A Learning Approach to Secure Learning

Linh Nguyen, Arunesh Sinha
University of Michigan, Ann Arbor
{lvnguyen,arunesh}@umich.edu

arXiv:1709.04447v1 [cs.CR] 13 Sep 2017

Abstract

Deep Neural Networks (DNNs) have been shown to be vulnerable against adversarial examples, which are data points cleverly constructed to fool the classifier. Such attacks can be devastating in practice, especially as DNNs are being applied to ever increasing critical tasks like image recognition in autonomous driving. In this paper, we introduce a new perspective on the problem. We do so by first defining robustness of a classifier to adversarial exploitation. Next, we show that the problem of adversarial example generation and defense both can be posed as learning problems, which are duals of each other. We also show formally that our defense aims to increase robustness of the classifier. We demonstrate the efficacy of our techniques by experimenting with the MNIST and CIFAR-10 datasets.

1 Introduction

Recent advances in deep learning have led to its wide adoption in various challenging tasks such as image classification. However, the current state of the art has been shown to be vulnerable to *adversarial examples*, small perturbations of the original inputs, often indistinguishable to a human, but carefully crafted to misguide the learning models into producing incorrect outputs. Recent results have shown that generating these adversarial examples are inexpensive (Moosavi-Dezfooli, Fawzi, and Frossard 2015; Szegedy et al. 2013b; Goodfellow, Shlens, and Szegedy 2014). Moreover, as safety critical applications such as autonomous driving increasingly rely on these tasks, it is imperative that the learning models be reliable and secure against such adversarial examples.

Prior work has yielded a lot of attack methods that generate adversarial samples, and defense techniques that improve the accuracy on these samples (see related work for details). However, defenses are often specific to certain attacks and cannot adaptively defend against any future attack and some general defense techniques have been shown to be ineffective against more powerful novel attacks. More generally, attacks and defenses have followed the cat-and-mouse game that is typical of many security settings.

In this paper, with the goal of generality, we pose both attack and defense in the context of adversarial examples as a *learning* problem. That is, an attack that can learn to generate adversarial examples against a given classifier and, given a *new* attack, a defense technique that can learn to defend against that attack. While our guiding principle is general, this paper focuses on the specific domain of adversarial examples in image classification.

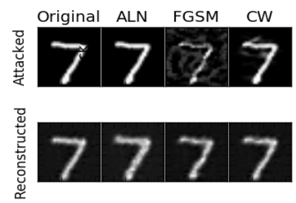

Figure 1: Samples of ALN attack and DLN sanitization for different attacks using MNIST dataset

Our *first contribution* is a definition of *robustness of classifiers*. Towards the definition, we define the exploitable space by the adversary which includes data points already mis-classified (errors) of any given classifier and any data points that can be perturbed to force mis-classifications. Robustness is defined as the probability of data points occurring in the exploitable space. We believe our definition captures the essence of the defender-adversary interaction, and is natural as our robustness is a strictly stronger concept than accuracy.

Our *second contribution* is an *attack learning neural network* (ALN). ALN is motivated by the fact that adversarial examples for a given classifier C are subsets of the input space that the C mis-classifies. Thus, given a data distribution with data points x and a classifier C trained on such data, we train a feed forward neural network A with the goal of generating output points $A(x)$ in the mis-classified space. Towards this end, we re-purpose an autoencoder to work as our ALN A with a special choice of loss function that aims to make (1) the classifier C mis-classify $A(x)$ and (2) minimize the difference between x and $A(x)$. A sample ALN attack is shown in the top row and second column of Figure 1. Corresponding attacks using FGSM (Goodfellow, Shlens, and Szegedy 2014) and CW (Carlini and Wagner 2017b) attacks are also shown.

Our *third contribution* is a *defense learning neural network* (DLN). Following the motivation and design of ALN, we motivate DLN D as a neural network that, given any classifier C attacked by an attack technique A, takes in an adversarial example $A(x)$ and aims to generate benign example $D(A(x))$ that *does not* lie in the mis-classified space of C. The DLN also ensures that if x is not an adversarial example then $D(x)$ is almost same as x. The DLN is prepended to the classifier C acting as a sanitizer for C. Again, similar to the ALN, we re-purpose an autoencoder with a special choice of loss function that aims to revert back any adversarial example $A(x)$ to the original data point x with minimal difference between $A(x)$ and x. For non-adversarial inputs the DLN is encouraged to reproduce the input as well as make the classifier predict correctly. A sample DLN undoing three attacks is shown in the bottom row of Figure 1. Further, we show that DLN can be trained to simultaneously defend against a set of attacks $\mathcal{A}$. This greatly improves the applicability of the DLN based approach as defending against multiple attacks allows the resultant classifier to be resilient to known attacks, and any novel attack that shows up in future. We further show that DLN allows for attack and defense to be set up as a repeated competition leading to more robust classifiers.

We tested our approach on two datasets: MNIST and CIFAR-10. Our ALN based attack was able to attack all classifiers we considered and achieve performance comparable to other known attacks. Our defense approach made the resultant classifier robust to the attacks we consider: our attack ALN, FGSM and CW. Detailed experiments are presented in Section 4 and 5.

