EVALUATION OF DEFENSIVE METHODS FOR DNNS AGAINST MULTIPLE ADVERSARIAL EVASION MODELS

Xinyun Chen
Shanghai Jiao Tong University
jungyhuk@gmail.com

Bo Li
University of Michigan
bbbli@umich.edu

Yevgeniy Vorobeychik
Vanderbilt University
yevgeniy.vorobeychik@vanderbilt.edu

ABSTRACT

Due to deep cascades of nonlinear units, deep neural networks (DNNs) can automatically learn non-local generalization priors from data and have achieved high performance in various applications. However, such properties have also opened a door for adversaries to generate the so-called adversarial examples to fool DNNs. Specifically, adversaries can inject small perturbations to the input data and therefore decrease the performance of deep neural networks significantly. Even worse, these adversarial examples have the transferability to attack a black-box model based on finite queries without knowledge of the target model. Therefore, we aim to empirically compare different defensive strategies against various adversary models and analyze the cross-model efficiency for these robust learners. We conclude that the adversarial retraining framework also has the transferability, which can defend adversarial examples without requiring prior knowledge of the adversary models. We compare the general adversarial retraining framework with the state-of-the-art robust deep neural networks, such as distillation, autoencoder stacked with classifier (*AEC*), and our improved version, *IAEC*, to evaluate their robustness as well as the vulnerability in terms of the distortion required to mislead the learner. Our experimental results show that the adversarial retraining framework can defend most of the adversarial examples notably and consistently without adding additional vulnerabilities or performance penalty to the original model.

1 INTRODUCTION

Despite the success of deep neural networks (DNNs) in diverse areas, ranging from image recognition and machine translation to autonomous driving, its vulnerabilities have been exploited in the adversarial environments. Evasion attacks against such deep learning systems have recently received considerable attention. It has been shown that with small magnitude of noise added, the original instance can easily be misclassified by the otherwise accurate deep neural networks (Goodfellow et al., 2014; Papernot et al., 2016c; Nguyen et al., 2015; Szegedy et al., 2013). Such instances are also called adversarial examples.

Given the strong evasion properties of these adversarial examples, some works have been proposed to test and investigate the robustness of the deep neural networks against the adversarial examples (Goodfellow et al., 2014; Kurakin et al., 2016; Huang et al., 2015; Gu & Rigazio, 2014; Jin et al., 2015). However, most of the existing works only evaluate the robustness of the proposed defense strategies over adversarial examples generated using a single attack method, or several similar methods. Meanwhile, since the evaluated adversary models, i.e., adversarial example generation methods, vary among different works that study the effectiveness of defense strategies, it remains a question how to make a comparison among different defense strategies.

In this paper, we focus on providing thorough analysis for different algorithmic strategic defensive learners against various adversary models considering their robustness against adversarial examples, efficiency for cross-model learning process, resilience against additional attacks, and the vulnerabilities of these learners. Here by “cross-model test”, we mean to apply one adversarial model to generate adversarial examples, while test them on the learner trained with instances generated from different adversarial models. High “cross-model test” accuracy indicates higher robustness for learner. In addition, we propose to test the “additional attacks” in a repeated game setting to estimate learner based on against further attacks. A nice symmetry analysis for both the adversary and learner is provided through these analyses. We show that the general adversarial retraining framework performs significantly robust compared with the state-of-the-art defensive algorithms. For example, even for the black box attack, which is considered hard to defend, as long as there is a way to generate these adversarial evasion examples, the robust adversarial retraining framework can always improve the learning ability without knowing the actual adversary model. To our best knowledge, this work is the first to provide comprehensive analysis for different adversarial models and possible defensive solutions.

In summary, we made the following contributions:

1. Evaluate the robustness of the general robust adversarial retraining framework (*RAD*) with the state-of-the-art *AEC*, Distillation, and the improved *AEC*, against different adversary models;
2. Propose an improved AutoEncoder stacked with Classifier (*IAEC*);
3. Compare the cross-model learning efficiency of different defensive methods and demonstrate the ability to defend against black-box attacks;
4. Demonstrate the robustness of the retraining framework *RAD*, *AEC*, *IAEC*, and Distillation, against new attacks by attacking these robust learners repeatedly;
5. Analyze the vulnerabilities induced by different defensive strategies/models based on their tolerance of the malicious distortions required to mislead the classifier.

We illustrate the applicability and efficiency of different defensive strategies against various state-of-the-art adversary models based on both MNIST and CIFAR-10 datasets.

2 Related Work

Efforts have been made to understand adversarial examples. Goodfellow et al. (2014) pointed out that the adversarial examples actually make use of the linear nature of the DNNs based on the observation of their generalization across architectures and training sets. Tabacof & Valle (2015) analyzed the adversarial image space and showed that adversarial images appear in large regions in the pixel space. Papernot et al. (2016c) studied the limitation of adversarial evasion examples and showed that some instances are more difficult to manipulate than the others. Sabour et al. (2015) demonstrated that the attacker can change classification to an arbitrary class by malicious manipulations. The reverse engineering problem has been proposed in Vorobeychik & Li (2014), and it theoretically proved that the black-box attack is possible and also showed one could learn a sufficiently similar classifier from queries both theoretically and empirically. Similarly, even without knowing exactly the learning algorithm, several black-box attacks have been proposed targeting DNNs, which demonstrates the transferability of such adversarial examples (Papernot et al., 2016a;b).

