Mitigating Evasion Attacks to Deep Neural Networks via Region-based Classification

Xiaoyu Cao, Neil Zhenqiang Gong
ECE Department, Iowa State University
{xiaoyuc, neilgong}@iastate.edu

ABSTRACT

Deep neural networks (DNNs) have transformed several artificial intelligence research areas including computer vision, speech recognition, and natural language processing. However, recent studies demonstrated that DNNs are vulnerable to adversarial manipulations at testing time. Specifically, suppose we have a testing example, whose label can be correctly predicted by a DNN classifier. An attacker can add a small carefully crafted noise to the testing example such that the DNN classifier predicts an incorrect label, where the crafted testing example is called *adversarial example*. Such attacks are called *evasion attacks*. Evasion attacks are one of the biggest challenges for deploying DNNs in safety and security critical applications such as self-driving cars.

In this work, we develop new DNNs that are robust to state-of-the-art evasion attacks. Our key observation is that adversarial examples are close to the classification boundary. Therefore, we propose *region-based classification* to be robust to adversarial examples. Specifically, for a benign/adversarial testing example, we ensemble information in a hypercube centered at the example to predict its label. In contrast, traditional classifiers are *point-based classification*, i.e., given a testing example, the classifier predicts its label based on the testing example alone. Our evaluation results on MNIST and CIFAR-10 datasets demonstrate that our region-based classification can significantly mitigate evasion attacks without sacrificing classification accuracy on benign examples. Specifically, our region-based classification achieves the same classification accuracy on testing benign examples as point-based classification, but our region-based classification is significantly more robust than point-based classification to state-of-the-art evasion attacks.

1 INTRODUCTION

Deep neural networks (DNNs) are unprecendentedly effective at solving many challenging artificial intelligence problems such as image recognition [9], speech recognition [6], natural language processing [14], and playing games [21]. For instance, DNNs can recognize images with accuracies that are comparable to human [9]; and they can outperform the best human Go players [21].

However, researchers in various communities–such as security, machine learning, and computer vision–have demonstrated that DNNs are vulnerable to attacks at testing time [2, 3, 11, 15, 17, 18, 22]. For instance, in image recognition, an attacker can add a small noise to a testing example such that the example is misclassified by a DNN classifier. The testing example with noise is called *adversarial example* [22]. In contrast, the original example is called *benign example*. Usually, the noises are so small such that, to human, the benign example and adversarial example still have the same label. Figure 1 shows some adversarial examples for digit recognition in the MNIST dataset. The adversarial examples were generated by the state-of-the-art evasion attacks proposed by Carlini and Wagner [2]. We use the same DNN classifier as the one used by them. The examples in the ith row have true label i, while the examples in the jth column are predicted to have label j by the DNN classifier, where $i, j = 0, 1, \cdots, 9$.

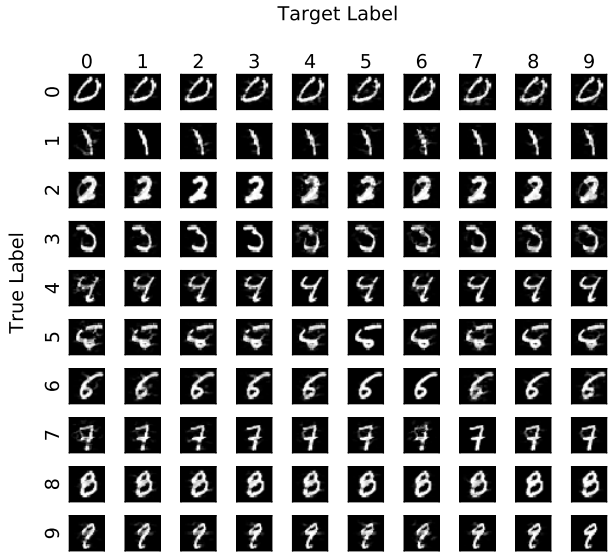


Figure 1: Adversarial examples generated by an evasion attack proposed by Carlini and Wagner [2].

Evasion attacks limit the use of DNNs in safety and security critical applications such as self-driving cars. The adversarial examples can make self-driving cars make unwanted decisions. For instance, one basic capability of self-driving cars is to automatically recognize stop signs and traffic lights. Suppose an adversary creates an adversarial stop sign, i.e., the adversary adds several human-unnoticeable dots to a stop sign, such that the self-driving car does not recognize it as a stop sign. As a result, self-driving

ACSAC'17, San Juan, Puerto Rico

DOI: 10.1145/nnnnnnn.nnnnnnn

cars will not stop at the stop sign and may collide with other cars, resulting in severe traffic accidents.

To defend against evasion attacks, Goodfellow et al. [3] proposed to train a DNN via augmenting the training dataset with adversarial examples, which is known as *adversarial training*. Specifically, for each training benign example, the learner generates a training adversarial example using evasion attacks. Then, the learner uses a standard algorithm (e.g., back propagation) to learn a DNN using the original training benign examples and the corresponding adversarial examples. Adversarial training is not robust to adversarial examples that are unseen during training. Papernot et al. [19] proposed a distillation based method to train DNNs. A DNN that is trained via distillation can significantly reduce the success rates of the evasion attacks also proposed by Papernot et al. [18]. However, Carlini and Wagner [2] demonstrated that their attacks can still achieve 100% success rates for DNNs trained with distillation. Moreover, the noises added to the benign examples when generating adversarial examples are just slightly higher for distilled DNNs than those for undistilled DNNs. Carlini and Wagner [2] concluded that all defenses should be evaluated against state-of-the-art evasion attacks, i.e., the attacks proposed by them at the time of writing this paper. For simplicity, we call their attacks CW.

Our work: We propose a new defense method called *region-based classification*. Our method can reduce success rates of the CW attacks from 100% to less than 16%, while not impacting classification accuracy on testing benign examples. First, we performed a measurement study about the adversarial examples generated by the CW attacks. We trained a 10-class DNN classifier on the standard MNIST dataset to recognize digits in images. The DNN has the same architecture as the one used by Carlini and Wagner [2]. Suppose we have a testing digit 0. We use a CW attack to generate an adversarial example for each target label 1, 2, $\cdots$, 9. Each example is represented as a data point in a high-dimensional space. For each adversarial example, we sample 10,000 data points from a small *hypercube* centered at the adversarial example in the high-dimensional space. We use the DNN classifier to predict labels for the 10,000 data points. We found that a majority of the 10,000 data points are still predicted to have label 0. Our measurement results indicate that 1) the adversarial examples generated by the CW attacks are close to the classification boundary, and 2) ensembling information in the hypercube around an adversarial example could correctly predict its label.