2 Attack Model

Given the adversarial setting, it is imperative to define the capabilities of the adversary, which we do in this section. First, we use *inference phase* of a classifier to mean the stage when the classifier is actually deployed as an application (after all training and testing is done). The attacker attacks *only* in the inference phase and can channel his attack *only* through the inputs. In particular, the attacker cannot change the classifier weights or inject any noise in the hidden layers. The attacker has access to the classifier weights, so that it can compute gradients if required. The attacker's goal is to produce adversarial data points that get mis-classified by the classifier. These adversarial examples should be legitimate (that is not a garbage noisy image) and the true class and the predicted class of the data point could be additional constraints for the adversary.

3 Approach

This section formally describes our approach to the adversarial example generation and defense problem using the notion of robustness we define. We start by defining basic notations. Let the function $C : X \to Y$ denote a classifier that takes input data points with feature values in X and outputs a label among the possible k labels $Y = \{1, \dots, k\}$. Further, for neural networks based classifiers we can define $C_p : X \to \Delta Y$ as the function that takes in data and produces a probability distribution over labels. Thus, $C = \max\{C_p(x)\}$, where $\max$ provides the maximum component of the vector $C_p(x)$. Let $sim(x, x')$ denote the similarity between x and x'. Let $H(p, q)$ denote the cross entropy $-\sum_i p_i \log(q_i)$. In particular, let $H(p)$ denotes the entropy given by $H(p, p)$. For this paper, we assume X is the set of legitimate images (and not garbage images or ambiguous images). Legitimate images are different for different domains, e.g., they are digits for digit classification.

3.1 Robustness

We first introduce some concepts from PAC learning (Anthony and Bartlett 2009), in order to present the formal results in this section. It is assumed that data points arise from a fixed but unknown distribution $\mathcal{P}$ over X. We denote the probability mass over a set $Z \subset X$ as $\mathcal{P}(Z)$. A loss function $l(y_x, C(x))$ captures the loss of predicting $C(x)$ when the true label for x is y_x. As we are focused on classification, we restrict ourselves to the ideal $0/1$ loss, that is, 1 for correct classification and 0 otherwise. A classifier C is chosen that minimizes the empirical loss over the n training data points $\sum_{i=1}^{n} l(y_{x_i}, x_i)$. Given enough data, PAC learning theory guarantees that C also minimizes the expected loss $\int_X l(y_x, C(x))\mathcal{P}(x)$. Given, $0/1$ loss this quantity is just $\mathcal{P}(M_C(X))$, where $M_C(X) \subset X$ denote the region where the classifier C mis-classifies. Accuracy for a classifier is then just $1 - \mathcal{P}(M_C(X))$. In this paper we will assume that the amount of data is always enough to obtain low expected loss. Observe that a classifier can achieve high accuracy (low expected loss) even though its prediction on the low probability regions may be wrong.

All classifier families have a capacity that limits the complexity of separators (hypothesis space) that they can model. A higher capacity classifier family can model more non-smooth separators[1]. Previous work (Goodfellow, Shlens, and Szegedy 2014) has conjectured that adversarial examples abound due to the low capacity of the classifier family used. See Figure 2A for an illustration.

Adversarial exploitable space: Define $E_{C,\epsilon}(X) = M_C(X) \cup \{x \mid sim(x, M_C(X)) \leq \epsilon\}$, where sim is a similarity measure that depends on the domain and $sim(x, M_C(X))$ denotes the similarity of x and the most similar data point to x in $M_C(X)$. For image classification sim can just be the l_2 (Euclidean) distance: $\sqrt{\sum_i (x_i - x_i')^2}$ where i indexes the pixels. $E_{C,\epsilon}(X)$ is the adversarial exploitable space, as this space includes all points that are either mis-classified or can be mis-classified by a minor ϵ-perturbation. Note that we assume that any already present mis-classifications of the classifier is exploitable by the adversary without the need of any perturbation. For example, if a stop sign image in a dataset is mis-classified then an adversary can simply use this image as is to fool an autonomously driven vehicle.

Robustness: Robustness is simply defined as $1 - \mathcal{P}(E_{C,\epsilon}(X))$. First, it is easy to see that robustness is a strictly stronger concept than accuracy, that is, a classifier with high robustness has higher accuracy. We believe this

[1] While capacity is defined for any function class (Anthony and Bartlett 2009) (includes deep neural networks), the value is known only for simple classifiers like single layered neural networks.

property makes our definition more natural than other current definitions. Further, another readily inferable property from the definition of $E_{C,\epsilon}$ that we utilize later is that a classifier C' with $M_{C'}(X) \subset M_C(X)$ is more robust than classifier C in the same setting. We call a classifier C' perfect if the robustness is 100%.