Some training methods have been proposed to improve the robustness of deep neural networks. Jan et al. (2002) has proposed to explore the perturbed regions and apply ensemble method to enhance the robustness of classification. Zheng et al. have proposed to stabilize the state-of-the-art Inception architecture against different distortions, and it focuses on general random noise or distortions, such as compression, rescaling and cropping on images Zheng et al. (2016). Miyato et al. (2015) have proposed to apply the local distribution smoothness for statistical model to promote the smoothness of the model distribution and conduct the virtual adversarial training to enhance the performance of deep neural networks. However, all these works did not test on the adversarial examples and still had a long way to perform robustly against these real adversarial instances.

While the existence of adversarial examples is attracting more and more attention, some defense strategies have been proposed to defend against such adversarial examples. In Goodfellow et al. (2014), Goodfellow et al. proposed to train the network with an adversarial objective function based on fast gradient sign method: $\tilde{J}(\theta, x, y) = \alpha J(\theta, x, y) + (1-\alpha)J(\theta, x + \nabla_x J(\theta, x, y), y)$. In a concurrent and independent work, Kurakin et al. provided an adversarial training method for a large scale dataset, i.e., ImageNet dataset Kurakin et al. (2016). However, they only used fast gradient sign-based methods to generate adversarial examples for both training and evaluation, which fails to consider the generality of defensive strategy. Huang et al. Huang et al. (2015) proposed an alternative method for adversarial training by considering an empirically stronger adversary. In their work, suppose $r^\star$ is the optimal adversarial perturbation for an instance (x, y), instead of adding $(x + r^\star, y)$ into the training dataset, they used "pseudo-samples" for training controlled by a hyperparameter c, which represents the magnitude of perturbation, i.e., $(x + c\frac{r^\star}{||r^\star||}, y)$. Several autoencoder structures Vincent et al. (2008) have been proposed against the adversarial examples by reconstructing the original images ahead of classification Gu & Rigazio (2014). Jin et al. have proposed a feedforward CNN structure to improve the robustness in the presence of adversarial noise, which is restricted to the specific type of models in Jin et al. (2015). However, the focus of these researches perform too aggressively on designing robust learning algorithms against arbitrary small perturbations (e.g., noise) neglecting the properties of actual adversarial evasion models. Therefore, studying various adversarial models and building resilient learners accordingly is important. Here we will provide comparisons for defensive algorithms facing different adversarial models to provide insights and encourage devising more efficient learner.

3 Problem

To understand the phenomenon of adversarial examples in deep neural networks, we aim to analyze potential defending methods against different adversary models from various perspectives, such as the robustness of the learner itself, the cross-model generalization ability, the resilience against additional attacks, and the vulnerabilities in terms of the required distortion to attack the robust learner again. Let $X \subseteq R^n$ represent the feature space, with n the number of features. For every instance $x_i \in X$, which is drawn from certain distribution $x_i \sim D$, there is a corresponding label $y_i \in \mathcal{Y}$ to comprise the data pair (x_i, y_i), where x_{ij} denotes the jth feature of x_i.

In the adversarial environments, adversary would like to accomplish the goal of evading the classifier. To formalize, suppose that $M \subseteq \mathcal{Y}$ is a set of labels which an adversary wishes to attack, and let $z(m)$ be the *target* label for each $m \in M$. For example, for autonomous driving, potential adversaries may aim to manipulate a stop sign or a dead-end warning sign, to a lamppost, a tree, or an advertisement sign, to cause accidents. Since such perturbations on images towards deep neural networks are often imperceptible to human eyes, it can cause serious vulnerabilities when deploying the DNNs in real adversarial environments. The defender's goal is to learn a classifier with parameters w, $g_w : x_i \to \mathcal{Y}$, using a training data set of labeled instance $T = \{(x_1, y_1), ..., (x_m, y_m)\}$. Here, we focus on deep neural networks representing the function $g_w(\cdot)$. Therefore, the learner's objective is to minimize the following general loss function:

$$\min_w \mathcal{L}(w; \mathcal{A}) = \sum_{i: y_i \in \mathcal{Y} \setminus M} l(g_w(x_i), y_i) + \sum_{i: y_i \in M} l(g_w(\mathcal{A}(w, x_i), y_i) + \alpha \|w\|_p^p, \tag{1}$$

where $l(\cdot)$ can be arbitrary loss function and $\mathcal{A}$ represents the adversary model.

The adversarial risk function in Equation 1 is general: it can be any adversary model oracle, $\mathcal{A}$, which is used to generate the adversarial evasion instances. Traditionally, this adversarial oracle may capture evasion attack models based on minimizing evasion cost (Lowd & Meek, 2005; Li & Vorobeychik, 2014; Biggio et al., 2014), or based on actual attacker evasion behavior obtained from experimental data (Ke et al., 2016). More formally, we will discuss the potential adversary models for deep neural networks and the possible defensive models for the learner in detail below.

