TARGET TRAINING: TRICKING ADVERSARIAL ATTACKS TO FAIL

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent adversarial defense approaches have failed. Untargeted gradient-based attacks cause classifiers to choose any wrong class. Our novel white-box defense tricks untargeted attacks into becoming attacks targeted at designated target classes. From these target classes, we derive the real classes. The Target Training defense tricks the minimization at the core of untargeted, gradient-based adversarial attacks: minimize the sum of (1) perturbation and (2) classifier adversarial loss. Target Training changes the classifier minimally, and trains it with additional duplicated points (at 0 distance) labeled with designated classes. These differently-labeled duplicated samples minimize both terms (1) and (2) of the minimization, steering attack convergence to samples of designated classes, from which correct classification is derived. Importantly, Target Training eliminates the need to know the attack and the overhead of generating adversarial samples of attacks that minimize perturbations. We obtain 86.2% for CW-L_2(κ=0) in CIFAR10, exceeding even unsecured classifier accuracy on non-adversarial samples. Target Training presents a fundamental change in adversarial defense strategy.

1 INTRODUCTION

Neural network classifiers are vulnerable to malicious adversarial samples that appear indistinguishable from original samples (Szegedy et al., 2013), for example, an adversarial attack can make a traffic stop sign appear like a speed limit sign (Eykholt et al., 2018) to a classifier. An adversarial sample created using one classifier can also fool other classifiers (Szegedy et al., 2013; Biggio et al., 2013), even ones with different structure and parameters (Szegedy et al., 2013; Goodfellow et al., 2014; Papernot et al., 2016b; Tramèr et al., 2017b). This transferability of adversarial attacks (Papernot et al., 2016b) matters because it means that classifier access is not necessary for attacks. The increasing deployment of neural network classifiers in security and safety-critical domains such as traffic (Eykholt et al., 2018), autonomous driving (Amodei et al., 2016), healthcare (Faust et al., 2018), and malware detection (Cui et al., 2018) makes countering adversarial attacks important.

Gradient-based attacks use the classifier gradient to generate adversarial samples from non-adversarial samples. Gradient-based attacks minimize at the same time classifier adversarial loss and perturbation (Szegedy et al., 2013), though attacks can relax this minimization to allow for bigger perturbations, for example in the Carlini&Wagner attack (CW) (Carlini & Wagner, 2017c) for κ >0, in the Projected Gradient Descent attack (PGD) (Madry et al., 2017), in FastGradient-Method (FGSM) (Goodfellow et al., 2014). Other gradient-based adversarial attacks include DeepFool (Moosavi-Dezfooli et al., 2016), Zeroth order optimization (ZOO) (Chen et al., 2017), Universal Adversarial Perturbation (UAP) (Moosavi-Dezfooli et al., 2017).

Many recent proposed defenses have been broken (Athalye et al., 2018; Carlini & Wagner, 2016; 2017a;b; Tramer et al., 2020). They fall largely into these categories: (1) adversarial sample detection, (2) gradient masking and obfuscation, (3) ensemble, (4) customized loss. Detection defenses (Meng & Chen, 2017; Ma et al., 2018; Li et al., 2019; Hu et al., 2019) aim to detect, correct or reject adversarial samples. Many detection defenses have been broken (Carlini & Wagner, 2017b;a; Tramer et al., 2020). Gradient obfuscation is aimed at preventing gradient-based attacks from access to the gradient and can be achieved by shattering gradients (Guo et al., 2018; Verma & Swami, 2019; Sen et al., 2020), randomness (Dhillon et al., 2018; Li et al., 2019) or vanishing or exploding gradients (Papernot et al., 2016a; Song et al., 2018; Samangouei et al., 2018). Many gra-

dient obfuscation methods have also been successfully defeated (Carlini & Wagner, 2016; Athalye et al., 2018; Tramer et al., 2020). Ensemble defenses (Tramèr et al., 2017a; Verma & Swami, 2019; Pang et al., 2019; Sen et al., 2020) have also been broken (Carlini & Wagner, 2016; Tramer et al., 2020), unable to even outperform their best performing component. Customized attack losses defeat defenses (Tramer et al., 2020) with customized losses (Pang et al., 2020; Verma & Swami, 2019) but also, for example ensembles (Sen et al., 2020). Even though it has not been defeated, Adversarial Training (Szegedy et al., 2013; Kurakin et al., 2016; Madry et al., 2017) assumes that the attack is known in advance and takes time to generate adversarial samples at every iteration. The inability of recent defenses to counter adversarial attacks calls for new kinds of defensive approaches.

In this paper, we propose an adversarial defense that turns untargeted gradient-based attacks into attacks targeted at designated classes. Target Training forces attacks to converge to adversarial samples from designated classes, from which correct classification is derived. The Target Training defense is based on the minimization at the core of untargeted gradient-based attacks, introduced by Szegedy et al. (2013) for generation of adversarial attacks. We make the following major contributions:

- We develop Target Training - a novel, white-box adversarial defense that converts untargeted gradient-based attacks into attacks targeted at designated target classes, from which correct classes are derived. Target Training is based on the minimization at the core of untargeted gradient-based adversarial attacks.
- For all attacks that minimize perturbation, we eliminate the need to know the attack or to generate adversarial samples.
- Target Training surpasses default accuracy of 84.3% on non-adversarial samples in CIFAR10 for most attacks that minimize perturbation. We achieve: 86.2% for CW-L_2(κ=0), 84.2% for CW-L_∞(κ=0), 86.6% for DeepFool, 89.0% for ZOO and 86.8% for UAP.

2 Background and related work

Here, we present the state-of-the-art in adversarial attacks and defenses, as well as a summary.