There are a number of subtle aspects of the definition that we elaborate upon below:

- A 100% robust classifier can still have $M_{C'}(X) \neq \phi$. This is because robustness is still defined w.r.t. $\mathcal{P}$, for example, large compact regions of zero probability with small sub-region of erroneous prediction far away from the boundary can still make robustness 100%. However, $M_{C'}(X) = \phi$ provides 100% robustness for any $\mathcal{P}$. Thus, robustness based on just $M_{C'}(X)$ and not $\mathcal{P}$ is a stronger but much more restrictive concept of robustness than ours.
- A perfect classifier (100% robust) is practically impossible due to large data requirement especially as the capacity of the classifier family grows. As shown in Figure 2 low capacity classifiers cannot model complex separators, thus, large capacity is required to achieve robustness. On the other hand, classifiers families with large capacity but not enough data tend to overfit the data (Anthony and Bartlett 2009). Thus, there is a delicate balance between the capacity of the classifier family used and amount of data available. The relation between amount of data and capacity is not very well understood for Dep Neural Networks. In any case, perfect robustness provides a goal that robust classifiers should aim to achieve. In this paper, for the purpose of defense, we seek to increase the robustness of classifiers.
- Robustness in practice may apparently seem to be computable by calculating the accuracy for the test set and the adversarially perturbed test set for any given dataset, which we also do and has been done in all prior work. However, this relies on the fact that the attack is all powerful, i.e., it can attack *all* perturb-able points. It is easy to construct abstract examples with probability measure zero mis-classification set (single mis-classified point in a continuous Euclidean space) that is computationally intractable for practical attacks to discover. A detailed analysis of computing robustness is beyond the scope of this paper and is left for future work.
- The definition can be easily extended to weigh some kinds of mis-classification more, if required. For example, predicting a malware as benign is more harmful than the opposite erroneous prediction. For our focus area of image classification in this paper, researchers have generally considered all mis-classification equally important. Also the sim function in the definition is reasonably well understood for images. Instantiating the definition for other domains such as malware classification requires exploring sim further such as how to capture that two malwares are functionally similar.

Lastly, compared to past work (Wang, Gao, and Qi 2016; Fawzi, Fawzi, and Frossard 2015), our robustness definition has a clear relation to accuracy and not orthogonal to it. Also, our definition uses the ideal 0/1 loss function rather than an approximate loss function l (often used in practice due to smoothness). We posit that this measures robustness accurately, as it prevents an adversary from bypassing the definition by exploiting the approximation by l when the true loss is 0/1.

3.2 ALN

Our goal is to train a neural network ALN to produce samples in the misclassification region of a given neural network based classifier. Thus, we choose the following loss function for the ALNN that takes into account the output for the given classifier:

$$\alpha sim(x, x') - H(Cat(y_x), C_p(x')) \ ,$$

where $Cat(y_x)$ is the categorical probability distribution with the component for y_x set to 1 and all else 0. The similarity term in the loss function aims to produce data points x' that are similar to the original input x while the cross entropy term aims to maximize the difference between the true label of x and prediction of C on x'. The α is a weight that is tuned through a simple search. Observe that this loss function is general and can be used with any classifier (by inferring C_p from C in case of specific non neural network based classifiers). For the image classification problem we use the l_2 distance for sim.

Note that an alternate loss function is possible that does not use the actual label of x, rather using $C_p(x)$. This would also work assuming that the classifier is good; for poor classifiers a lot of the data points are as it is mis-classified and hence adversarial example generation is interesting only for good classifiers. Further, this choice would allow using unlabeled data for conducting such an attack, making attack easier for an attacker. However, in our experiments we use the more powerful attack using the labels.

Further, for colored images we used an additional term to encourage the proportion of R:G:B pixel values to remain unchanged—this was informed by the observation that the attack often perturbs only one of the RGB channels to produce an adversarial example, but the output visually appears different from the input, which points to the fact that low l_2 distance is not enough to capture similarity for colored images. The additional term computes the l_2 distance between the vectors formed by concatenating R:B and G:B ratio for each pixel.

Next, we provide a formal intuition of what ALN actually achieves. Any adversarial example generation can be seen as a distribution transformer F such that acting on the data distribution D the resultant distribution $F(D)$ has support mostly limited to $M_C(X)$. The support may not completely limited to $M_C(X)$ as the attacks are never 100% effective. Also, attacks in literature aim to find points in $M_C(X)$ that are close to given images in the original dataset. ALN is essentially a neural network representation of such a function F against a given classifier C. See Figure 2B for an illustration. We return to this interpretation in the next sub-section to provide formal intuition about the defense.

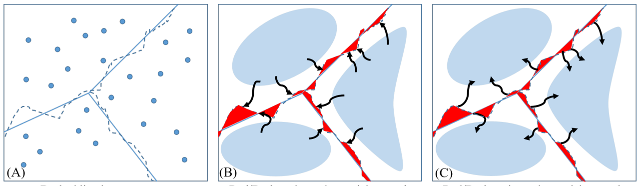

Figure 2: Intuition behind ALN and DLN. (A) shows a linear classifier (low capacity) is not able to accurately model a non-linear boundary. (B) shows the ALN as the distribution mapping function F. (C) shows that DLN does the reverse mapping of ALN.