3.1 Adversary Model

To mislead deep neural networks, various methods have been proposed to generate the adversarial examples. We mainly discuss three state-of-the-art adversary models $\mathcal{A}$ here for further evaluation.

Fast Gradient Sign. Based on the linear view of adversarial examples, a fast way of generating these adversarial examples were proposed in Goodfellow et al. (2014). Suppose x_i is the original feature vector, based on adversary model $\mathcal{A}(fgs)$, we have ${x_i}' = x_i + \eta$, where η represents the perturbation added for the original instance. Therefore, the dot product between the weighted parameter vector w and an adversarial example x_i' becomes:

$$w^T x_i' = w^T x_i + w^T \eta.$$

Let $J(w, x_i, y_i)$ be the cost used to train the neural network. By linearizing the cost function around the current value of w, an optimal max-norm constrained perturbation is generated as

$$\eta = \epsilon \text{sign}(\nabla_x J(w, x_i, y_i)),$$

where the adversary can vary ϵ to generate adversarial examples with different attacking abilities for different deep neural networks.

Coordinate Greedy. Another more general adversary model $\mathcal{A}(cg)$ is the local search framework *Coordinate Greedy (cg)* proposed in Li et al. (2016) for approximating the optimal adversarial instance. As an illustration, we focus on binary classification, and assume that $g_w(x) = \text{sign}(f(x))$ for some continuous function f, which in this case would be represented by a deep neural network.

The coordinate greedy approach is quite general, but we consider a specific adversary objective in which the adversary here tries to balance between two considerations: 1) appear as benign as possible to the classifier, and 2) minimize the cost of modification of the original instance (e.g., minimally manipulate the image). Note that it is also natural to assume that the attacker obtains no value from a manipulation to the original feature vector if the result is still classified as malicious. Therefore, an adversary aiming to transform an instance x_i into an adversarial example x_i' is solving the following optimization problem:

$$\min_{x_i' \in X} \min\{0, f(x_i')\} + c(x_i', x_i), \tag{2}$$

where $c(x_i', x_i)$ is the cost function of modifying from x_i to x_i'. Here $c(x_i', x_i) \geq 0$, $c(x_i', x_i) = 0$ iff $x_i' = x_i$, and the cost function c is strictly increasing in $\|x_i' - x_i\|_2$ and strictly convex in x_i'. Because Problem 2 is non-convex, so the objective of adversary can be formed to minimize an upper bound:

$$\min_{x_i'} Q(x_i') \equiv f(x_i') + c(x_i', x_i). \tag{3}$$

So the high-level idea of *cg* is to iteratively choose a feature, and greedily update this feature according to the partial derivatives of the attacker's objective as 3 to evade the classifier. Below, we take the exponential cost function $c(x_i', x_i) = \exp\left(\lambda(\sum_j (x_{ij}' - x_{ij})^2 + 1)^{1/2}\right)$ as an example to estimate the modification cost, which is also quite natural: options become exponentially less desirable to an attacker as they are more distant from their ideal attack. Then we take the following partial derivative to update the adversary's objective until the convergence.

$$\frac{\partial Q(x_i')}{\partial x_{ij}} = \frac{\partial f(x_i')}{\partial x_{ij}} + \frac{\partial c(x_i', x_i)}{\partial x_{ij}} = \frac{\partial f(x_i')}{\partial x_{ij}} + \frac{\lambda c(x_i', x_i)(x_{ij}' - x_{ij})}{(\sum_j (x_{ij}' - x_{ij})^2 + 1)^{1/2}},$$

To avoid the algorithm converges only to a locally optimal solution, random restarts strategy is applied to randomly select the starting points in the feature space. As long as a global optimum has a basin of attraction with positive Lebesgue measure, or the feature space is finite, this process will asymptotically converge to a globally optimal solution with enough random restarts.

Adam. Another adversary model $\mathcal{A}(adam)$ applies the stochastic gradient-based optimization algorithm Adam Kingma & Ba (2014) to generate adversarial examples. Specifically, the adversary uses Adam to solve the same optimization problem as in Equation 3.

3.2 Defender Model

Given the possible adversary models, several defensive strategies have been proposed focusing on different perspectives. Basically, the learner tries to integrate the prior knowledge of either the adversary model or the data distribution with the classification process. Here we consider different defensive strategies given the adversary model and form the interaction as a Stackelberg game. We will also consider the repeated game setting in section 4.4.

Adversarial Retraining framework (*RAD*). A systematic defensive approach based on adversarial retraining (*RAD*) has been proposed in Li et al. (2016). At the high level, *RAD* starts with the original training data and iteratively updating the learner with adversarial instances that evade the previously computed classifier until the convergence. It has been proved that the algorithm will terminate and the lower bound of the empirical loss of *RAD* is also provided. The important part for *RAD* is to select the adversarial retraining instances. In practice, it is hard to exactly estimate the adversary model as well as the parameters used within their model. Therefore, the generalization ability of *RAD* across different adversary models is quite important. Surprisingly, *RAD* generalizes quite well among various adversary models without requiring to know the exact attacker strategy. We will present the cross-model analysis for *RAD* in details in section 4.3.