Notation A k-class neural network classifier that has θ parameters is denoted by a function $f(x)$ that takes input $x \in \mathbb{R}^d$ and outputs $y \in \mathbb{R}^k$, where d is the dimensionality and k is the number of classes. An adversarial sample is denoted by x_{adv}. Classifier output is y, y_i is the probability that the input belongs to class i. Norms are denoted as L_0, L_2 and L_∞.

2.1 Adversarial attacks

Szegedy et al. (2013) were first to formulate the generation of adversarial samples as a constrained minimization of the perturbation under an L_p norm. Because this formulation can be hard to solve, Szegedy et al. (2013) reformulated the problem as a gradient-based, two-term minimization of the weighted sum of perturbation, and classifier loss:

$$\begin{aligned} &\text{minimize} && c \cdot \|x_{adv} - x\|_2^2 + loss_f(x_{adv}, l) \\ &\text{subject to} && x_{adv} \in [0,1]^n, \end{aligned} \tag{Minimization 1}$$

where l is an adversarial label and c a constant. Term (1) is a norm to ensure a small perturbation in the adversarial sample. Term (2) utilizes the classifier gradient to minimize classifier adversarial loss on an adversarial label when finding adversarial samples.

Minimization 1 is the foundation for many gradient-based attacksSome attacks follow Minimization 1 implicitly (Moosavi-Dezfooli et al., 2016), and others explicitly (Carlini & Wagner, 2017c). The type of L_p norm in term (1) of the minimization also varies. For example the CW attack (Carlini & Wagner, 2017c) uses L_0, L_2 and L_∞, whereas DeepFool (Moosavi-Dezfooli et al., 2016) uses the L_2 norm. A special perturbation case is the Pixel attack by Su et al. (2019) which changes exactly one pixel. Attacks can even exclude term (1) from the Minimization 1 to allow more perturbation by using an external parameter to control perturbation. The FGSM attack by Goodfellow et al. (2014), for example, uses an ϵ parameter, while the CW attack (Carlini & Wagner, 2017c) uses a κ confidence parameter.

The Fast Gradient Sign Method by Goodfellow et al. (2014) is a simple, L_∞-bounded attack that constructs adversarial samples by perturbing each input dimension in the direction of the gradient by a magnitude of ϵ: $x_{adv} = x + \epsilon \cdot sign(\nabla_x loss(\theta, x, y))$.

The current strongest attack, CW (Carlini & Wagner, 2017c), changes the basic Minimization 1 by passing the c parameter to the second term and using it to tune the relative importance of the terms:

$$\begin{array}{ll} \text{minimize} & \|x_{adv} - x\|_2^2 + c \cdot loss_f(x_{adv}, l) \\ \text{subject to} & x_{adv} \in [0,1]^n. \end{array}$$

With a further change of variable, CW obtains an unconstrained minimization problem that allows it to optimize directly through back-propagation. CW has a κ parameter for controlling perturbation.

The DeepFool attack by Moosavi-Dezfooli et al. (2016) follows the Minimization 1 implicitly. DeepFool (Moosavi-Dezfooli et al., 2016) looks at the smallest distance of a point from the classifier decision boundary as the minimum amount of perturbation needed to change its classification. DeepFool approximates the classifier with a linear one, estimates the distance from the linear boundary, and then takes steps in the direction of the closest boundary until an adversarial sample is found.

Black-box attacks Black-box attacks assume no access to classifier gradients. Such attacks with access to output class probabilities are called score-based attacks, for example the ZOO attack (Chen et al., 2017), a black-box variant of the CW attack (Carlini & Wagner, 2017c). Attacks with access to only the final class label are decision-based attacks, for example the Boundary (Brendel et al., 2017) and the HopSkipJumpAttack (Chen et al., 2019) attacks.

Multi-step attacks The PGD attack (Madry et al., 2017) is an iterative method with an α parameter that determines a step-size perturbation magnitude. PGD starts at a random point x_0, projects the perturbation on an L_p-ball B at each iteration: $x(j+1) = Proj_B(x(j) + \alpha \cdot sign(\nabla_x loss(\theta, x(j), y))$. The BIM attack (Kurakin et al., 2016) applies FGSM (Goodfellow et al., 2014) iteratively with an α step.To find a universal adversarial perturbation, UAP (Moosavi-Dezfooli et al., 2017) iterates over the images and aggregates perturbations calculated as in DeepFool.

2.2 Adversarial defenses

Adversarial Training (Szegedy et al., 2013; Kurakin et al., 2016; Madry et al., 2017) is one of the first and few, undefeated defenses. Adversarial Training defends by populating low probability, so-called blind spots (Szegedy et al., 2013; Goodfellow et al., 2014) with adversarial samples labelled correctly, which redraws classifier boundaries. Adversarial Training is simple to implement, but its drawback is that it needs to know the attack, and it needs to generate adversarial samples during training. The Adversarial Training algorithm 2 in the Appendix is based on Kurakin et al. (2016).

Detection defenses Such defenses detect adversarial samples implicitly or explicitly, then correct or reject them. So far, many detection defenses have been defeated. For example, ten diverse detection methods (other network, PCA, statistical properties) were defeated by attack loss customization (Carlini & Wagner, 2017a); Tramer et al. (2020) used attack customization against (Hu et al., 2019); attack transferability (Carlini & Wagner, 2017b) was used against MagNet by Meng & Chen (2017); deep feature adversaries (Sabour et al., 2016) against (Roth et al., 2019).