Second, based on our measurement results, we propose a *region-based classification*. In our region-based classification, we learn a DNN classifier using standard training algorithms. When predicting label for a testing example (benign or adversarial), we sample m data points uniformly at random from the hypercube that is centered at the testing example and has a length of r. We use the DNN classifier to predict label for each sampled data point. Finally, we predict the label of the testing example as the one that appears the most frequently in the sampled data points. To distinguish our region-based classification with traditional DNN classification, we call traditional DNN *point-based classification*.

One challenge for our region-based classification is how to determine the length r of the hypercube. r is a critical parameter that controls the tradeoff between robustness to adversarial examples and classification accuracy on benign examples. To address the challenge, we propose to learn the length r using a validation dataset consisting of only benign examples. We do not use adversarial examples because the adversarial examples used by the attacker may not be accessible to the defender. Our key idea is to select the maximal length r such that the classification accuracy of our region-based classification on the validation dataset is no smaller than that of the standard point-based DNN classifier. We propose to select the maximal possible length, so an adversarial example needs a larger noise to move further away from the classification boundary in order to evade our region-based classification.

Third, we evaluate our region-based classification using two standard image recognition datasets, MNIST and CIFAR-10. We use the CW attacks to generate adversarial examples. First, our evaluation results demonstrate that our region-based classification achieves the same classification accuracy on testing benign examples with the standard point-based classification. However, adversarial training and distillation sacrifice classification accuracy. Second, for our region-based classification, the CW attacks have less than 16% and 7% success rates on the MNIST and CIFAR-10 datasets, respectively. In contrast, for standard point-based classification, adversarial training, and distillation, the CW attacks achieve 100% success rates on both datasets. Third, we consider an attacker strategically adapts the CW attacks to our region-based classification. In particular, the attacker adds more noise to an adversarial example generated by a CW attack to move it further away from the classification boundary. Our results demonstrate that our region-based classification can also effectively defend against such adapted attacks. In particular, the largest success rate that the adapted attacks can achieve on the MNIST dataset is 64%, when the attacker doubles the noises added to adversarial examples. We conclude that, in the future, researchers who develop powerful evasion attacks should evaluate their attacks against our region-based classification instead of standard point-based classification.

In summary, our contributions are as follows:

- We perform a measurement study to characterize the adversarial examples generated by state-of-the-art evasion attacks.
- We propose a region-based classification to defend against state-of-the-art evasion attacks, while not impacting classification accuracy on benign examples.
- We evaluate our region-based classification using two image datasets. Our results demonstrate that 1) our method does not impact classification accuracy on benign examples, and 2) our method can significantly reduce success rates of state-of-the-art evasion attacks as well as attacks that are strategically adjusted to our region-based classification.

2 BACKGROUND AND RELATED WORK

2.1 Deep Neural Networks (DNNs)

A deep neural network (DNN) consists of an input layer, several hidden layers, and an output layer. The output layer is often a *softmax* layer. The neurons in one layer are connected with neurons in the next layer with certain patterns, e.g., fully connected, convolution, or max pooling [9]. In the *training phase*, the weights on the connections are often learnt via back-propagation with a

Table 1: State-of-the-art evasion attacks.

Attack	Noise metric
CW-L_0 [2]	L_0
CW-L_2 [2]	L_2
CW-L_∞ [2]	L_∞

training dataset. In the *testing phase*, the DNN is used to predict labels for examples that are unseen in the training phase. Specifically, suppose we have L classes, denoted as $\{1, 2, \cdots, L\}$. Both the layer before the output layer and the output layer have L neurons. Let $x \in R^n$ be an unseen example, which is a n-dimension vector; x_j represents the jth dimension of x. We denote the output of the ith neuron before the output layer as $Z_i(x)$, and we denote the output of the ith neuron in the output layer as $F_i(x)$, where $i = 1, 2, \cdots, L$. The outputs $Z_1(x), Z_2(x), \cdots, Z_L(x)$ are also called *logits*. Since the output layer is a softmax layer, $F_i(x)$ represents the probability that x has a label i; and the L outputs sum to 1, i.e., $\sum_{i=1}^{L} F_i(x) = 1$. The label of x is predicted to be the one that has the largest probability, i.e., $C(x) = \text{argmax}_i\, F_i(x)$, where $C(x)$ is the predicted label.

A classifier essentially can be viewed as a *classification boundary* that divides the n-dimension space into L *class regions*, denoted as $R_1, R_2, \cdots, R_L$. Any data point in the region R_i will be predicted to have label i by the classifier.

2.2 Evasion Attacks

Poisoning attacks and *evasion attacks* [7] are two well-known attacks to machine learning/data mining. A poisoning attack aims to pollute the training dataset such that the learner produces a bad classifier. Various studies have demonstrated poisoning attacks to spam filter [16], support vector machines [1], deep neural networks [20], and recommender systems [10, 24]. In an evasion attack, an attacker adds a small noise to a normal testing example (we call it *benign example*) such that a classifier predicts an incorrect label for the example with noise. A testing example with noise is called *adversarial example*. From a perspective of geometrics, an evasion attack moves a testing example from one class region to another.

In this work, we focus on DNNs and evasion attacks. A number of recent studies [2, 3, 11, 15, 17, 18, 22] have demonstrated that DNNs are vulnerable to evasion attacks at the testing phase. We denote by C a DNN classifier. $C(x)$ is the predicted label of a testing example $x \in [0, 1]^n$. Note that we assume each dimension of x is normalized to be in the range $[0, 1]$, like previous studies [2, 15]. An evasion attack adds noise δ to a benign example x such that the adversarial example $x + \delta$ is predicted to have a label t by the classifier C. L_0, L_2, and L_∞ norms are often used as the metric to measure the noise δ. Specifically, L_0 norm is the number of dimensions of x that are changed, i.e., the number of non-zero dimensions of δ; L_2 norm is the standard Euclidean distance between x and $x + \delta$; and L_∞ norm is the maximum change to any dimension of x, i.e., $\max\{\delta_1, \delta_2, \cdots, \delta_n\}$.

Carlini and Wagner [2] recently proposed a family of *targeted* evasion attacks, which achieve state-of-the-art attack performance. They concluded that all defense methods should be evaluated against their attacks. Therefore, in this work, we focus on their attacks. They didn't name their attacks. For simplicity, we call their attacks Carlini and Wagner (CW) attacks. Table 1 summarizes different versions of their attacks. Like other existing evasion attacks to DNN, CW attacks require that the DNN outputs are *differentiable* with respect to the input example. When attacking a classifier that is non-differentiable or the classifier parameters are unknown (i.e., black-box setting), an attacker needs to generate adversarial examples with respect to an auxiliary classifier and the adversarial examples are likely to also evade the target classifier.

