EAD: Elastic-Net Attacks to Deep Neural Networks via Adversarial Examples. In *AAAI Conference on Artificial Intelligence (AAAI)*.
- [5] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. 2018. Boosting Adversarial Attacks with Momentum. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [6] Ian J. Goodfellow, Patrick McDaniel, and Nicolas Papernot. 2018. Making Machine Learning Robust Against Adversarial Inputs. *Commun. ACM* 61, 7 (2018), 56–66.
- [7] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. In *International Conference on Learning Representations (ICLR)*.
- [8] Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick D. McDaniel. 2017. On the (Statistical) Detection of Adversarial Examples. *CoRR abs/1702.06280* (2017). <http://arxiv.org/abs/1702.06280>
- [9] Alex Krizhevsky. 2009. Learning multiple layers of features from tiny images. (2009).
- [10] Aditya Kuppe, Slawomir Grzonkowski, Muhammad Rizwan Asghar, and Nhien-An Le-Khac. 2019. Black Box Attacks on Deep Anomaly Detectors. In *International Conference on Availability, Reliability and Security (ARES)*. 1–10. <https://doi.org/10.1145/3339252.3339266>
- [11] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. 2017. Adversarial Examples in the Physical World. In *International Conference on Learning Representations (ICLR) Workshop Track*.
- [12] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. 2017. Adversarial Machine Learning at Scale. In *International Conference on Learning Representations (ICLR)*.
- [13] Yann Lecun, Lon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based Learning Applied to Document Recognition. In *Proceedings of the IEEE*. 2278–2324.
- [14] Yann LeCun and Corinna Cortes. 2010. MNIST handwritten digit database. <http://yann.lecun.com/exdb/mnist/>. (2010). <http://yann.lecun.com/exdb/mnist/>
- [15] Xin Li and Fuxin Li. 2017. Adversarial Examples Detection in Deep Networks with Convolutional Filter Statistics. *IEEE International Conference on Computer Vision (ICCV) abs/1612.07767*. <https://arxiv.org/abs/1612.07767>
- [16] Jiajun Lu, Theerassit Issaranon, and David A. Forsyth. 2017. SafetyNet: Detecting and Rejecting Adversarial Examples Robustly. In *IEEE International Conference on Computer Vision (ICCV)*. 446–454. <https://doi.org/10.1109/ICCV.2017.56>
- [17] JShiqing Ma, Yingqi Liu, Guanhong Tao, Wen-Chuan Lee, and Xiangyu Zhang. 2019. NIC: Detecting Adversarial Samples with Neural Network Invariant Checking. In *Network and Distributed Systems Security Symposium (NDSS)*.
- [18] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations (ICLR)*.
- [19] Dongyu Meng and Hao Chen. 2017. MagNet: A Two-Pronged Defense against Adversarial Examples. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 135–147. <https://doi.org/10.1145/3133956.3134057>
- [20] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. 2017. On Detecting Adversarial Perturbations. In *International Conference on Learning Representations (ICLR)*.
- [21] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. DeepFool: a simple and accurate method to fool deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [22] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian J. Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, Rujun Long, and Patrick McDaniel. 2018. Technical Report on the CleverHans v2.1.0 Adversarial Examples Library. *CoRR abs/1610.00768* (2018).
- [23] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. 2017. Practical Black-Box Attacks Against Machine Learning. In *Asia Conference on Computer and Communications Security*. 506–519.
- [24] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. 2016. The Limitations of Deep Learning in Adversarial Settings. In *IEEE European Symposium on Security and Privacy*. 372–387.
- [25] Nicolas Papernot, Patrick D. McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2016. Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks. In *IEEE Symposium on Security and Privacy (SP)*. 582–597.
- [26] Yi Qin, Ryan Hunt, and Chuan Yue. 2019. On Improving the Effectiveness of Adversarial Training. In *ACM International Workshop on Security and Privacy Analytics (IWSPA)*. 5–13. <https://doi.org/10.1145/3309182.3309190>
- [27] Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. 2009. Classifier Chains for Multi-label Classification. In *European Conference on Machine Learning and Knowledge Discovery in Databases*. 254–269.
- [28] Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J. Fleet. 2016. Adversarial Manipulation of Deep Representations. In *International Conference on Learning Representations (ICLR)*.
- [29] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2014. Intriguing Properties of Neural Networks. In *International Conference on Learning Representations (ICLR)*.
- [30] Jonathan Uesato, Brendan O'Donoghue, Aron van den Oord, and Pushmeet Kohli. 2018. Adversarial Risk and the Dangers of Evaluating Against Weak Attacks. In *International Conference on Machine Learning (ICML)*.
- [31] Jingyi Wang, Jun Sun, Peixin Zhang, and Xinyu Wang. 2018. Detecting Adversarial Samples for Deep Neural Networks through Mutation Testing. *CoRR abs/1805.05010* (2018). <http://arxiv.org/abs/1805.05010>
- [32] Weilin Xu, Yanjun Qi, and David Evans. 2016. Automatically Evading Classifiers: A Case Study on PDF Malware Classifiers. In *Network and Distributed System Security Symposium (NDSS)*.
- [33] Hsiang-Fu Yu, Prateek Jain, Purushottam Kar, and Inderjit S. Dhillon. 2014. Large-scale Multi-label Learning with Missing Labels. In *International Conference on Machine Learning (ICML)*.
- [34] Min-Ling Zhang and Zhi-Hua Zhou. 2014. A Review on Multi-Label Learning Algorithms. *IEEE Transactions on Knowledge and Data Engineering* 26, 8 (2014), 1819–1837.