Gradient masking and obfuscation Many defenses that mask or obfuscate the classifier gradient have been defeated (Carlini & Wagner, 2016; Athalye et al., 2018). Athalye et al. (2018) identify three types of gradient obfuscation: (1) Shattered gradients - incorrect gradients caused by non-differentiable components or numerical instability, for example with multiple input transformations by Guo et al. (2018). Athalye et al. (2018) counter such defenses with Backward Pass Differentiable Approximation. (2) Stochastic gradients in randomized defenses are overcome with Expectation Over Transformation (Athalye et al., 2017) by Athalye *et al*. Examples are Stochastic Activation Pruning (Dhillon et al., 2018), which drops layer neurons based on a weighted distribution, and (Xie et al., 2018) which adds a randomized layer to the classifier input. (3) Vanishing or exploding gradients are used, for example, in Defensive Distillation (DD) (Papernot et al., 2016a) which reduces the amplitude of gradients of the loss function. Other examples are PixelDefend (Song et al., 2018) and Defense-GAN (Samangouei et al., 2018). Vanishing or exploding gradients are broken with parameters that avoid vanishing or exploding gradients (Carlini & Wagner, 2016).

Complex defenses Defenses combining several approaches, for example (Li et al., 2019) which uses detection, randomization, multiple models and losses, can be defeated by focusing on the main defense components (Tramer et al., 2020).(Verma & Swami, 2019; Pang et al., 2019; Sen et al., 2020) are defeated ensemble defenses combined with numerical instability (Verma & Swami, 2019), regularization (Pang et al., 2019), or mixed precision on weights and activations (Sen et al., 2020).

2.3 SUMMARY

Many defenses have been broken. They focus on changing the classifier. Instead, our defense changes the classifier minimally, but forces attacks to change convergence. Target Training is the first defense based on Minimization 1 at the core of untargeted gradient-based adversarial attacks.

3 TARGET TRAINING

Target Training converts untargeted attacks to attacks targeted at designated classes, from which correct classification is derived. Untargeted gradient-based attacks are based on Minimization 1 (on page 2) of the sum of (1) perturbation and (2) classifier adversarial loss. Target Training trains the classifier with samples of designated classes that minimize both terms of the minimization at the same time. These samples are exactly what adversarial attacks look for, based on Minimization 1: nearby points (at 0 distance) that minimize adversarial loss. For attacks that relax the minimization by removing the perturbation term, we adjust Target Training to use adversarial samples against attacks that do not minimize perturbation.

Target Training eliminates the need to know the attack or to generate adversarial samples against attacks that minimize perturbation. Instead of adversarial samples, we use original samples labelled as designated classes because the they have minimum 0-distance from original samples. Following, we outline how Target Training minimizes both terms of Minimization 1 simultaneously.

Term (1) of the minimization - perturbation Against attacks that *do minimize* perturbation, such as CW ($\kappa = 0$) and DeepFool, Target Training uses duplicates of original samples in each batch instead of adversarial samples because these samples minimize the perturbation to 0 - no other points can have smaller distance. This eliminates the need and overhead of calculating adversarial samples against all attacks of that minimize perturbation. Algorithm 1 shows Target Training against attacks that minimize perturbations. Against attacks that do *not* minimize perturbation, such as CW ($\kappa > 0$), PGD and FGSM, Target Training replaces duplicated samples with adversarial samples from the attack. Target Training has no norm limitation because it aims to minimize perturbation to 0, which means L_p norms of 0, for any p. The adjusted algorithm is shown in Algorithm 3 in the Appendix.

Algorithm 1 Target Training of classifier N against attacks that minimize perturbation

Require: Batch size is m, number of dataset classes is k, untrained classifier N with $2k$ output classes, TRAIN trains a classifier on a batch and its ground truth
Ensure: Classifier N is Target-Trained against all attacks that minimize perturbation
  while training not converged **do**
    $B = \{x^1, ..., x^m\}$ ▷ Get random batch
    $G = \{y^1, ..., y^m\}$ ▷ Get batch ground truth
    $B' = \{x^1, ..., x^m, x^1, ..., x^m\}$ ▷ Duplicate batch
    $G' = \{y^1, ..., y^m, y^1 + k, ..., y^m + k\}$ ▷ Duplicate ground truth and increase duplicates by k
    TRAIN(N, B', G') ▷ Train classifier on duplicated batch and new ground truth
  end while

Term (2) of the minimization - classifier adversarial loss In a converged, multi-class classifier, the probabilities of adversarial classes are ~ 0 with no distinction among them (the real class has high ~ 1 probability). This causes untargeted attacks to converge to samples of whichever highest probability adversarial class, due to the minimization of adversarial loss in term (2). In order to force attack convergence to designated classes, a classifier would need to train so that one of the adversarial classes has higher probability than the other adversarial classes. Which means that the classifier would need to have two high probability outputs: the real class, and the designated adversarial class.