3.3 DLN

Our defense approach is to insert a neural network DLN D between the input and classifier so that D sanitizes the input enabling the classifier to correctly classify the input. We start with a single attack A and our dataset for training the DLN consists of original training images x's and adversarially perturbed training images $A(x)$'s. Each data point for training DLN has three parts: x' is the image to sanitize, x is the original image corresponding to x' (this is same as x when x' comes from the original training data) and the label y_x of x.

We formulate a loss function for DLN that, similar to ALN, has two terms: $sim(x, x')$ that aims to produce output x' close to x and $H(Cat(y_x), H(C_p(D(x')))$ that aims to make the classifier output on x and x' be the same. Thus, the loss function is

$$\alpha sim(x, x') + H(Cat(y_x), C_p(D(x'))) \ .$$

In this paper we only use $\alpha = 1$. Note that the attack A is used as a black box here to generate training data and is not a part of the loss function. After training the DLN, our new classifier is C' which is C prepended by the DLN. The working of DLN can be interpreted as an inverse map F^{-1} for the mapping F induced by the attack A. See Figure 2C for an illustration.

An important point to note is that the original classifier C is unchanged. What this ensures is the mis-classification space $M_C(X)$ does not change and allows us to prove an important result about C' under certain assumptions. For the sake of this result, we assume perfect attacks A that generate adversarial examples in a sub region $M_{C,A}(X) \subset M_C(X)$. We also assume a good DLN D, that is, $C(D(x))$ is correct for a non-empty subset $Z \subset M_{C,A}(X)$ and $C(D(x))$ continues to be correct for all $x \notin M_C(X)$. Then, we prove

Lemma 1. *Assuming $M_{C,A}(X) \subset M_C(X)$, DLN is good as defined above, and $M_{C,A}(X) \neq \phi$, then $M_{C'}(X) \subset M_C(X)$*

Proof. Since DLN does not decrease the performance of C on points outside $M_C(X)$, C''s prediction on inputs outside $M_C(X)$ is correct, hence $M_{C'}(X) \subseteq M_C(X)$. Any data point not mis-classified by a classifier does not belong to its mis-classification space. Good sanitization by DLN makes C' predict correctly on $Z \subset M_{C,A}(X)$, which makes $M_{C,A}(X) \cap M_{C'}(X) \subset M_{C,A}(X)$. Thus, we can claim the result in the lemma statement. □

While the above proof is under ideal assumptions, it provides an intuition to how the defense works. Namely, the reduction in the adversarial exploitable space makes the new classifier more robust (see robustness properties earlier). This also motivates the generalization of this technique to multiple attacks presented in the next sub-section.

3.4 DLN Against Multiple Attacks

The above DLN can be naturally extended to multiple attacks, say $A_1, \ldots, A_n$. The only change required is to feed in all possible adversarial examples $A_1(x)$'s, . . . ,$A_n(x)$'s. It is straightforward to see that under assumptions of Lemma 1 for all the attacks, the resultant classifier C' has an adversarial example space $M_{C'}(X)$ that removes subsets of $M_{C,A_i}(X)$ for all $A_i \in \mathcal{A}$ from $M_C(X)$. This provides, at least theoretically under ideal assumptions, a monotonic robustness improvement property with increasing number of attacks for the DLN based approach.

In fact, if all the attacks combined act as a generator for all the points in $M_C(X)$, then given enough data and perfect sanitization the resultant classifier C' tends towards achieving $M_{C'}(X) = \phi$ which essentially would make C' a perfect classifier. Perfect classifiers have no adversarial examples. Thus, more powerful attacks used to train the DLN pushes the resultant classifier towards perfection. Of course, the ability of DLN to sanitize well is limited by its capacity, which in the worst case may need to be infinity. Our experiments show that for more complex domains (CIFAR dataset) the capacity of DLN does become an impediment to increasing robustness.

4 Experiments for ALN

All our experiments, including for DLN, were conducted using the Tensorflow framework on a NVIDIA K40 GPU. The learning problem can be solved using any gradient-based optimizer. In our case, we used Adam with learning rate 0.01. We use two well-known datasets: MNIST digits dataset and CIFAR-10 colored images dataset.

We consider two classifiers one for MNIST and one for CIFAR-10: we call them C_M and C_C. These classifiers are exactly the same ones used in (Carlini and Wagner 2017b). As stated earlier, we consider three attacks: ALN, FGSM and CW. CW has been referred to in the literature (Xu, Evans, and Qi 2017) as one of the best attacks till date (at the time of writing of this paper), while FGSM runs extremely fast. For the autoencoder we use a three hidden layer convolutional architecture for MNIST and a single hidden layer convolutional architecture for CIFAR-10. The single layer autoencoder has lower complexity and hence lower capacity; this choice allows us check the effect of capacity of the ALN/DLN network in our attacks and defenses. For our experiments we pre-process all the images so that the pixels values lie between $[-0.5, 0.5]$, so all components (attacks, autoencoders, classifiers) work in space $[-0.5, 0.5]$.