CW-L_2 attack [2]: CW attacks have three variants that are tailored to the L_0, L_2, and L_∞ norms, respectively. The variant CW-L_2 attack is tailored to find adversarial examples with small noises measured by L_2 norm. Formally, the evasion attack solves the following optimization problem:

$$\min ||\frac{1}{2}(\tanh(w) + 1) - x||_2^2 + c \times f(\frac{1}{2}(\tanh(w) + 1)), \quad (1)$$

where $f(x') = \max(\max\{Z_i(x') : i \neq t\} - Z_t(x'), 0)$. t is the target label the attacker wants for the adversarial example. The adversarial example is $\frac{1}{2}(\tanh(w) + 1)$, which automatically constrains each dimension to be in the range [0,1]. The noise δ is $\delta = \frac{1}{2}(\tanh(w) + 1) - x$. CW-$L_2$ iterates over the parameter c via binary search in a relatively large range of candidate values. For each given c, CW-L_2 uses the Adam optimizer [8] to solve the optimization problem in Equation 1 to find the noise. The iterative process is halted at the smallest parameter c that the classifier predicts the target label t for the adversary example.

CW-L_0 attack [2]: This variant is tailored to find adversarial examples with small noises measured by L_0 norm. This attack iteratively identifies the dimensions of x that do not have much impact on the classifier's prediction and fixes them. The set of fixed dimensions increases until the attack has identified a minimal subset of dimensions that can be changed to construct a successful adversarial example. In each iteration, the set of dimensions that can be fixed are identified by the CW-L_2 attack. Specifically, in each iteration, CW-L_0 calls CW-L_2, which can only modify the unfixed dimensions. Suppose δ is the found noise for the benign example x. CW-L_0 computes the gradient $g = \Delta f(x + \delta)$ and selects the dimension $i = \text{argmin}_i\, g_i \times \delta_i$ to be fixed. The iterative process is repeated until CW-L_2 cannot find a successful adversarial example. Again, the parameter in CW-L_2 is selected via a searching process: starting from a very small c value; if CW-L_2 fails, then doubling c until finding a successful adversarial example.

CW-L_∞ attack [2]: This variant is tailored to find adversarial examples with small noises measured by L_∞ norm. Formally, the evasion attack aims to solve the following optimization problem to find the adversarial example:

$$\min \sum_i (\delta_i - \tau)^+ + c \times f(x + \delta), \quad (2)$$

where f is the same function as in CW-L_2; $(\delta_i - \tau)^+ = 0$ if $\delta_i < \tau$, otherwise $(\delta_i - \tau)^+ = \delta_i - \tau$. CW-$L_\infty$ iterates over c until finding a successful adversarial example. Specifically, c is iteratively doubled from a small value. For each given c, CW-L_∞ further iterates over τ. In particular, τ is initialized to be 1. For a given τ, CW-L_∞ solves the optimization problem in Equation 2. If $\delta_i < \tau$ for every i, then τ is reduced by a factor of 0.9, and then CW-L_∞ solves the optimization

problem with the updated τ. This process is repeated until such a noise vector δ that $\delta_i < \tau$ for every i cannot be found.

Success rate: An adversarial example is successful if it satisfies two conditions: 1) the adversarial example and the original benign example have the same true label (determined by human) and 2) the classifier predicts the target label t for the adversarial example. It is unclear how to check the first condition automatically because we do not have a way to model human perception yet. Therefore, existing studies leveraged L_0, L_2, or L_∞ norms to measure the noises added to the adversarial examples; and if the noises are small enough, an adversarial example is *assumed* to satisfy the first condition. However, it is still an open question on how to define "small enough" noises. Ideally, we aim to find a noise threshold; adversarial examples whose noises are less than the threshold do not change the true label. Whether there exists such a noise threshold and what the threshold is are still open questions and are valuable directions for future works.

In principle, *success rate* of a targeted evasion attack should be the fraction of its generated adversarial examples that satisfy both conditions. However, due to the challenges of checking the first condition, existing studies approximate success rate of an attack as the fraction of its generated adversarial examples that satisfy the second condition alone. For attacks that add very small noises to adversarial examples, the approximate success rates are close to the real success rates. In this work, we will focus on the approximate success rates and will call them success rates for simplicity.

2.3 Defenses Against Evasion Attacks

Roughly speaking, existing defenses against evasion attacks include designing new methods to train DNNs and defense in depth.

New methods to train DNNs: Goodfellow et al. [3] proposed to train a DNN via augmenting the training dataset with adversarial examples, which is called *adversarial training*. Specifically, for each training benign example, the learner generates a training adversarial example using evasion attacks. Then, the learner uses a standard algorithm (e.g., back propagation) to learn a DNN using the original training benign examples and the adversarial examples. However, as we will demonstrate in our experiments, adversarial training is not robust to powerful evasion attacks and adversarial examples that are unseen during training. In particular, the CW evasion attacks can still achieve 100% success rates at generating adversarial examples.

Papernot et al. [19] proposed a distillation based method to train a DNN. The DNN is first trained using a standard method. For each training example, the DNN produces a vector of confidence scores. The confidence scores are treated as the soft label for the training example. Given the soft labels and the training examples, the weights of the DNN are retrained. A parameter T named *distillation temperature* is used in softmax layer during both training sessions to control confidence scores. A DNN that is trained via distillation can significantly reduce the success rates of the evasion attacks also proposed by Papernot et al. [18]. However, Carlini and Wagner [2] demonstrated that their attacks CW can still achieve 100% success rates for DNNs trained with distillation. Moreover, the noises added to the benign examples when generating adversarial examples are just slightly higher for distilled DNNs than those for undistilled DNNs. Our experimental results confirm such findings.

Defense in depth: Meng and Chen proposed MagNet [12], a defense-in-depth approach to defend against evasion attacks to DNNs. Specifically, given a testing example, they first use a *detector* to determine whether the testing example is an adversarial example or not. If the testing example is predicted to be an adversarial example, the DNN classifier will not predict its label. If the testing example is not predicted to be an adversarial example, they will reform the testing example using a *reformer*. In the end, the DNN classifier will predict label of the reformed testing example and treat it as the label of the original testing example. MagNet designs both the detector and the reformer using auto-encoders, which are trained using only benign examples. Meng and Chen demonstrated that MagNet can reduce the success rates of various known evasion attacks. However, MagNet has two key limitations. First, MagNet decreases the classification accuracy on benign testing examples. For instance, on the CIFAR-10 dataset, their trained point-based DNN achieves an accuracy of 90.6%. However, MagNet reduces the accuracy to be 86.8% using the same point-based DNN. Second, it is unclear how to handle the testing examples that are predicted to be adversarial examples by the detector. We suspect that those testing examples eventually would require human to manually label them, i.e., the entire system becomes a human-in-the-loop system, losing the benefits of automated decision making. We note that the second limitation also applies to other approaches [4, 13] that aim to detect adversarial examples.