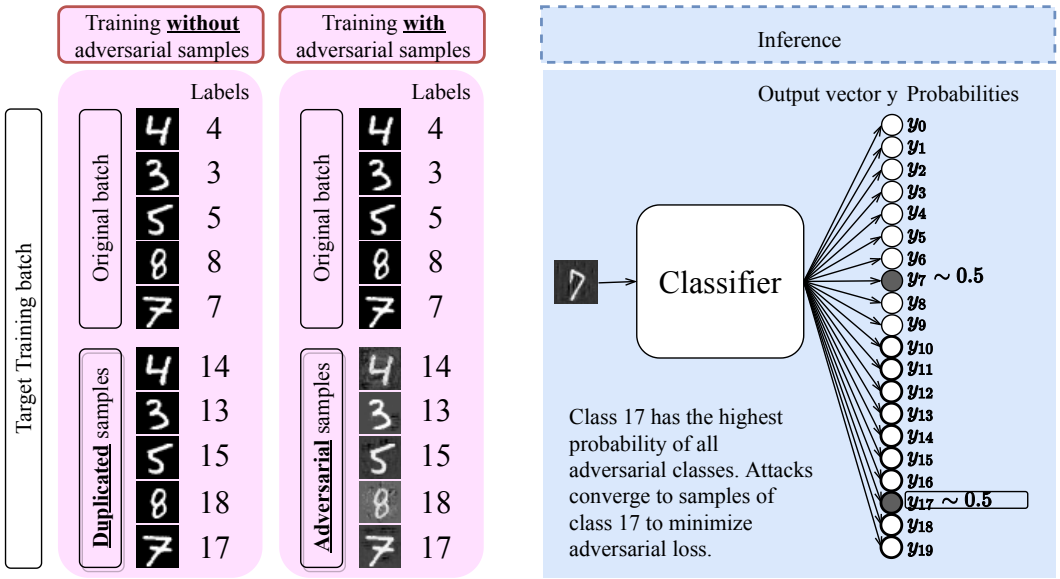

Figure 1: Target Training with and without adversarial samples, and output probabilities at inference. Example images are from the MNIST dataset, smaller batch size shown for brevity. Actual inference output probability values are shown in the Appendix in Table 7 for MNIST, in Table 8 for CIFAR10.

Target Training Against attacks that minimize perturbation, Target Training duplicates training samples in each batch and labels duplicates with designated classes. Against attacks that do not minimize perturbation, adversarial samples are used instead of duplicates. As a result, the real class and the designated class have equal inference probabilities of ~ 0.5, as shown in Figure 1. Since attacks minimize adversarial loss in term (2) of Minimization 1, attacks converge to adversarial samples from the designated class. The same samples also minimize term (1) because the classifier was trained with: samples with 0 distance, or adversarial samples close to original samples. It does not affect Target Training whether the c constant of the Minimization 1 is at term (1) or term (2) in an attack because we minimize both terms.

Model structure and inference The only change to classifier structure is doubling the number of output classes from k to $2k$, loss function remains standard softmax cross entropy. Inference calculation is: $C(x) = \arg\max_i (y_i + y_{i+k}), i \in [0 \dots (k-1)]$.

4 Experiments and Results

Threat model We assume that the *adversary goal* is to generate adversarial samples that cause untargeted misclassification. We perform white-box evaluations, assuming the adversary has complete *knowledge* of the classifier and how the defense works. In terms of *capabilities*, we assume that the adversary is gradient-based, has access to the CIFAR10 and MNIST image domains and is able to manipulate pixels. Against attacks that minimize perturbations, no adversarial samples are used in training and no further assumptions are made about attacks. For attacks that do not minimize perturbations, we assume that the attack is of the same kind as the attack used to generate the adversarial samples used during training. Perturbations are assumed to be L_p-constrained.

Attack parameters For *PGD*, we use the same attack parameters as Madry et al. (2017). For *CW*, we use $1,000$ iterations by default but we also run experiments with up to $100,000$ iterations. Confidence values in CW range 0-40. For *ZOO* attack, we use parameters specified in the ZOO attack paper (Chen et al., 2017). For *FGSM*, $\epsilon = 0.3$, as in (Madry et al., 2017).

Classifier models We purposefully do not use high capacity models, such as ResNet (He et al., 2016), used for example by Madry et al. (2017). The reason is to show that Target Training does not necessitate high model capacity to defend against adversarial attacks. The architectures of MNIST and CIFAR datasets are shown in Table 4 in the Appendix. No data augmentation used. Default accuracies without attack are 99.1% for MNIST and 84.3% for CIFAR10.

Datasets The MNIST (LeCun et al., 1998) and the CIFAR10 (Krizhevsky et al., 2009) datasets are 10-class datasets that have been used throughout previous work. The MNIST (LeCun et al., 1998) dataset has $60K$, $28 \times 28 \times 1$ hand-written, digit images. The CIFAR10 (Krizhevsky et al., 2009) dataset has $70K$, $32 \times 32 \times 3$ images. All experimental evaluations are with testing samples.

Tools Adversarial samples generated with CleverHans 3.0.1 (Papernot et al., 2018) for CW (Carlini & Wagner, 2017c), DeepFool (Moosavi-Dezfooli et al., 2016), FGSM (Goodfellow et al., 2014) attacks and IBM Adversarial Robustness 360 Toolbox (ART) toolbox 1.2 (Nicolae et al., 2018) for other attacks. Target Training is written in Python 3.7.3, using Keras 2.2.4 (Chollet et al., 2015).

4.1 Target Training against attacks that minimize perturbation

Target Training does not use adversarial samples against adversarial attacks that minimize perturbation. Thus, the undefeated Adversarial Training defense cannot be used as a baseline because it uses adversarial samples during training for all types of attack. Since other defenses have been defeated (Carlini & Wagner, 2017b;a; 2016; Athalye et al., 2018; Tramer et al., 2020) successfully, we use an unsecured classifier as a baseline. Table 1 shows that Target Training exceeds by far accuracies by unsecured classifier on adversarial samples in both CIFAR10 and MNIST. In CIFAR10, Target Training even exceeds the accuracy of the unsecured classifier on non-adversarial samples (84.3%) for most attacks. With CW-L_2 attack iterations from 1K-100K, accuracy increases for CIFAR10 from 85.6% to 86.2%, for MNIST from 96.3% to 96.6%.

Table 1: Target Training defends against attacks that minimize perturbations without using adversarial samples. In addition, Target Training exceeds even the accuracy of unsecured classifier on non-adversarial samples in CIFAR10. Target Training defends against attacks of different norms, against black-box attacks, and does not decrease performance for CW attacks with more iterations.