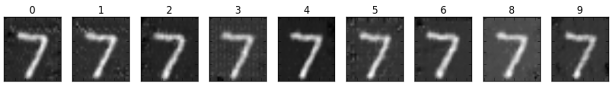

Figure 3: Targeted attacks by ALN: target class on top

Observe that all these attacks work against a given classifier C, thus, we will use the function $A(C, .)$ that acts on image x to produce the adversarial example $A(C, x)$ (A can be any of the three attacks). $A(C, Z)$ denotes the set of adversarial examples produced by using $A(C, .)$ on each image in the set Z. We test accuracies on various test sets: (1) original test dataset (OTD): these images are the original test dataset from the dataset under consideration, (2) $A(C, OTD)$ is the adversarially perturbed dataset using attack A against classifier C, for example, this could be FGSM(C_M, OTD). We also report distortion numbers as has been done in literature (Carlini and Wagner 2017b). Distortion measures how much perturbation on average was added by an attack and is meant to capture how visually similar the image is to the original image. Distortion is measured as the average over all test images of the l_2 distance between the original and perturbed image.

Results: Untargeted attacks refers to attack that aim to produce mis-classification but not with the goal of making the classifier output a specific label. It is also possible for ALN to perform targeted attacks by modifying the loss function to look like the DLN loss function but using the target class label instead of the original class label in the cross entropy term. Then, we can perform an ALN untargeted attack in two ways: ALNU uses the ALN loss function as stated and ALNT constructed a targeted attack per class label differing from the original label and chooses the one with least distortion. Figure 3 shows an example of targeted attack with different target labels. Table 1 shows this approach for MNIST with the targeted version performance better than other attacks. Our observation was that ALNU tends to modify images so that all perturbed images belong to one class, which is why ALNU hits a limit of 10% accuracy. For the rest of this paper, we use the ALNU attack (calling it ALN).

Test data type	Accuracy	Distortion
OTD	99.45 %	–
FGSM(C_M, OTD)	8.18 %	0.514
CW(C_M, OTD)	0.05 %	0.053
ALNU(C_M, OTD)	12.72 %	0.197
ALNT(C_M, OTD)	0.0 %	0.087

Table 1: Attacks on MNIST Dataset

Table 2 shows the result of untargeted attacks using ALN, FGSM and CW on the CIFAR-10 dataset. We can see that ALN produces the same accuracy as for MNIST, but its distortion is high. However, we argue that distortion is just one aspect of the difference between colored images. Recall that we also controlled the R:G:B ratio and hence our images do

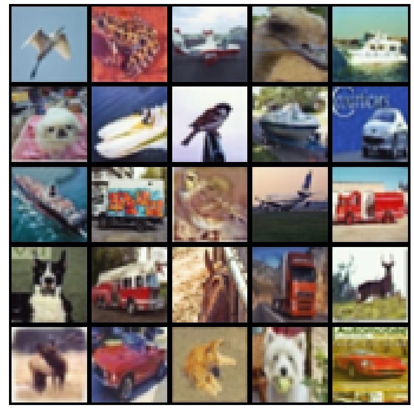

Figure 4: Untargeted attack by ALN on CIFAR-10

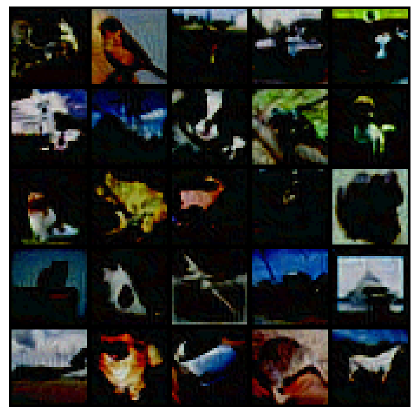

Figure 5: Attack using FGSM against C'_C trained against FGSM

not look visually distorted even at high distortion number—see Figure 4 for randomly chosen 25 perturbed images using ALN. Also, we can attribute the higher distortion partially to the low capacity of the autoencoder we used for CIFAR-10.

Test data type	Accuracy	Distortion
OTD	81.59 %	–
FGSM(C_C, OTD)	15.56 %	10.06
CW(C_C, OTD)	0.02 %	0.16
ALN(C_C, OTD)	12.57 %	9.62

Table 2: Attacks on CIFAR-10 Dataset

5 Experiments for DLN

For defense, we denote the new classifier using the DLN trained against attack $A_1, \ldots, A_n$ as C'_M or C'_C. Also, we additionally test accuracies on test data modified by attacks on the new classifiers, for example, following our convention one such dataset would be denoted as $A(C'_M, OTD)$.

5.1 Defense Against Single Attack

Table 3 shows the results when the DLN is trained to defense against specific attacks using MNIST dataset. Table 4 shows the same results for CIFAR-10 dataset. Along expected lines, the accuracy on OTD drops whereas the new classifier becomes much more difficult to attack.

One number that stands out is the success of FGSM in attacking the newer classifier C'_M and C'_C. However, for CIFAR this comes at an additional cost of much higher distortion. Figure 5 shows 25 randomly chosen adversarial images generated by FGSM for this case; these images are difficult to perceive. Thus, this attack cannot be called a success for CIFAR. For MNIST, we show in later sections that the performance of the classifier greatly improves when DLN is trained against multiple attacks or is repeatedly trained against FGSM—revealing that the DLN approach is flexible enough to keep on improving its performance.