AutoEncoder stacked with Classifier (*AEC*). One of the recent and efficient defensive method is the AutoEncoder stacked with a classifier to initialize deep architectures proposed in Gu & Rigazio (2014). To assess the structure of the adversarial noise, an autoencoder on mapping adversarial examples back to the original data samples is trained and stacked with the classifier. We train the AutoEncoder with different adversarial algorithms, including the fast gradient sign method (*fgs*), the coordinate greedy (*cg*) method, as well as Adam.

Improved AutoEncoder stacked with Classifier (*IAEC*). Since the baseline *AEC* cannot perform very well by only mapping the adversarial images back to the original image, we apply an improved AutoEncoder stacked with classifier (*IAEC*) defensive method. As AutoEncoder itself can not ensure that adversarial examples are denoised, we add a cross-entropy regularizer term as the loss function to help ensure that the output of AutoEncoder is classified correctly. Let y_i be the one-hot representation of ground truth label of an input instance x_i, then our loss function becomes:

$$J(x_i) = \|s(x_i) - {x_i}'\| + H(y_i, f(x_i)),$$

where $s(x_i)$ represents the mapping result of x_i by the AutoEncoder, and the cross-entropy function $H(y_i, f(x_i)) = -\sum\limits_{x_i} y_i \log f(x_i)$.

Distillation. Considering the fact that the knowledge extracted during training, which is in the form of probability vectors, and transferred in smaller networks to maintain accuracy comparable with those of larger networks can also be beneficial to improving generalization capabilities of deep neural networks outside of their training dataset, a defensive strategy against the adversarial examples has been proposed in Papernot et al. (2015). This defensive strategy transfers the knowledge contained in probability vectors through the distillation training step, then applies these probabilities in the next training step instead of using the original hard labels, and therefore enhances its resilience to perturbations. This defensive model is independent of the adversary models and we will evaluate its robustness and vulnerabilities in details in section 4.

4 Experimental Analysis

In this section, we empirically compare the adversarial retraining framework *RAD* with other state-of-the-art baseline methods Distillation Papernot et al. (2015), AutoEncoder stacked with Classifier (*AEC*) Gu & Rigazio (2014) and our improved AutoEncoder stacked with Classifier (*IAEC*) against various adversary models based on both MNIST and CIFAR-10 datasets.

Basically, we first analyze the robustness of *RAD* and Distillation, which performs the best against adversarial examples currently, by comparing the classification results before and after applying the adversarial retraining technique based on both MNIST and CIFAR-10 datasets. Then we estimate the cross-model classification robustness for *RAD*, *AEC*, the improved *IAEC*, and Distillation. Precisely, during the cross-model evaluation, we allow the attacker to generate the adversarial examples with different adversarial algorithms, while the defender has no clue about what adversarial algorithm is used. Therefore, we are able to evaluate the resilience of the "black-box" defensive strategies without requiring to know the actual adversary model.

Besides, we allow the attacker to attack these robustly enhanced learners and we compare the resilience of the *RAD* with the baseline defensive models and show that with retraining instances generated by adam, the *RAD* is almost unassailable for attacks based on the fast gradient sign method, which is promising to design universal defensive algorithms based on *RAD*.

Additionally, another perspective to measure the robustness of the learners is to evaluate how much noise is needed to make the learner misclassify an otherwise correct instance. As pointed out by Gu & Rigazio (2014), even a learner can be demonstrated to perform robustly against certain adversarial examples, it may become more vulnerable in the sense of being attacked by adding much smaller magnitude of adversarial noise. This means increasing the noticeability of the smallest adversarial noise for each example becomes the key to solve the adversarial examples problem. Therefore, we compare the malicious distortion required to attack each model, aiming to evaluate the vulnerability of different learners. The distortion is measured by $d(x_i{}', x_i) = \frac{1}{n}\sqrt{\sum(x_i{}' - x_i)^2}$, where $x_i{}' = \mathcal{A}(\beta, x_i)$ representing the adversarial manipulated instance based on arbitrary adversary model $\mathcal{A}$.

4.1 Experimental Setup

In our experiments, we focus on binary classification, and the adversary tries to modify a malicious instance (classified as +1) to evade the classifier and be classified as benign (-1). On MNIST, we select digit “4” as the malicious (positive) class, and “7” as the benign (negative) class. On CIFAR-10, we use “Airplane” as the malicious class, and “Cat” as the benign class. We use LeNet-5 LeCun et al. (1998) to perform the binary classification, and all classifiers used to evaluate the efficiency of different adversary models and defender models are based on this model architecture. All input pixel values are normalized into $[-0.5, 0.5]$.