3 DESIGN GOALS

We aim to achieve the following two goals:

1) Not sacrificing classification accuracy on testing benign examples. Our first design goal is that the defense method should maintain the high accuracy of the DNN classifier on testing benign examples. Neural networks re-gained unprecedented attention in the past several years under the coat of "deep learning". The major reason is that neural networks with multiple layers (i.e., DNN) achieve significantly better classification accuracy than other machine learning methods for a variety of artificial intelligence tasks such as computer vision, speech recognition, and natural language processing. Therefore, our defense method should maintain such advantage of DNNs.

2) Increasing robustness. We aim to design a defense method that is robust to powerful evasion attacks. In particular, our new classifier should have better robustness than conventional DNN classifiers with respect to state-of-the-art evasion attacks, i.e., the CW attacks. Suppose we have a DNN classifier C. After deploying a certain defense method, we obtain another classifier D. Suppose we have an evasion attack. The success rate (SR) of the attack for the classifiers C and D are denoted as SR_C and SR_D, respectively. We say that the classifier D is more robust than the classifier C if $SR_D < SR_C$. In other words, a defense method is said to be effective with respect to an evasion attack if the defense method decreases the attack's success rate.

We note that our goal is not to completely eliminate adversarial examples. Instead, our goal is to reduce attackers' success rates without sacrificing classification accuracy on benign examples.

4 MEASURING EVASION ATTACKS

We first show some measurement results on evasion attacks, which motivate the design of our region-based classification method. We performed our measurements on the standard MNIST dataset. In the dataset, our task is to recognize the digit in an image, which is a 10-class classification problem. We normalize each pixel to be in the range [0,1]. We adopted the same DNN classifier that was used by Carlini and Wagner [2]. The classifier essentially classifies the digit image space into 10 class regions, denoted as $R_0, R_1, \cdots, R_9$. Any data point in the class region R_i will be predicted to have label i by the classifier.

We sample a benign testing image of digit 0 uniformly at random. We use the CW-L_2, CW-L_0, and CW-L_∞ attacks to generate adversarial examples based on the sampled benign example. We obtained the open-source implementation of the CW attacks from its authors [2]. For each target label i, we use an evasion attack to generate an adversarial example with the target label i based on the benign example, where $i = 1, 2, \cdots, 9$. We denote the adversarial example with the target label i as $x'(i)$. The DNN classifier predicts label i for the adversarial example $x'(i)$, while its true label is 0.

We denote the *hypercube* that is centered at x and has a length of r as $B(x, r)$. Formally, $B(x, r) = \{y | y_j \in [0, 1] \text{ and } |y_j - x_j| \leq r, \forall j = 1, 2, \cdots, n\}$, where x_j and y_j are the jth dimensions of x and y, respectively. For each adversarial example $x'(i)$, we sample 10,000 data points from the hypercube $B(x'(i), r)$ uniformly at random, where we set $r = 0.3$ in our experiments (we will explain the setting of r in experiments). We treat each data point as a testing example and feed it to the DNN classifier, which predicts a label for it. For the 10,000 data points, we obtain a histogram of their labels predicted by the DNN classifier.

Figure 6a, Figure 6b, and Figure 6c (the figures are shown at the end of the paper) show the label histograms for the 10,000 randomly sampled data points from the hypercube around the benign example and the 9 adversarial examples generated by the CW-L_2 attack, CW-L_0 attack, and CW-L_∞ attack, respectively. For instance, in Figure 6a, the first graph in the first row shows the histogram of labels for the 10,000 data points that are sampled from the hypercube centered at the benign example; the second graph (from left to right) in the first row shows the histogram of labels for the 10,000 data points that are sampled from the hypercube centered at the adversarial example that has a predicted label 1, where the adversarial example is generated by the CW-L_2 attack.

For the benign example, almost all the 10,000 randomly sampled data points are predicted to have label 0, which is the true label of the benign example. For most adversarial examples, a majority of the 10,000 randomly sampled data points are predicted to have label 0, which is the true label of the adversarial examples. From these measurement results, we have the following two observations:

- **Observation I:** The hypercube $B(x, r)$ centered at a benign example x intersects the most with the class region R_i, where i is the true label of the benign example x. This indicates that we can still correctly predict labels for benign examples by ensembling information in the hypercube.
- **Observation II:** For most adversarial examples, the hypercube $B(x', r)$ intersects the most with the class region R_i, where i is the true label of the adversarial example x'. This indicates that we can also correctly predict labels for adversarial examples by ensembling information in the hypercube.

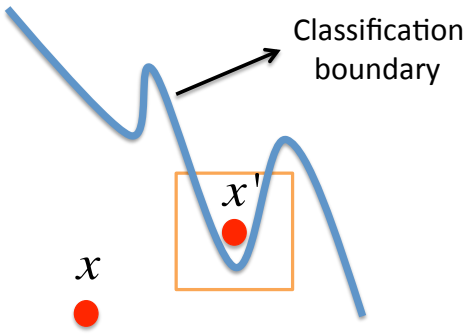


Figure 2: Illustration of our region-based classification. x is a testing benign example and x' is the corresponding adversarial example. The hypercube centered at x' intersects the most with the class region that has the true label.

These measurement results motivate us to design our region-based classification, which we will introduce in the next section.

5 OUR REGION-BASED CLASSIFICATION

We propose a defense method called *Region-based Classification* (RC). Traditional DNN classifier is *point-based*, i.e., given a testing example, the DNN classifier predicts its label. Therefore, we call such a classifier Point-based Classification (PC). In our RC classification, given a testing example, we ensemble information in the region around the testing example to predict its label. For any point-based DNN classifier, our method can transform it to be a region-based classifier that is more robust to adversarial examples, while maintaining its accuracy on benign examples.

5.1 Region-based Classification

Suppose we have a point-based DNN classifier C. For a testing example x (either benign example or adversarial example), we create a hypercube $B(x, r)$ around the testing example. Recall that the DNN classifier essentially divides the input space into L class regions, denoted as $R_1, R_2, \cdots, R_L$; all data points in the class region R_i are predicted to have label i by the classifier, where $i = 1, 2, \cdots, L$. In our RC classifier, we predict the label of a testing example x to be the one whose class region intersects the most with the hypercube $B(x, r)$. Formally, we denote our RC classifier as $RC_{C,r}$ since it relies on the point-based DNN classifier C and the length r. We denote the area of the intersection between R_i and $B(x, r)$ as $A_i(x, r)$. Then, our RC classifier predicts the label of x to be $RC_{C,r}(x) = \text{argmax}_i A_i(x, r)$. Figure 2 illustrates our region-based classification.