DLN Trained	Test data type	Accuracy	Distortion
FGSM	OTD	83.11 %	–
FGSM	FGSM(C_M, OTD)	80.8 %	4.55
FGSM	FGSM(C'_M, OTD)	18.38 %	6.98
CW	OTD	88.98 %	–
CW	CW(C_M, OTD)	72.27 %	5.77
CW	CW(C'_M, OTD)	91.00 %	3.5
ALN	OTD	89.96 %	–
ALN	ALN(C_M, OTD)	82.60 %	3.82
ALN	ALN(C'_M, OTD)	81.20 %	3.79

Table 3: New DLN prepended classifier C'_M for MNIST

DLN Trained	Test data type	Accuracy	Distortion
FGSM	OTD	67.61 %	–
FGSM	FGSM(C_C, OTD)	32.41 %	13.4
FGSM	FGSM(C'_C, OTD)	14.11%	10.33
CW	OTD	77.80 %	–
CW	CW(C_C, OTD)	76.5 %	0.5
CW	CW(C'_C, OTD)	75.00 %	0.445
ALN	OTD	52.10 %	–
ALN	ALN(C_C, OTD)	49.16 %	1.96
ALN	ALN(C'_C, OTD)	47.56%	2.15

Table 4: New DLN prepended classifier C'_C for CIFAR-10

5.2 Defense Against Multiple Attacks

Table 5 shows the results of the classifier obtained after training DLN against FGSM and CW using MNIST dataset. As can be seen, the accuracy numbers are much better than when DLN was trained against single attacks. Further, FGSM was unable to attack the new classifier showing that training against multiple attacks makes the classifier robust overall. It can also be seen that the new classifier becomes slightly more resilient to the ALN attack also.

Table 6 shows the results of the classifier obtained after training DLN against FGSM and CW using CIFAR dataset. Here the accuracy numbers are in general worse than when DLN was trained against single attacks. However, the distortion numbers are much higher and the attack images are visually not discernible. Thus, the attacks do become more difficult. The drop in accuracy on the OTD can be attributed to the low capacity (single layer auto-encoder) of the DLN used, so that it is unable to do perfect sanitization. It can also be seen that the new classifier becomes slightly more resilient to the ALN attack.

Test data type	Accuracy	Distortion
OTD	98.27 %	–
FGSM(C_M, OTD)	89.44 %	3.02
FGSM(C'_M, OTD)	87.47 %	3.35
CW(C_M, OTD)	94.50 %	2.65
CW(C'_M, OTD)	91.00 %	5.55
ALN(C_M, OTD)	88.94 %	4.79
ALN(C'_M, OTD)	84.67 %	5.12

Table 5: New DLN prepended classifier C'_M with DLN trained against FGSM+CW for MNIST

Test data type	Accuracy	Distortion
OTD	59.89 %	–
FGSM(C_C, OTD)	24.66 %	11.55
FGSM(C'_C, OTD)	23.64 %	13.97
CW(C_C, OTD)	53.68 %	13.85
CW(C'_C, OTD)	59.00 %	13.87
ALN(C_C, OTD)	55.15 %	12.96
ALN(C'_C, OTD)	53.25 %	13.35

Table 6: New DLN prepended classifier C'_C with DLN trained against FGSM+CW for CIFAR-10

6 Towards Perfection by a Competition

A natural extension for DLN based approach is to repeatedly attack and re-learn a DLN in rounds. One way to do this is to conduct an attack on the classifier obtained in every round and train a DLN in a round using the attacked training data from all the previous rounds and the original training data. The following result provides formal intuition for this approach:

Lemma 2. *Assume the following conditions hold for every round i:* $M_{C_{i-1},A}(X) \subset M_{C_{i-1}}(X)$ *and the DLN* D_i *has good memory, which means that given there exists a largest set* $Z_i \subset M_{C_{i-1},A}(X)$ *which the DLN* D_i *correctly sanitizes so that* $C(D_i(x))$ *is correct for all* $x \in Z_i$ *then* $Z_{i-1} \subset Z_i$*. That is DLN* D_i *can correctly sanitize data points that the previous round DLN did plus possibly more data points. Further,* $C(D_i(x))$ *continues to be correct for all* $x \notin M_C(X)$*. Then the classifier* C_n *after* n *rounds satisfies* $M_{C_n}(X) \subset M_{C_{n-1}}(X)$*.*

Round	Acc. OTD	Acc. FGSM(C_i, OTD)	Distortion
0	99.45 %	8.18 %	0.514
1	83.11 %	18.38 %	6.98
2	93.28 %	21.41 %	5.89
3	90.60 %	33.26 %	5.74
4	87.36 %	59.99 %	3.93
5	91.54 %	64.95 %	3.76

Table 7: Classifier trained repeatedly against FGSM

Proof. Arguing similarly to Lemma 1 we can show that $M_{C_n}(X) \subseteq M_C(X)$ due to the correct classification outside of $M_C(X)$. Further, it is easily inferable that $M_{C_n}(X) = M_C(X) \backslash Z_n$ given Z_n is a subset of $M_C(X)$ and given the largest such set condition on Z_i. Then, the good memory property leads to the required result. □