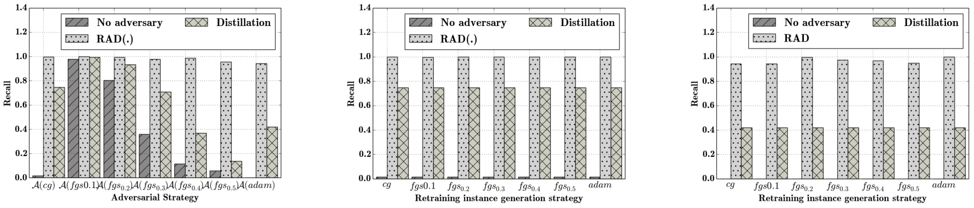
With respect to adversary models, during the evaluation, all of them modify the malicious instances in the original testset to evade the classifier, and keep the benign instances untouched. Meanwhile, for iterative attack methods evaluated in our experiments, i.e., *cg* and *adam*, according to our experiments, actually we can find adversarial examples with small modification cost using any λ, even when setting $\lambda = 0$, i.e., not considering the cost function $c(x_i{}', x_i)$ for optimization. Therefore, we set $\lambda = 0$ for all experiments using these two attack methods.

With respect to defensive models, for RAD, we only add adversarial examples generated on original malicious instances into the dataset for retraining, since the goal of adversary is trying to fool a classifier to label a malicious instance as benign, which follows the framework proposed in Li et al. (2016). As for AEC and IAEC, we use the same autoencoder architecture for removing adversarial noises proposed in Gu & Rigazio (2014), i.e., a three-hidden-layer autoencoder (784-256-128-256-784 neurons). We train the autoencoder to map adversarial examples generated on original malicious instances to the original images, and as suggested in Gu & Rigazio (2014), we also train the autoencoder to map original data back to itself. Both AEC and IAEC stack the autoencoder with a LeNet-5 classifier.

4.2 Robustness Analysis for Defensive Learners

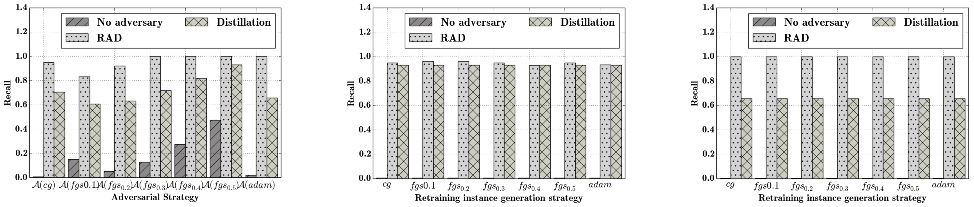
To evaluate the robustness and efficiency of the adversarial retraining framework and other defensive learners, we generate adversarial examples based on the the coordinate gradient algorithm (*cg*), adam, and the fast gradient sign algorithm (*fgs*$_\epsilon$) with the size of perturbation $\epsilon = 0.1 \sim 0.5$ (Goodfellow et al. (2014)), respectively. Figure 1 shows the analysis of recall for the traditional LeNet-5 and the robust *RAD* classifiers on MNIST. The test error of LeNet-5 on the original dataset is 0.045%. It is obvious that after the adversarial retraining process based on *RAD*, the classifiers perform nearly optimal. It is interesting to observe that with the ϵ of *fgs* increases, the adversarial examples generated by *fgs* can attack the original LeNet-5 more efficiently.

Figure 2 presents the comparisons of recall for the original LeNet-5 and the adversarial retraining framework on CIFAR-10. It shows that the adversarial retraining framework works robustly against different adversarial example generation methods. Note the test error of LeNet-5 on the original dataset is 5.5%. From the results of recall, we can see that almost all the “generated” adversarial instances are correctly classified by the retraining framework. Additionally, sometimes the test error of *RAD* is even smaller than that of the original model LeNet-5 based on the uncontaminated (no adversary) data. This means, with the adversarial robust retraining process, some “blind-spots” in the input space volume can be filled out without decreasing the performance on the normal test data. Moreover, surprisingly, with the increase of ϵ, the fast gradient sign method works worse for generating adversarial examples against LeNet-5, which is different for MNIST. This is actually caused by the properties of the fast gradient sign method itself. By following the gradient, the generated instance can be trapped into sub-optimal and therefore fail to converge to the global optima,



(a) (b) (c)

Figure 1: Performance of retraining with instances generated from different models based on MNIST. (a) The retraining instances are generated by cg; (b) the adversarial examples are generated by cg; (c) the adversarial examples are generated by adam.



(a) (b) (c)

Figure 2: Performance of retraining with instances generated from different models based on CIFAR-10. (a) The retraining instances are generated by CG; (b) the adversarial examples are generated by CG; (c) the adversarial examples are generated by adam.

so different step size can affect their final convergence. Therefore, by comparing with the results of MNIST, we can see learners on CIFAR-10 is easier to be trapped by the sub-optima and larger ϵ values can lead the learner to be trapped into these points with higher probability. On the other hand, no matter how much the strength of adversarial ability is affected by different parameters, the adversarial retraining framework works robustly by almost identifying all the manipulated instances correctly on different datasets consistently.