Approximating the areas $A_i(x, r)$: One challenge of using our RC classifier is how to compute the areas $A_i(x, r)$, because the class regions might be very irregular. We address the challenge via sampling m data points from the hypercube $B(x, r)$ uniformly at random and use them to approximate the areas $A_i(x, r)$. In particular, for each sampled data point, we use the point-based

Algorithm 1 Learning Length r by Searching

Input: Validation dataset V, point-based DNN classifier C, step size ϵ, initial length r_0.
Output: Length r.

Initialize $r = r_0$.
ACC = Accuracy of C on V.
ACC_{RC} = Accuracy of the $RC_{C,r}$ classifier on V.
while $ACC_{RC} \geq ACC$ **do**
  $r = r + \epsilon$.
  ACC_{RC} = Accuracy of the $RC_{C,r}$ classifier on V.
end while
return $r - \epsilon$.

classifier C to predict its label. We denote by $a_i(x, r)$ the number of sampled data points that are predicted to have label i by the classifier C. Then, our RC classifier predicts the label of x as $RC_{C,r}(x) = \text{argmax}_i\, a_i(x, r)$.

Learning the length r: Another challenge for our RC classifier is how to determine the length r of the hypercube. r is a critical parameter for our method RC (we will show the impact of r on the effectiveness of RC in our experiments). Specifically, r controls the tradeoff between robustness to adversarial examples and classification accuracy on benign examples. Suppose we want to classify an adversarial example x', whose true label is i. On one hand, if the length of the hypercube $B(x', r)$ is too small, the hypercube will not intersect with the class region R_i, which means that our RC classifier will not be able to correctly classify the adversarial example. On the other hand, if the length is too large, the hypercube around a benign example will intersect with the incorrect class regions, which makes our method predict incorrect labels for benign examples.

To address the challenge, we propose to learn the length r using a validation dataset consisting of only benign examples. We do not use adversarial examples because the adversarial examples used by the attacker may not be accessible to the defender. Our key idea is to select the maximal length r such that the classification accuracy of our classifier $RC_{C,r}$ on the validation dataset is no smaller than that of the point-based classifier C. There are many choices of r, with which our classifier $RC_{C,r}$ has no smaller classification accuracy than the point-based classifier C. We propose to select the maximum one, so an adversarial example needs a larger noise to move further away from the classification boundary of C in order to evade $RC_{C,r}$.

Specifically, we learn the radius through a search process. Suppose a point-based DNN classifier C has classification accuracy ACC on the validation dataset. We transform the classifier C into a RC classifier. Initially, we set r to be a small value. For each benign example in the validation dataset, we predict its label using our classifier $RC_{C,r}$. We compute the classification accuracy of the classifier $RC_{C,r}$ on the validation dataset. If the classification accuracy is no smaller than ACC, we increase the radius r by a *step size* ϵ and repeat the process. This search process is repeated until the classifier $RC_{C,r}$ achieves a classification accuracy on the validation dataset that is smaller than ACC. Algorithm 1 shows the search process.

5.2 Evasion Attacks to Our RC Classifier

We consider a strong attacker who knows all the model parameters of our classifier $RC_{C,r}$. In particular, the attacker knows the architecture and parameters of the point-based DNN classifier C, the length r, and m, the number of data points sampled to approximate the areas. Our threat model is also known as the *white-box setting*.

5.2.1 Existing evasion attacks. An attacker can use any attack shown in Table 1 to find adversarial examples to evade our classifier $RC_{C,r}$. All state-of-the-art evasion attacks to DNN classifiers require the classifier to be differentiable, in order to propagate the gradient flow from the outputs to the inputs. However, our classifier $RC_{C,r}$ is *non-differentiable*. Therefore, we consider an attacker generates adversarial examples based on the point-based classifier C, which is the key component of our classifier $RC_{C,r}$; and the attacker uses the adversarial examples to attack $RC_{C,r}$. This is also known as transferring adversarial examples from one classifier to another.

We note that Carlini and Wagner [2] proposed to adjust their CW-L_2 attack to generate high-confidence transferable adversarial examples, which are more likely to transfer from one classifier to another. However, Meng and Chen [12] demonstrated that such high-confidence transferable adversarial examples can be easily detected. Therefore, we do not consider high-confidence transferable adversarial examples in our work.

5.2.2 New evasion attacks. An attacker, who knows our region-based classification, can also strategically adjust its attacks. Specifically, since our classifier ensembles information within a region, an attacker can first use an existing evasion attack to find an adversarial example based on the point-based classifier C and then strategically add more noise to the adversarial example. The goal is to move the adversarial example further away from the classification boundary such that the hypercube centered at the adversarial example does not intersect or intersects less with the class region that has the true label of the adversarial example.

Specifically, suppose we have a benign example x. The attacker uses an evasion attack shown in Table 1 to find the corresponding adversarial example x'. The added noise is $\delta = x' - x$. Then, the attacker strategically constructs another adversarial example as $x'' = x + (1 + \alpha)\delta$. Essentially, the attacker moves the adversarial example further along the direction of the current noise. Note that, we will clip the adversarial example x'' to be in the space $[0, 1]^n$. Specifically, for each dimension i of x'', we set $x''_i = 0$ if $x''_i < 0$, we $x''_i = 1$ if $x''_i > 1$, and x''_i keeps unchanged if $0 < x''_i < 1$. The parameter α controls how much further to move the adversarial example away from the classification boundary. For L_2 and L_∞ norms, α is the increased fraction of noise. Specifically, suppose an evasion attack shown in Table 1 finds an adversarial example x' with noise δ, whose L_2 and L_∞ norms are $||\delta||_2$ and $||\delta||_\infty$, respectively. Then, the adapted adversarial example x'' has noise $(1+\alpha)\delta$, whose L_2 and L_∞ norms are $(1 + \alpha)||\delta||_2$ and $(1 + \alpha)||\delta||_\infty$, respectively. A larger α indicates a larger noise (for L_2 and L_∞ norms) and a possibly larger success rate.

For convenience, for an evasion attack, we append the suffix *-A* at the end of the attack's name to indicate the attack that is adapted to our classifier $RC_{C,r}$. For instance, CW-L_0-A means the adapted

Table 2: Dataset statistics.

	Training	Validation	Testing
MNIST	55,000	5,000	10,000
CIFAR-10	45,000	5,000	10,000

version of the attack CW-L_0. In our experiments, we will explore how α impacts the success rates of the adapted evasion attacks and noises added to the adversarial examples.