The attack-defense competition technique is somewhat akin to GANs (Goodfellow et al. 2014). However, there is a big difference, since in every round the dataset used to train the DLN grows. Practically, this requires DLN to have a large capacity in order to be effective; also depending on the capacity and the size of dataset over or under fitting problems could arise, which needs to be taken care of in practice. We show the effectiveness of our repetition approach for a special case using the FGSM attack against MNIST. As noted earlier in Table 3 FGSM was able to successfully attack the new classifier that was trained against FGSM. After repeatedly training the DLN as described above, the resultant classifier becomes much more resistant to attack by FGSM, as shown in Table 7. That table shows that the accuracy on the original test data decreases by a small amount but the accuracy on the adversarially perturbed dataset improves greatly with increasing number of rounds.

7 Related Work

A thorough survey of security issues in machine learning is present in surveys (Papernot et al. 2016c) and some of the first results appeared in (Lowd and Meek 2005; Dalvi et al. 2004). Here we discuss the most closely related work.

Attacks: Most previous attack work focuses on adversarial examples for computer vision tasks. Multiple techniques to create such adversarial examples have been developed recently. Broadly, such attacks can be categorized as either using costs gradients (Goodfellow, Shlens, and Szegedy 2014; Moosavi-Dezfooli, Fawzi, and Frossard 2015; Huang et al. 2015; Biggio et al. 2013) or the forward gradient of the neural network (Papernot et al. 2016b) and perturbing along most promising direction or directly solving an optimization problem (possibly using gradient ascent/descent) to find a perturbation (Moosavi Dezfooli et al. 2017; Carlini and Wagner 2017b). In addition, adversarial examples have been shown to transfer between different network architectures, and networks trained on disjoint subsets of data (Szegedy et al. 2013a; Papernot, McDaniel, and Goodfellow 2016). Adversarial examples have also been shown to translate to the real world (Kurakin, Goodfellow, and Bengio 2016), that is, adversarial images can remain adversarial even after being printed and recaptured with a cell phone camera. Attacks on non-neural networks have also been explored in literature (Biggio et al. 2013). Our approach is distinctly different from all these approaches as we pose the problem of generating adversarial samples as a generative learning problem, and demonstrate generation of adversarial examples given access to any given classifier.

Defense: Also, referred to as robust classification in many papers, defense techniques can be roughly categorized into techniques that do (1) adversarial (re)training, which is adding back adversarial examples to the training data and retraining the classifier, often repeatedly (Li, Vorobeychik, and Chen 2016), or modifying loss function to account for attacks (Huang et al. 2015); (2) gradient masking, which targets that gradient based attacks by trying to make the gradient less informative (Papernot et al. 2016a); (3) input modification, which are techniques that modify (typically lower the dimension) the feature space of the input data to make crafting adversarial examples difficult (Xu, Evans, and Qi 2017); (4) game-theoretic formulation, which modifies the loss minimization problem as a constrained optimization with constraints provided by adversarial utility in performing perturbations (Li and Vorobeychik 2014), and (5) filtering and de-noising, which aims to detect and filter or de-noise adversarial examples (cited below).

Our defense approach differs from the first four kinds of defense as we aim to never modify the classifier or inputs but add a sanitizer (DLN) before the classifier. First, this increases the capacity of the resultant classifier C', so that it can model more complex separators, which is not achieved when the classifier family stays the same. Further, our defense is agnostic to the type of attack and does not utilize properties of specific types of attacks. We demonstrated the effectiveness of our defense against multiple attacks.

More closely related to our work are some defense techniques that have focused on detecting and filtering out adversarial samples (Li and Li 2016; Grosse et al. 2017) or de-noising input (Gu and Rigazio 2014); here the filter or de-noiser with the classifier could be considered as a larger neural network. However, unlike these work, our goal is targeted sanitization. Moreover, recent attack work (Carlini and Wagner 2017a) have produced attack techniques to defeat many known detection techniques. Our technique provides the flexibility to be resilient against more powerful attacks by training the DLN with such an attack.

Lastly, two concurrent drafts (available online) have independently from our work proposed an attack (Baluja and Fischer 2017) and a defense (Chen, Li, and Vorobeychik 2016) similar to ours. The difference for the attack work is in using the class label vs classifier output in cross entropy for the attack and an additional term that we use for colored images. For the defense work, we show how the DLN technique extends to multiple attacks and can be repeatedly used in an attack-defense competition. Moreover, unlike these experimental papers, we also define robustness and show that our defense technique approximately aims to achieve our definition of robustness. Also our formal reasoning reveals that our attack and defense against single attacks are duals of each other.

8 Conclusion

Our work provides a new learning perspective of the adversarial examples generation and defense problems with a formal intuition of how these approaches work. Further, unlike past work, our defense technique does not claim to a catchall or specific to any attack; in fact, it is flexible enough to possibly defend against any attack. Posing the attack and defense as learning problems allows for the possibility of using the rapidly developing research in machine learning to make the defense more effective in future, for example, by using a different specialized neural network architecture than an autoencoder.