4.3 Cross-model analysis for different defensive learners

Aiming to defend a more broad class of attacks, here we assume the the learner has no clue about which adversarial algorithm the attacker uses to generate the adversarial instances. Therefore, the defender can perform robustly as the "black-box" learner against arbitrary adversaries. Here we use different attack algorithms to generate the adversarial examples, and the retraining instances for *RAD* are also generated across various adversary models to evaluate the learners' generalization ability. We also compare the results with the state-of-the-art Distillation, *AEC* and our improved *IAEC* algorithm based on different adversarial models. Here the *AEC* is trained on the adam model, which offers the best classification results. The *IAEC* is also trained corresponding to different adversary models to compare the cross-model learning ability with *RAD*. Table 1 shows the test error comparisons for these cross-model learners. "No adversary" presents the test error of different learners on the clean data. Basically, the adversarial retraining framework performs consistently better than *AEC*, *IAEC*, and Distillation on all different adversarial examples in terms of the classification error. This conclusion is independent of what models are used to generate adversarial retraining instances for *RAD*. Based on the results, the adversarial retraining framework has the potential to be applied to defend against any arbitrary attacks without requiring to know the exact adversary model. Based on the classification error results for our improved *IAEC* in Table 1, it is obvious that the *IAEC* with the same adam adversary model works much more robust than *AEC*. This means the proposed *IAEC* is much more robust compared with the original *AEC* by adding the cross-entropy regularization. Additionally, we also evaluate the cross-model classification error for *IAEC* to test its generalization ability. Table 1 shows that the *IAEC* can also defend against different adversarial examples without requiring to know the exact adversary model.

Table 1: Classification error of different learners against various adversary models based on MNIST

Model	$\mathcal{A}(fgs_{0.1})$	$\mathcal{A}(fgs_{0.5})$	$\mathcal{A}(cg)$	$\mathcal{A}(adam)$	No adversary
LeNet-5	1.2%	46.1%	48.2%	48.9%	0.045%
RAD$(fgs_{0.1})$	0.1%	0.5%	0.4%	3.0%	0.045%
RAD$(fgs_{0.5})$	0.5%	0.1%	0	2.5%	0.045%
RAD(cg)	0.1%	1.4%	0.4%	2.9%	0.045%
RAD$(adam)$	0	0.1%	0.1%	0.1%	0.045%
AEC$(adam)$	3.2%	20.6%	9.7%	2.6%	4.5%
IAEC$(fgs_{0.1})$	1.3%	28.0%	18.3%	9.6%	1.1%
IAEC$(fgs_{0.5})$	1.2%	1.4%	2.6%	5.5%	1.0%
IAEC(cg)	1.6%	1.6%	1.5%	7.4%	1.2%
IAEC$(adam)$	1.2%	5.2%	7.3%	2.3%	1.7%
Distillation($T = 1$)	0.6%	47.2%	29.4%	41.9%	0.2%
Distillation($T = 100$)	0.3%	42.3%	12.4%	28.5%	0.2%

Similarly, we show the classification error comparison results of *RAD* across different adversary models in Table 2 compared with Distillation. As CIFAR-10 images are more complex, the error rates for adversarial retraining framework get larger compared with that on MNIST. However, overall the classification error for the retraining framework on different adversarial examples are below 13% with zero knowledge of the adversary model, while the classification error on normal data is around 6%. Therefore, even on CIFAR-10 dataset, the adversarial retraining framework is still promising to perform the "black-box" defending resiliently against various attacks. Additionally, the distillation with $T = 1$ and $T = 100$ both encounter higher test error than *RAD*, even the distillation method performs more robustly when $T = 100$ than $T = 1$.

Table 2: Comparisons for the error rate of RAD based on different adversary models on CIFAR-10

Model	$\mathcal{A}(fgs_{0.1})$	$\mathcal{A}(fgs_{0.5})$	$\mathcal{A}(cg)$	$\mathcal{A}(adam)$	No adversary
LeNet-5	1.2%	46.1%	54.0%	52.7%	5.5%
RAD$(fgs_{0.1})$	2.35%	2.0%	4.65%	3.0%	5.3%
RAD$(fgs_{0.5})$	4.4%	2.7%	5.6%	2.6%	5.8%
RAD(cg)	7.5%	2.45%	5.05%	2.2%	5.7%
RAD$(adam)$	16.2%	2.8%	6.15%	2.4%	5.9%
Distillation($T = 1$)	21.3%	30.8%	13.8%	22.0%	11.0%
Distillation($T = 100$)	19.3%	25.2%	9.2%	20.2%	7.2%

4.4 Robustness Against additional Attacks

In order to test the robustness of the learner against the repeated attacks, where the attacker can again conduct attacks on the robust learners, here we evaluate how the robust learner behaves given additional attacks based on different adversary models. Table 3 presents the test error rate comparison when the attacker generates adversarial examples to attack the robust *RAD* learner, *IAEC*, and Distillation on MNIST. It is shown that the coordinate greedy (cg) and adam are somehow efficient to attack *RAD*, while the fast gradient sign methods fail to attack the robust *RAD*. So if the *RAD* is retrained with instances generated by arbitrary adversary models, it can be resilient against adversarial examples produced by the fast gradient sign method with various ϵ values. This means the *RAD* can confer robustness to single-step attack methods but not the iterative ones. However, adversaries based on cg and adam can still find the vulnerabilities to attack the model. Compared with the performance of the adversarial retraining framework (*RAD*) against these "repeated attacks", the *IAEC* encounters much higher classification error when being attacked. This indicates that the adversarial retraining framework can not only enhance the resilience of the original learner (LeNet-5), but also perform robustly against the additional attacks compared with the *IAEC*.