	CIFAR10 (84.3%)		MNIST (99.1%)	
Attack	Target Training	Unsecured Classifier	Target Training	Unsecur. Classif.
CW-L_2, $\kappa = 0$, 1K iterations	85.6%	8.8%	96.3%	0.8%
CW-L_2, $\kappa = 0$, 10K iterations	86.1%	8.7%	96.6%	0.8%
CW-L_2, $\kappa = 0$, 100K iterations	86.2%	8.9%	96.6%	0.8%
CW-L_∞, $\kappa = 0$, 1K iterations	84.2%	42.0%	96.3%	82.1%
DeepFool	86.6%	9.2%	94.9%	1.3%
ZOO	89.0%	81.5%	93.0%	96.0%
UAP	86.8%	17.24%	98.6%	42.1%

4.2 Target Training against attacks that do not minimize perturbation

Against adversarial attacks that do not minimize perturbation, Target Training uses adversarial samples and performs slightly better than Adversarial Training. We choose Adversarial Training as a baseline an undefeated adversarial defense, more details in Section 2. Our implementation of Adversarial Training is based on (Kurakin et al., 2016) by Kurakin *et al.*, shown in Algorithm 2. In Table 2, we show that Target Training performs slightly better than Adversarial Training against attacks that do not minimize perturbation.

In Table 6 in the Appendix, we show that both Target Training and Adversarial Training are able to defend in some cases against attacks, the adversarial samples of which have not been used during training - Target Training in more cases than Adversarial Training.

Table 2: Using adversarial samples in training, Target Training performs slightly better than Adversarial Training against attacks that do not minimize perturbation, both one-step and multi-step.

Attack	CIFAR10 (84.3%)			MNIST (99.1%)		
	Target Training	Adversarial Training	Unsecured Classifier	Target Training	Adversarial Training	Unsecured Classifier
CW-$L_2(\kappa = 40)$	77.7%	77.4%	8.5%	98.0%	98.0%	0.7%
PGD	76.3%	76.2%	32.7%	92.3%	91.7%	79.7%
FGSM($\epsilon = 0.3$)	72.1%	71.8%	11.8%	98.0%	98.4%	10.0%

4.3 Target Training performance on original, non-adversarial samples

In Table 5 in the Appendix, we show that Target Training maintains performance on original, non-adversarial samples. This is true when Target Training is trained with or without adversarial samples.

4.4 Summary of results

Our Section 4.1 experiments show that we substantially improve performance against attacks that minimize perturbation without using adversarial samples, surpassing even default accuracy for CIFAR10. Section 4.2 experiments show that Target Training maintains performance againt attacks that do not minimize perturbation, compared to previous non-broken defense, even slightly exceeding Adversarial Training performance.

4.5 Transferability analysis

For a defense to be strong, we need to show that it breaks the transferability of attacks. A good source of adversarial samples for transferability is the unsecured classifier (Carlini et al., 2019). We experiment on the transferability of attacks from the unsecured classifier to a classifier secured with Target Training. In Table 3, we show that Target Training breaks the transferability of adversarial samples generated by attacks that do not minimize perturbation.

Table 3: Target Training breaks the transferability of attacks generated from unsecured classifier by maintaining high accuracy against attacks generated using the unsecured classifier in attacks that minimize perturbation for CIFAR10 and MNIST. For attacks that do not minimize perturbation, Target Training breaks the transferability in MNIST, and partially in CIFAR10.

Attack	CIFAR10 (84.3%)		MNIST (99.1%)	
	Target Training	Unsecured Classifier	Target Training	Unsecured Classifier
CW-$L_2(\kappa = 0)$, iterations=1K	69.9%	8.8%	78.3%	0.8%
CW-$L_\infty(\kappa = 0)$, iterations=1K	76.6%	42.0%	93.5%	82.1%
DeepFool	74.8%	9.2%	96.5%	1.3%
CW-$L_2(\kappa = 40)$, iterations=1K	34.7%	8.5%	95.1%	0.7%
PGD	36.8%	32.7%	92.2%	79.7%

5 Adaptive evaluation

Many recent defenses have failed to anticipate attacks that have defeated them (Carlini et al., 2019; Carlini & Wagner, 2017a; Athalye et al., 2018). To avoid that, we perform an adaptive evaluation (Carlini et al., 2019; Tramer et al., 2020) of our Target Training defense.

Adaptive attacks that could defeat the Target Training defense Most current untargeted attacks, including the strongest one CW, are based on this Minimization 1. But this minimization is

a simplification of the hard problem of generating adversarial attacks, as outlined by Szegedy et al. (2013). In order to defeat Target Training, attacks would need to solve the much harder problem of generating adversarial attacks without using gradients. This remains an open problem.

Whether Target Training could be defeated by methods used to break other defenses Adaptive attack approaches (Carlini & Wagner, 2017b;a; 2016; Athalye et al., 2018; Tramer et al., 2020) used to defeat most current defenses cannot break Target Training defense because we use none of the previous defense approaches, such as: adversarial sample detection, preprocessing, obfuscation (shattered, vanishing or exploding gradients, or randomization), ensemble, customized loss, sub-component, non-differentiable component, or special model layers. We also keep the loss function simple - standard softmax cross-entropy and no additional loss. Unlike other attacks that try to prohibit attacks from doing their optimization, Target Training facilitates the convergence of attacks, steering the attacks to designated target classes.

Transferability We show that Target Training breaks the transferability of attacks generated using the unsecured classifier, as suggested by Carlini et al. (2019).

Iterative attacks Iterative attacks, such as PGD, decrease Target Training accuracy more than single-step attacks, which suggests that our defense is working correctly, according to Carlini et al. (2019).