6 EVALUATIONS

6.1 Experimental Setup

Datasets: We perform evaluations on two standard image datasets used to benchmark object recognition methods: MNIST and CIFAR-10. Table 2 shows the statistics of the datasets. For each dataset, we sample 5,000 of the predefined training examples uniformly at random and treat them as the validation dataset used to learn the length r in our RC classifier.

Compared methods: We compare the following DNN classifiers.

- **Standard point-based DNN.** For each dataset, we trained a standard point-based DNN classifier. For the MNIST dataset, we adopt the same DNN architecture as the one adopted by Carlini and Wagner [2]. For the CIFAR-10 dataset, the DNN architecture adopted by Carlini and Wagner is not state-of-the-art. Therefore, we do not adopt their DNN architecture for the CIFAR-10 dataset. Instead, we use the DNN architecture proposed by He et al. [5]. We obtained implementation from Carlini and Wagner to train the DNN for MNIST; and we obtained the implementation from [23] to train the DNN for CIFAR-10.
- **Adversarial training DNN.** For each dataset, we use adversarial training [3] to learn a DNN classifier. The DNN classifiers have the same architectures as the standard point-based DNNs. We note that CW attacks are inefficient to generate adversarial examples. Specifically, among the three evasion attacks, CW-L_2 is the most efficient one, but it still requires 15-20 days to generate adversarial examples for all training examples in the MNIST dataset on our machine. The CIFAR-10 dataset even takes a much longer time to generate adversarial examples for all training examples. Therefore, we use DeepFool [15], a less powerful but orders of magnitude more efficient evasion attack to generate adversarial examples for each training example.
- **Distillation DNN.** For each standard point-based DNN classifier, we use distillation [19] to re-train the DNN classifier with a temperature $T = 100$.
- **Our region-based DNN.** For each dataset, we transform the corresponding standard point-based DNN classifier to our region-based DNN classifier. The length r is learnt through our Algorithm 1 using the validation dataset. Specifically, we set the initial length value r_0 and step size ϵ in Algorithm 1 to be 0 and 0.01, respectively. Figure 3 shows the classification accuracy of our RC classifier on the MNIST validation dataset as we increase the length r in Algorithm 1. We observe that our classifier $RC_{C,r}$ has slightly higher accuracies than the standard point-based classifier C when r is small. Moreover, when r is larger than around 0.3, accuracy of $RC_{C,r}$ starts to decrease. Therefore, according to Algorithm 1, the length r is set to be 0.3 for the MNIST dataset. Moreover, via Algorithm 1, the length r is set to be 0.02 for the CIFAR-10 dataset. To estimate the areas between a hypercube and class regions, we sample 1,000 data points from the hypercube, i.e., the parameter m is set to be 1,000.

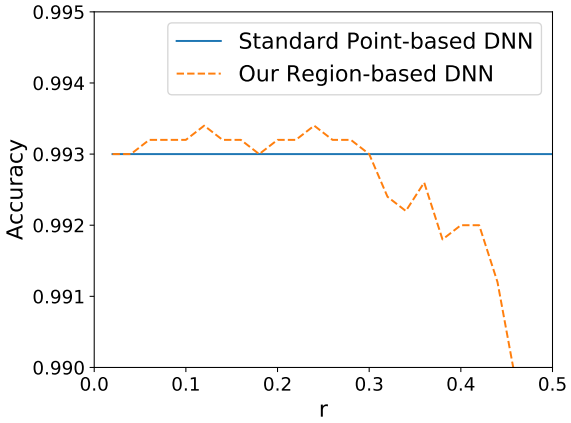


Figure 3: Classification accuracies of the standard point-based DNN and our region-based DNN on the MNIST validation dataset as we increase the length r.

6.2 Results

Classification accuracies: Table 3 and Table 4 show the classification accuracies of the compared classifiers for the MNIST and CIFAR-10 datasets, respectively. First, our region-based DNN achieves the same classification accuracy on the testing dataset with the standard point-based DNN for both the MNIST and CIFAR-10 datasets. Second, adversarial training DNN and distillation DNN achieve lower classification accuracies than standard point-based DNN, though the differences are smaller for the MNIST dataset.

Robustness to existing state-of-the-art evasion attacks: Table 3 and Table 4 show the success rates of the state-of-the-art evasion attacks CW-L_0, CW-L_2, and CW-L_∞ for different DNN classifiers on the MNIST and CIFAR-10 datasets. Since the CW attacks are inefficient, for each dataset, we randomly sample 100 testing benign examples that the standard point-based DNN correctly classifies and generate adversarial examples for them. First, we observe that adversarial training and distillation are not effective to mitigate state-of-the-art evasion attacks. Specifically, the CW attacks still achieve 100% success rates for adversarial training and distillation DNNs on both the MINIST and CIFAR-10 datasets.

Second, our region-based DNN can substantially reduce success rates of the CW attacks. Specifically, for the MNIST dataset, our region-based DNN reduces success rates of CW-L_0, CW-L_2, and CW-L_∞ to be 16%, 0%, and 0%, respectively; for the CIFAR-10 dataset, our

Table 3: Classification accuracy on benign examples and robustness to CW attacks on the MNIST dataset.

	Classification Accuracy	Success Rate		
		CW-L_0	CW-L_2	CW-L_∞
Standard point-based DNN	99.4%	100%	100%	100%
Adversarial training DNN	99.3%	100%	100%	100%
Distillation DNN	99.2%	100%	100%	100%
Our region-based DNN	99.4%	16%	0%	0%

Table 4: Classification accuracy on benign examples and robustness to CW attacks on the CIFAR-10 dataset.

	Classification Accuracy	Success Rate		
		CW-L_0	CW-L_2	CW-L_∞
Standard point-based DNN	90.1%	100%	100%	100%
Adversarial training DNN	88.1%	100%	100%	100%
Distillation DNN	88.3%	100%	100%	100%
Our region-based DNN	90.1%	7%	2%	6%

region-based DNN reduces success rates of CW-L_0, CW-L_2, and CW-L_∞ to be 7%, 2%, and 6%, respectively. For our region-based DNN classifier, CW-L_0 achieves the largest success rates across the two datasets. We speculate the reason is that CW-L_0 modifies a small number of pixels of a benign example, but it can change the pixel values substantially. As a result, the adversarial examples generated by CW-L_0 are further away from the classification boundary and thus are more likely to evade our region-based DNN.