Similarly, Table 4 presents the test error for attacking different robust learners with various adversary models on CIFAR-10. *RAD* again produces lower test error compared with Distillation ($T = 1$,

Table 3: Error rate of attacking the robust learners with additional attacks on MNIST

Model	$\mathcal{A}(fgs_{0.1})$	$\mathcal{A}(fgs_{0.5})$	$\mathcal{A}(cg)$	$\mathcal{A}(adam)$
RAD$(fgs_{0.1})$	0.3%	9.6%	48.1%	49.0%
RAD$(fgs_{0.5})$	0.8%	0.1%	45.7%	49.0%
RAD(cg)	0.8%	3.4%	44.6%	49.0%
RAD$(adam)$	0.1%	0.1%	40.2%	48.7%
IAEC$(fgs_{0.1})$	4.2%	10.3%	49.9%	49.5%
IAEC$(fgs_{0.5})$	5.2%	3.8%	49.8%	49.9%
IAEC(cg)	5.3%	3.9%	49.9%	49.4%
IAEC$(adam)$	4.6%	7.0%	49.9%	49.9%
Distillation($T = 100$)	0.2%	0.2%	49.0%	48.7%

Table 4: Error rate of attacking the robust learners with additional attacks on CIFAR-10

Model	$\mathcal{A}(fgs_{0.1})$	$\mathcal{A}(fgs_{0.5})$	$\mathcal{A}(cg)$	$\mathcal{A}(adam)$
RAD$(fgs_{0.1})$	3.7%	2.7%	42.0%	52.7%
RAD$(fgs_{0.5})$	5.3%	2.8%	49.0%	52.4%
RAD(cg)	7.9%	2.8%	52.0%	52.7%
RAD$(adam)$	6.3%	3.1%	54.0%	52.7%
Distillation($T = 100$)	9.05%	8.6%	54.0%	54.1%

$T = 100$) given diverse adversarial attacking strategies. What is worth to mention is that these robust learners all perform accurately on the normal dataset without adversarial manipulation, which offers more potentials for the robust learners.

4.5 Vulnerability of the defensive learners

Given the fact that the attacker can attack the learning model continuously, here we are concerned with how vulnerable the robust models become in terms of the amount of distortion needed to add to mislead the learner. We compare the average distortion for attacking the LeNet-5, *RAD*, *IAEC*, and Distillation to evaluate their robustness. As mentioned by Gu & Rigazio (2014), *AEC* demands smaller distortion to attack, which means *AEC* is quite fragile, and we also gain the similar observation and confirm that attacking the original LeNet-5 model requires larger magnitude of noise than *AEC*. Thus, we focus on the improved *IAEC*.

In the Table 5 we present the demanded distortion to maliciously attack the *RAD*, the *IAEC*, and Distillation on MNIST. Note that the fast gradient sign method here is a one-step method, which will stop after computing one gradient to find the optimal perturbation of a linear approximation of the cost or model, so it cannot guarantee to find the evasion instance x_i' and we do not consider its distortion. so here we only consider cg and adam to generate distortions. We use *RAD*(.) to represent the adversarial retraining framework retrained with arbitrary adversarial instances since they all require the same amount of distortion to be attacked given their similar network structures. From Table 5 *RAD* requires the same distortion as attacking the original LeNet-5 model. However, the distortion needed for attacking the *IAEC* is substantially smaller than that for attacking the original models. From this perspective, the *IAEC* becomes more vulnerable compared with the original model even though it can be resilient against the adversarial examples. Similar for Distillation, smaller distortion is demanded to attack the robust learner, which means more vulnerabilities are introduced by the robust Distillation. On the contrary, the adversarial retraining framework *RAD* can perform robustly against various diverse adversarial attacks without increasing the vulnerability penalty.

Figure 3 shows the results of adding the corresponding adversarial noise to generate the misclassification for LeNet-5 model by different adversarial algorithms qualitatively. It shows that by using fast gradient sign method with $\epsilon = 0.5$, the original image is almost distorted. This indicates different adversary models have different attacking strengths, so taking the stronger adversary model into

Table 5: Adversarial distortion required for attacking different models on MNIST

Model	$\mathcal{A}(cg)$	$\mathcal{A}(adam)$
LeNet-5	0.0118	0.0060
RAD(.)	0.0118	0.0060
IAEC$(fgs_{0.1})$	0.0042	0.0031
IAEC$(fgs_{0.5})$	0.0058	0.0028
IAEC(cg)	0.0069	0.0023
IAEC$(adam)$	0.0064	0.0029
Distillation($T = 100$)	0.0106	0.0060

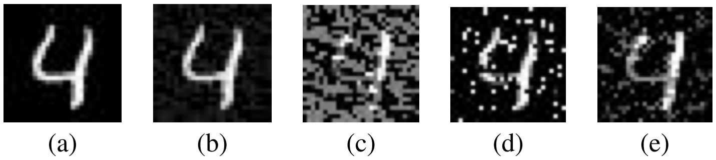


Figure 3: Visualization of adversarial examples generated by different attacker models based on MNIST. (a) Original image, (b) attacked by $fgs_{0.1}$, (c) attacked by $fgs_{0.5}$, (d) attacked by cg, (e) attacked by adam.

account may have a chance to defend the weaker adversaries, which makes the universal defensive model promising.