Stronger CW attacks lead to better Target Training accuracy We include this in the adaptive evaluation because many attacks fail to defend from attacks when attacks become stronger. Increasing iterations for CW-$L_2(\kappa = 0)$ 100-fold from $1K$ to $100K$ increases the defense accuracy. In CIFAR10 accuracy increases from 85.6% to 86.2%, in MNIST from 96.3% to 96.6%. Target Training accuracy increases because higher number of iterations means that the attack converges better to samples of the designated classes, leading to higher accuracy.

No gradient masking or obfuscation The fact that Target Training defends against black-box ZOO attack is also an indicator that Target Training does not do gradient masking or obfuscation (Carlini et al., 2019).

6 Discussion and conclusions

Target Training presents a fundamental shift in adversarial defense by countering the minimization that is at the foundation of gradient-based adversarial attacks. Our defense is the only defense able to convert untargeted gradient-based attacks to attacks targeted at designated classes. From designated classes, correct classification is derived.

In addition, Target Training eliminates the need to know the attack in advance, and the overhead of adversarial samples, for all attacks that minimize perturbation. In contrast, the previous non-broken Adversarial Training defense needs to know the attack and to generate adversarial samples of the attack during training, which is a limitation because in real applications, the attack might not be known.

Target Training achieves high accuracy against adversarial samples. When trained without adversarial samples, against attacks that minimize perturbation, we achieve even better accuracy than 84.3% accuracy of unsecured classifier on non-adversarial samples in CIFAR10. For example, 86.2% for CW-$L_2(\kappa = 0)$, 84.2% for CW-$L_\infty(\kappa = 0)$, 86.6% for DeepFool, 89.0% for ZOO and 86.8% for UAP. Target Training also maintains performance on original, non-adversarial samples.

We show that Target Training breaks the transferability of adversarial samples in attacks that minimize perturbation for both CIFAR10 and MNIST. Target Training also breaks the transferability of adversarial samples in attacks that do not minimize perturbation in MNIST. This is important because many defenses are broken by adversarial samples generated using other classifiers.

In conclusion, we show that Target Training succeeds by switching the focus from changing the classifier to changing indirectly how attacks behave.

REFERENCES

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul F. Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety. *CoRR*, abs/1606.06565, 2016. URL http://arxiv.org/abs/1606.06565.

Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397*, 2017.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 387–402. Springer, 2013.

Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017.

Nicholas Carlini and David Wagner. Defensive distillation is not robust to adversarial examples. *arXiv preprint arXiv:1607.04311*, 2016.

Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 3–14. ACM, 2017a.

Nicholas Carlini and David Wagner. Magnet and" efficient defenses against adversarial attacks" are not robust to adversarial examples. *arXiv preprint arXiv:1711.08478*, 2017b.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 39–57. IEEE, 2017c.

Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, and Alexey Kurakin. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019.

Jianbo Chen, Michael I Jordan, and Martin J Wainwright. Hopskipjumpattack: A query-efficient decision-based attack. *arXiv preprint arXiv:1904.02144*, 3, 2019.

Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 15–26. ACM, 2017.

François Chollet et al. Keras. https://keras.io, 2015.

Zhihua Cui, Fei Xue, Xingjuan Cai, Yang Cao, Gai-ge Wang, and Jinjun Chen. Detection of malicious code variants based on deep learning. *IEEE Transactions on Industrial Informatics*, 14(7): 3187–3196, 2018.

Guneet S Dhillon, Kamyar Azizzadenesheli, Zachary C Lipton, Jeremy Bernstein, Jean Kossaifi, Aran Khanna, and Anima Anandkumar. Stochastic activation pruning for robust adversarial defense. In *International Conference on Learning Representations*, 2018.

Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1625–1634, 2018.

Oliver Faust, Yuki Hagiwara, Tan Jen Hong, Oh Shu Lih, and U Rajendra Acharya. Deep learning for healthcare applications based on physiological signals: a review. *Computer methods and programs in biomedicine*, 2018.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. In *International Conference on Learning Representations*, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Shengyuan Hu, Tao Yu, Chuan Guo, Wei-Lun Chao, and Kilian Q Weinberger. A new defense against adversarial images: Turning a weakness into a strength. In *Advances in Neural Information Processing Systems*, pp. 1633–1644, 2019.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 and cifar-100 datasets. *URl: https://www. cs. toronto. edu/kriz/cifar. html*, 6, 2009.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

Yann LeCun, Corinna Cortes, and Christopher JC Burges. The mnist database of handwritten digits, 1998. *URL http://yann. lecun. com/exdb/mnist*, 10:34, 1998.

Yingzhen Li, John Bradshaw, and Yash Sharma. Are generative classifiers more robust to adversarial attacks? In *International Conference on Machine Learning*, 2019.

Xingjun Ma, Bo Li, Yisen Wang, Sarah M Erfani, Sudanthi Wijewickrema, Grant Schoenebeck, Dawn Song, Michael E Houle, and James Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. In *International Conference on Machine Learning*, 2018.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Dongyu Meng and Hao Chen. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 135–147. ACM, 2017.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2574–2582, 2016.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1765–1773, 2017.

Maria-Irina Nicolae, Mathieu Sinn, Minh Ngoc Tran, Beat Buesser, Ambrish Rawat, Martin Wistuba, Valentina Zantedeschi, Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, Ian Molloy, and Ben Edwards. Adversarial robustness toolbox v1.0.1. *CoRR*, 1807.01069, 2018. URL `https://arxiv.org/pdf/1807.01069`.

Tianyu Pang, Kun Xu, Chao Du, Ning Chen, and Jun Zhu. Improving adversarial robustness via promoting ensemble diversity. In *International Conference on Learning Representations*, 2019.