References

[Anthony and Bartlett 2009] Anthony, M., and Bartlett, P. L. 2009. *Neural Network Learning: Theoretical Foundations*. New York, NY, USA: Cambridge University Press, 1st edition.

[Baluja and Fischer 2017] Baluja, S., and Fischer, I. 2017. Adversarial transformation networks: Learning to generate adversarial examples. *CoRR* abs/1703.09387.

[Biggio et al. 2013] Biggio, B.; Corona, I.; Maiorca, D.; Nelson, B.; Šrndić, N.; Laskov, P.; Giacinto, G.; and Roli, F. 2013. Evasion attacks against machine learning at test time. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 387–402. Springer.

[Carlini and Wagner 2017a] Carlini, N., and Wagner, D. 2017a. Adversarial examples are not easily detected: Bypassing ten detection methods. *arXiv preprint arXiv:1705.07263*.

[Carlini and Wagner 2017b] Carlini, N., and Wagner, D. 2017b. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, 39–57. IEEE.

[Chen, Li, and Vorobeychik 2016] Chen, X.; Li, B.; and Vorobeychik, Y. 2016. Evaluation of defensive methods for DNNs against multiple adversarial evasion models. `https://openreview.net/forum?id=ByToKu9ll&noteId=ByToKu9ll`.

[Dalvi et al. 2004] Dalvi, N.; Domingos, P.; Sanghai, S.; Verma, D.; et al. 2004. Adversarial classification. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 99–108. ACM.

[Fawzi, Fawzi, and Frossard 2015] Fawzi, A.; Fawzi, O.; and Frossard, P. 2015. Fundamental limits on adversarial robustness. In *Proc. ICML, Workshop on Deep Learning*.

[Goodfellow et al. 2014] Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, 2672–2680.

[Goodfellow, Shlens, and Szegedy 2014] Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *CoRR* abs/1412.6572.

[Grosse et al. 2017] Grosse, K.; Manoharan, P.; Papernot, N.; Backes, M.; and McDaniel, P. 2017. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280*.

[Gu and Rigazio 2014] Gu, S., and Rigazio, L. 2014. Towards deep neural network architectures robust to adversarial examples. *arXiv preprint arXiv:1412.5068*.

[Huang et al. 2015] Huang, R.; Xu, B.; Schuurmans, D.; and Szepesvári, C. 2015. Learning with a strong adversary. *arXiv preprint arXiv:1511.03034*.

[Kurakin, Goodfellow, and Bengio 2016] Kurakin, A.; Goodfellow, I.; and Bengio, S. 2016. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*.

[Li and Li 2016] Li, X., and Li, F. 2016. Adversarial examples detection in deep networks with convolutional filter statistics. *arXiv preprint arXiv:1612.07767*.

[Li and Vorobeychik 2014] Li, B., and Vorobeychik, Y. 2014. Feature cross-substitution in adversarial classification. In *Advances in neural information processing systems*, 2087–2095.

[Li, Vorobeychik, and Chen 2016] Li, B.; Vorobeychik, Y.; and Chen, X. 2016. A general retraining framework for scalable adversarial classification. *arXiv preprint arXiv:1604.02606*.

[Lowd and Meek 2005] Lowd, D., and Meek, C. 2005. Adversarial learning. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, 641–647. ACM.

[Moosavi Dezfooli et al. 2017] Moosavi Dezfooli, S. M.; Fawzi, A.; Fawzi, O.; and Frossard, P. 2017. Universal adversarial perturbations. In *Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, number EPFL-CONF-226156.

[Moosavi-Dezfooli, Fawzi, and Frossard 2015] Moosavi-Dezfooli, S.; Fawzi, A.; and Frossard, P. 2015. Deepfool: a simple and accurate method to fool deep neural networks. *CoRR* abs/1511.04599.

[Papernot et al. 2016a] Papernot, N.; McDaniel, P.; Goodfellow, I.; Jha, S.; Celik, Z. B.; and Swami, A. 2016a. Practical black-box attacks against deep learning systems using adversarial examples. *arXiv preprint arXiv:1602.02697*.

[Papernot et al. 2016b] Papernot, N.; McDaniel, P.; Jha, S.; Fredrikson, M.; Celik, Z. B.; and Swami, A. 2016b. The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, 372–387. IEEE.

[Papernot et al. 2016c] Papernot, N.; McDaniel, P.; Sinha, A.; and Wellman, M. 2016c. Towards the science of security and privacy in machine learning. *arXiv preprint arXiv:1611.03814*.

[Papernot, McDaniel, and Goodfellow 2016] Papernot, N.; McDaniel, P.; and Goodfellow, I. 2016. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*.

[Szegedy et al. 2013a] Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013a. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

[Szegedy et al. 2013b] Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I. J.; and Fergus, R. 2013b. Intriguing properties of neural networks. *CoRR* abs/1312.6199.

[Wang, Gao, and Qi 2016] Wang, B.; Gao, J.; and Qi, Y. 2016. A theoretical framework for robustness of (deep) classifiers under adversarial noise. *arXiv preprint arXiv:1612.00334*.

[Xu, Evans, and Qi 2017] Xu, W.; Evans, D.; and Qi, Y. 2017. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*.