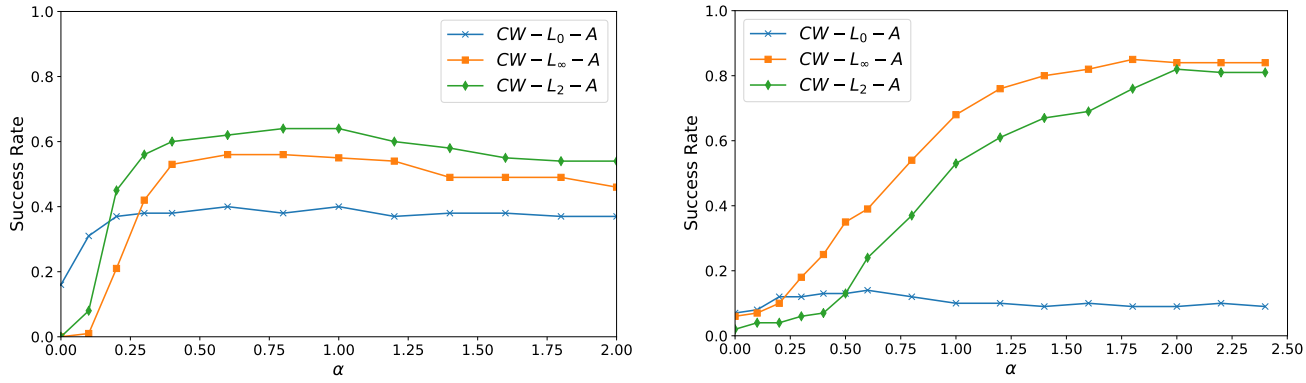
Robustness to new evasion attacks: Recall that we discussed adapting CW attacks to our region-based DNN in Section 5.2. The key idea is to move the adversarial example further away from the classification boundary. The parameter α controls the tradeoff between the increased fraction of noise (for L_2 and L_∞ norms) and success rates. Figure 4 shows such tradeoffs.

The adapted CW attacks cannot achieve 100% success rates anymore no matter how we set the parameter α. Specifically, the success rates first increase and then decrease as α increases. This is because adding too much noises to adversarial examples moves them to other class regions, resulting in an unsuccessful targeted evasion attack. Suppose an adversarial example has a target label i. The original adversarial example generated by a CW attack is in the class region R_i. When α is small, the adapted adversarial example generated by an adapted CW attack is still within the class region R_i. However, when α is large, the adapted adversarial example is moved to be in another class region R_j, which has a different label.

For the MNIST dataset, the best evasion attack is the adapted attack CW-L_2-A. The largest success rate the attack can achieve is 64%, when $\alpha = 1$, i.e., the added average noise of adversarial examples is doubled. When the attack CW-L_2-A wants to achieve 50% success rate, the adversarial examples need 25% more noises. For the CIFAR-10 dataset, the best adapted evasion attack is CW-L_∞-A. The attack achieves 85% success rates when $\alpha = 1.8$. The attack needs to set $\alpha = 0.75$ in order to achieve a 50% success rate.

Figure 5 shows the adversarial examples generated by the adapted evasion attack CW-L_2-A for the MNIST dataset when the attack achieves the highest success rate, i.e., $\alpha = 1$. For all these adversarial examples, our region-based DNN classifier predicts the target label for each of them. Recall that Figure 1 shows adversarial examples generated by the existing CW-L_2 attack. To compare the adversarial examples generated by CW-L_2 and CW-L_2-A, we use the same benign examples in Figure 5 and Figure 1. We observe that some adversarial examples generated by CW-L_2-A have changed the true labels. For instance, the sixth adversarial example in Figure 5 was generated from a benign example with true label 5. However, human can hardly classify the adversarial example to be a digit 5, i.e., the true label has been changed. Similarly, the third, eighth, and ninth adversarial examples almost change the true labels of the corresponding benign examples.

Recall that in Section 2.2, we discussed that a successful adversarial example should satisfy two conditions and we approximate success rate of an attack using its generated adversarial examples that satisfy the second condition only. Our results in Figure 5 show that some adversarial examples that satisfy the second condition do not satisfy the first condition, because of adding too much noises. Therefore, the real success rates of the adapted evasion attacks are even lower.



(a) MNIST

(b) CIFAR-10

Figure 4: Tradeoff between success rates and increased fraction of noises for adapted CW attacks.

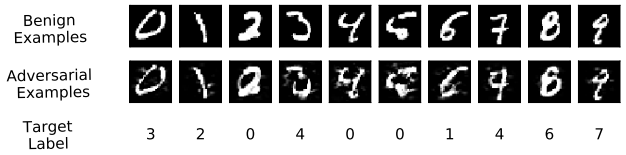


Figure 5: Adversarial examples generated by the adapted evasion attack CW-L_2-A for the MNIST dataset, where $\alpha = 1$.

7 CONCLUSION

In this work, we propose a region-based classification to mitigate evasion attacks to deep neural networks. First, we perform a measurement study about the adversarial examples generated by state-of-the-art evasion attacks. We observe that the adversarial examples are close to the classification boundary and the hypercube around an adversarial example significantly intersects with the class region that has the true label of the adversarial example. Second, based on our measurement study, we propose a region-based DNN classifier, which ensembles information in the hypercube around an example to predict its label. Third, we perform evaluations on the standard MNIST and CIFAR-10 datasets. Our results demonstrate that our region-based DNN classifier can significantly reduce success rates of both state-of-the-art evasion attacks and the evasion attacks that are strategically adapted to our region-based DNN classifier, without sacrificing classification accuracy on benign examples.

We encourage researchers who propose new evasion attacks to evaluate their attacks against our region-based classifier, instead of standard point-based classifier only.

REFERENCES

[1] Battista Biggio, Blaine Nelson, and Pavel Laskov. 2012. Poisoning attacks against support vector machines. In *ICML*.

[2] Nicholas Carlini and David Wagner. 2017. Towards Evaluating the Robustness of Neural Networks. In *IEEE S & P*.

[3] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. In *arXiv*.

[4] Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick McDaniel. 2017. On the (statistical) detection of adversarial examples. In *arXiv*.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *CVPR*.

[6] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, and others. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine* 29, 6 (2012), 82–97.

[7] Ling Huang, Anthony D Joseph, Blaine Nelson, Benjamin IP Rubinstein, and JD Tygar. 2011. Adversarial machine learning. In *ACM AISec*.

[8] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. In *arXiv*.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *NIPS*.

[10] Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. 2016. Data Poisoning Attacks on Factorization-Based Collaborative Filtering. In *NIPS*.

[11] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. 2017. Delving into Transferable Adversarial Examples and Black-box Attacks. In *ICLR*.

[12] Dongyu Meng and Hao Chen. 2017. MagNet: a Two-Pronged Defense against Adversarial Examples. In *CCS*.