Tianyu Pang, Kun Xu, Yinpeng Dong, Chao Du, Ning Chen, and Jun Zhu. Rethinking softmax cross-entropy loss for adversarial robustness. In *International Conference on Learning Representations*, 2020.

Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, pp. 582–597. IEEE, 2016a.

Nicolas Papernot, Patrick D. McDaniel, and Ian J. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *CoRR*, abs/1605.07277, 2016b. URL http://arxiv.org/abs/1605.07277.

Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, and Rujun Long. Technical report on the cleverhans v2.1.0 adversarial examples library. *arXiv preprint arXiv:1610.00768*, 2018.

Kevin Roth, Yannic Kilcher, and Thomas Hofmann. The odds are odd: A statistical test for detecting adversarial examples. *arXiv preprint arXiv:1902.04818*, 2019.

Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J Fleet. Adversarial manipulation of deep representations. In *International Conference on Learning Representations*, 2016.

Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. In *International Conference on Learning Representations*, 2018.

Sanchari Sen, Balaraman Ravindran, and Anand Raghunathan. Empir: Ensembles of mixed precision deep networks for increased robustness against adversarial attacks. In *International Conference on Machine Learning*, 2020.

Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. In *International Conference on Learning Representations*, 2018.

Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2013.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017a.

Florian Tramèr, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. The space of transferable adversarial examples. *arXiv preprint arXiv:1704.03453*, 2017b.

Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. *arXiv preprint arXiv:2002.08347*, 2020.

Gunjan Verma and Ananthram Swami. Error correcting output codes improve probability estimation and adversarial robustness of deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 8643–8653, 2019.

Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. In *International Conference on Learning Representations*, 2018.

A APPENDIX

Algorithm 2 Adversarial Training of classifier N

Require: Batch size is m, number of dataset classes is k, untrained classifier N with k output classes, ADV_ATTACK is an adversarial attack, TRAIN trains classifier on a batch and its ground truth

Ensure: Adversarially-Trained classifier N

while training not converged **do**

$B = \{x^1, ..., x^m\}$ ▷ Get random batch

$G = \{y^1, ..., y^m\}$ ▷ Get batch ground truth

$A = ADV_ATTACK(N, B)$ ▷ Generate adv. samples from batch

$B' = B \bigcup A = \{x^1, ..., x^m, x^1_{adv}, ..., x^m_{adv}\}$ ▷ New batch

$G' = \{y^1, ..., y^m, y^1, ..., y^m\}$ ▷ Duplicate ground truth

TRAIN(N, B', G') ▷ Train classifier on new batch and new ground truth

end while

Table 4: Architectures of Target Training classifiers for CIFAR10 and MNIST datasets. For the convolutional layers, we use L_2 kernel regularizer. Notice that the final Dense.Softmax layers in both models have 20 output classes, twice the number of dataset classes. The default, unsecured classifiers have the same architectures, except the final layers have 10 output classes: Dense.Softmax 10.

CIFAR10	MNIST
Conv.ELU 3x3x32	Conv.ReLU 3x3x32
BatchNorm	BatchNorm
Conv.ELU 3x3x32	Conv.ReLU 3x3x64
BatchNorm	BatchNorm
MaxPool 2x2	MaxPool 2x2
Dropout 0.2	Dropout 0.25
Conv.ELU 3x3x64	Dense 128
BatchNorm	Dropout 0.5
Conv.ELU 3x3x64	Dense.Softmax **20**
BatchNorm	
MaxPool 2x2	
Dropout 0.3	
Conv.ELU 3x3x128	
BatchNorm	
Conv.ELU 3x3x128	
BatchNorm	
MaxPool 2x2	
Dropout 0.4	
Dense.Softmax **20**	

Algorithm 3 Target Training of classifier N using adversarial samples against attacks that do not minimize perturbation

Require: Batch size is m, number of dataset classes is k, untrained classifier N with $2k$ output classes, ADV_ATTACK is an adversarial attack, TRAIN trains classifier on a batch and its ground truth

Ensure: Classifier N is Target-Trained against ADV_ATTACK

while training not converged **do**

$B = \{x^1, ..., x^m\}$ ▷ Get random batch

$G = \{y^1, ..., y^m\}$ ▷ Get batch ground truth

$A = ADV_ATTACK(N, B)$ ▷ Generate adv. samples from batch

$B' = B \bigcup A = \{x^1, ..., x^m, x^1_{adv}, ..., x^m_{adv}\}$ ▷ Assemble new batch from original batch and adversarial samples

$G' = \{y^1, ..., y^m, y^1 + k, ..., y^m + k\}$ ▷ Duplicate ground truth and increase duplicates by k

TRAIN(N, B', G') ▷ Train classifier on new batch and new ground truth

end while

Table 5: Target Training maintains performance on original, non-adversarial samples. When Target Training not using adversarial samples against attacks with no perturbation, Target Training even exceeds unsecured classifier accuracy on non-adversarial samples (Adversarial Training is not applicable here because it needs adversarial samples). When Target Training using adversarial samples against attacks with perturbation, Target Training and Adversarial Training perform equally.