Similarly, Table 6 lists the amount of distortion needed to fool the original learner based on CIFAR-10. It is shown that both the *RAD* and Distillation need exactly the same amount of distortion with the original LeNet-5 model, which means these robust learners do not increase the vulnerability of the original model.

Table 6: Adversarial distortion required for attacking different models on CIFAR-10

Model	$\mathcal{A}(cg)$	$\mathcal{A}(adam)$
LeNet-5	0.0025	0.0015
RAD(.)	0.0025	0.0015
Distillation($T = 100$)	0.0025	0.0015

The visual attacking results by injecting malicious noise are shown in Figure 4. It is clear that *fgs* with $\epsilon = 0.5$ can distort the original images the most compared with other adversary algorithms. Surprisingly, all the retraining framework based on different retraining instances only get the classification error lower than 3.0%.

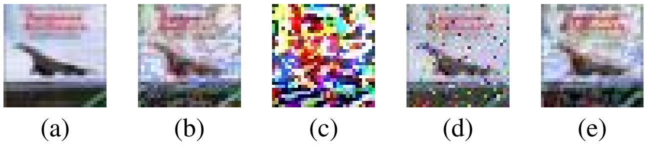


Figure 4: Visualization of adversarial examples generated by different attacker models based on CIFAR-10. (a) Original image, (b) attacked by $fgs_{0.1}$, (c) attacked by $fgs_{0.5}$, (d) attacked by cg, (e) attacked by adam.

5 CONCLUSION

To understand the adversarial examples better, as well as the potential adversary models and corresponding defensive learners, we conduct extensive experiments to evaluate properties of different defensive strategies. We point out that *RAD* works the best among all the defensive strategies against different adversary models, including one-step and iterative ones, in terms of the classification test error. The adversarial retraining framework, *RAD*, also generalizes well for the cross-model evaluation compared with *AEC*, *IAEC*, and Distillation. Moreover, both *RAD* and Distillation do not introduce additional vulnerability penalty to the original models, while still increase the robustness. So in the future work, to generalize the robust learner across different adversary models, one direction could be to generate retraining instances based on diverse adversarial algorithms to cover as much as possible the "blind-spots" within the input space. In addition, we will dynamically optimize the choice of adversary model and the quantity of retraining instances according to the robustness requirements of a specific learner. Therefore, the tradeoff between robustness and accuracy on the normal data can be balanced based on the specific resilience demand of the learner.

References

Battista Biggio, Giorgio Fumera, and Fabio Roli. Security evaluation of pattern classifiers under attack. *Knowledge and Data Engineering, IEEE Transactions on*, 26(4):984–996, 2014.

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

Shixiang Gu and Luca Rigazio. Towards deep neural network architectures robust to adversarial examples. *arXiv preprint arXiv:1412.5068*, 2014.

Ruitong Huang, Bing Xu, Dale Schuurmans, and Csaba Szepesvari. Learning with adversary. *arXiv preprint arXiv:1511.03034*, 2015.

JC Jan, Shih-Lin Hung, SY Chi, and JC Chern. Neural network forecast model in deep excavation. *Journal of Computing in Civil Engineering*, 16(1):59–65, 2002.

Jonghoon Jin, Aysegul Dundar, and Eugenio Culurciello. Robust convolutional neural networks under adversarial noise. *arXiv preprint arXiv:1511.06306*, 2015.

Liyiming Ke, Bo Li, and Yevgeniy Vorobeychik. Behavioral experiments in email filter evasion. In *AAAI Conference on Artificial Intelligence*, 2016.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Bo Li and Yevgeniy Vorobeychik. Feature cross-substitution in adversarial classification. In *Advances in Neural Information Processing Systems*, pp. 2087–2095, 2014.

Bo Li, Yevgeniy Vorobeychik, and Xinyun Chen. A general retraining framework for scalable adversarial classification. *arXiv preprint arXiv:1604.02606*, 2016.

Daniel Lowd and Christopher Meek. Adversarial learning. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pp. 641–647. ACM, 2005.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, and Shin Ishii. Distributional smoothing with virtual adversarial training. *stat*, 1050:25, 2015.

Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 427–436. IEEE, 2015.

Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. *arXiv preprint arXiv:1511.04508*, 2015.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016a.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against deep learning systems using adversarial examples. *arXiv preprint arXiv:1602.02697*, 2016b.

Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 372–387. IEEE, 2016c.

Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J Fleet. Adversarial manipulation of deep representations. *arXiv preprint arXiv:1511.05122*, 2015.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Pedro Tabacof and Eduardo Valle. Exploring the space of adversarial images. *arXiv preprint arXiv:1510.05328*, 2015.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pp. 1096–1103. ACM, 2008.

Yevgeniy Vorobeychik and Bo Li. Optimal randomized classification in adversarial settings. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pp. 485–492. International Foundation for Autonomous Agents and Multiagent Systems, 2014.

Stephan Zheng, Yang Song, Thomas Leung, and Ian Goodfellow. Improving the robustness of deep neural networks via stability training. *arXiv preprint arXiv:1604.04326*, 2016.