[13] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischof. 2017. On detecting adversarial perturbations. In *International Conference on Learning Representations (ICLR)*.

[14] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[15] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. DeepFool: a simple and accurate method to fool deep neural networks. In *CVPR*.

[16] B. Nelson, M. Barreno, F. J. Chi, A. D. Joseph, B. I. P. Rubinstein, U. Saini, C. Sutton, J. D. Tygar, and K. Xia. 2008. Exploiting machine learning to subvert your spam filter. In *LEET*.

[17] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In *AsiaCCS*.

[18] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. 2016. The Limitations of Deep Learning in Adversarial Settings. In *EuroS&P*.

[19] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2016. Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks. In *IEEE S & P*.

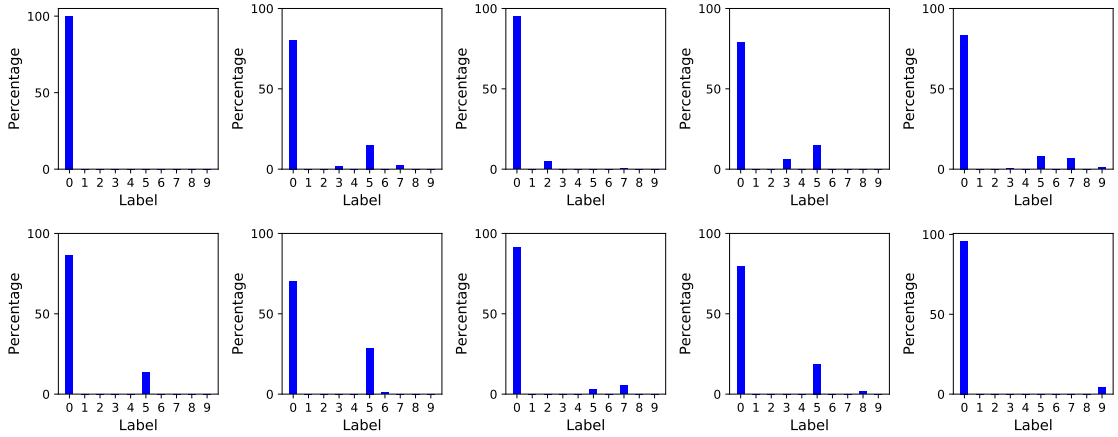
[20] Shiqi Shen, Shruti Tople, and Prateek Saxena. 2016. AUROR: Defending Against Poisoning Attacks in Collaborative Deep Learning Systems. In *ACSAC*.

[21] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, and others. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature* 529, 7587 (2016), 484–489.

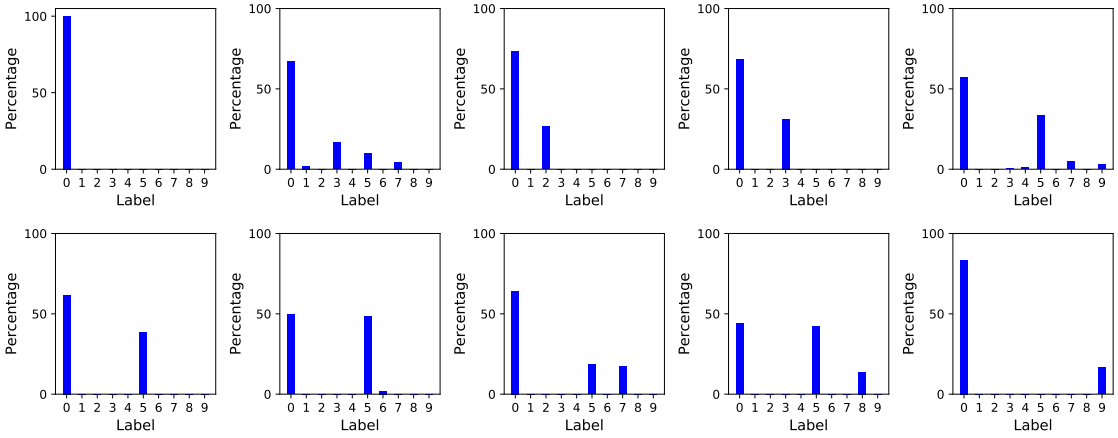
[22] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. In *arXiv*.

[23] Code to Train DNN for CIFAR-10. 2017. (September 2017). https://goo.gl/mEX7By

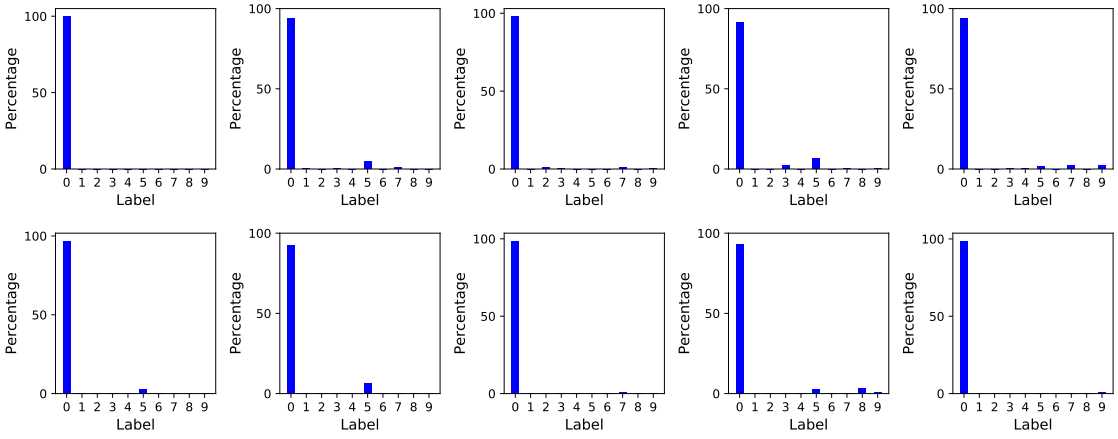
[24] Guolei Yang, Neil Zhenqiang Gong, and Ying Cai. 2017. Fake Co-visitation Injection Attacks to Recommender Systems. In *NDSS*.



(a) CW-L_2 attack



(b) CW-L_0 attack



(c) CW-L_∞ attack

Figure 6: Label histograms of 10,000 random data points in the hypercube around a benign example or its adversarial examples generated by the (a) CW-L_2 attack, (b) CW-L_0 attack, and (c) CW-L_∞ attack. Each histogram corresponds to an example. The benign example has label 0. In each subfigure, the first row (from left to right): the benign example, and the adversarial examples that have target labels 1, 2, 3, and 4, respectively; and the second row (from left to right): the adversarial examples that have target labels 5, 6, 7, 8, and 9, respectively.