	CIFAR10 (84.3%)			MNIST (99.1%)		
Attack used for adv. samples in training	Target Training	Adversarial Training	No Defense	Target Training	Adversarial Training	No Defense
none (against min. perturb. attacks)	86.7%	NA	84.3%	98.6%	NA	99.1%
CW-L_2 ($\kappa = 40$)	77.7%	77.4%	84.3%	98.0%	98.0%	99.1%
PGD	76.3%	76.9%	84.3%	98.3%	98.4%	99.1%
FGSM($\epsilon = 0.3$)	77.6%	76.6%	84.3%	98.6%	98.6%	99.1%

Table 6: Expanded comparison of performance against attacks that do not minimize perturbation shows that both Target Training and Adversarial Training can defend against attacks, the adversarial samples of which have not been used in training. Bold font has been used to highlight where such defense was effective. Both Target Training and Adversarial Training defend against some attacks that they have not been trained for, but not all. Unsecured classifier performance also provided as a performance baseline. Target Training appears to be slightly better at defending against attacks, the samples of which have not been used in training.

		CIFAR10 (84.3%)			MNIST (99.1%)		
Adv. samples in training	Adv. samples in testing	Target Train.	Advers. Train.	No Def.	Target Train.	Adv. Train.	No Def.
CW-L_2 ($\kappa = 40$)	*CW-L_2 ($\kappa = 40$)*	**77.7%**	**77.4%**	8.5%	**98.0%**	**98.0%**	0.7%
	CW-L_2 ($\kappa = 0$)	**71.3%**	12.3%	8.8%	**97.4%**	1.5%	8.8%
CW-L_2 (conf=40)	DeepFool	**75.8%**	13.2%	9.2%	**97.6%**	1.6%	1.3%
	PGD	10.0%	10.0%	32.7%	23.3%	2.9%	79.7%
	FGSM($\epsilon = 0.3$)	10.6%	9.9%	11.8%	**56.6%**	15.8%	10.0%
	FGSM($\epsilon = 0.01$)	48.9%	36.4%	40.4%	**97.7%**	**97.8%**	98.6%
PGD	*PGD*	**76.3%**	**76.2%**	32.7%	**92.3%**	**91.7%**	79.7%
	CW-L_2 ($\kappa = 40$)	7.3%	**57.3%**	8.5%	**83.2%**	**98.4%**	0.7%
	CW-L_2 ($\kappa = 0$)	12.8%	12.7%	8.8%	**94.3%**	22.7%	8.8%
	DeepFool	15.0%	13.0%	9.2%	**86.5%**	4.7%	1.3%
	FGSM($\epsilon = 0.3$)	10.7%	10.2%	11.8%	**79.9%**	**95.4%**	10.0%
	FGSM($\epsilon = 0.01$)	39.8%	41.5%	40.4%	**98.2%**	**98.4%**	98.6%
FGSM ($\epsilon = 0.3$)	*FGSM($\epsilon = 0.3$)*	**72.1%**	**71.8%**	11.8%	**98.0%**	**98.4%**	10.0%
	FGSM($\epsilon = 0.01$)	40.8%	42.1%	40.4%	**98.5%**	**98.5%**	98.6%
	CW-L_2 ($\kappa = 40$)	**49.9%**	**74.2%**	8.5%	**58.8%**	1.1%	0.7%
	CW-L_2 ($\kappa = 0$)	12.5%	12.7%	8.8%	**51.8%**	1.1%	8.8%
	DeepFool	12.7%	12.8%	9.2%	**48.3%**	1.2%	1.3%
	PGD	17.2%	1.2%	32.7%	**72.6%**	**42.5%**	79.7%

Table 7: Class output probabilities for Target Training on original, and adversarial samples from MNIST. Adversarial samples generated with CW-$L_2(\kappa = 0)$. Zero probability values and probability values rounded to zero have been omitted.

	Original images									
Labels										
0	0.51									
1		0.44								
2			0.62							
3				0.78						
4					0.75					
5						0.62				
6							0.65			
7								0.61		
8									0.52	
9										0.43
10	0.49									
11		0.57								
12			0.38							
13				0.22						
14					0.25					
15						0.38				
16							0.35			
17								0.39		
18									0.48	
19										0.57
	Adversarial images									
0	0.50									
1		0.50								
2			0.49							
3				0.49						
4					0.50					
5						0.50				
6							0.50			
7								0.46		
8									0.50	
9										0.50
10	0.50									
11		0.50								
12			0.51							
13				0.51						
14					0.50					
15						0.50				
16							0.50			
17								0.54		
18									0.50	
19										0.50

Table 8: Class output probabilities for Target Training on original, and adversarial samples from CIFAR10. Adversarial samples generated with CW-$L_2(\kappa = 0)$. Zero probability values and probability values rounded to zero have been omitted. The two highest class probabilities for each image are made bold. The deer (fifth image) appears to be misclassified as a horse.

	Original images									
Labels	air-plane	auto-mobile	bird	cat	deer	dog	frog	horse	ship	truck
0	**0.41**									
1		**0.46**								
2	0.01		**0.56**							
3				**0.60**						
4					0.08					
5					0.01	**0.48**				
6							**0.53**			
7					**0.39**			**0.56**		
8									**0.54**	
9										**0.47**
10	**0.58**									
11		**0.55**								
12	0.01		**0.43**							
13				**0.40**	0.01					
14					0.06					
15					0.01	**0.52**				
16							**0.47**			
17					**0.45**			**0.44**		
18									**0.46**	
19										**0.53**

	Adversarial images									
Labels	air-plane	auto-mobile	bird	cat	deer	dog	frog	horse	ship	truck
0	**0.49**		0.01							
1		**0.54**								
2	0.01		**0.49**							
3				**0.47**						
4					0.08					
5					0.01	**0.51**				
6							**0.48**			
7					**0.44**			**0.49**		
8									**0.46**	
9										**0.50**
10	**0.49**									
11		**0.46**								
12	0.01		**0.50**							
13				**0.53**	0.01					
14					0.05					
15					0.01	**0.49**				
16							**0.52**			
17					**0.41**			**0.51**		
18									**0.54**	
19										**0.50**