Towards Fast and Robust Adversarial Training for Image Classification

Erh-Chung Chen and Che-Rung Lee

National Tsing Hua University, Hsinchu, Taiwan
`s107062802@m107.nthu.edu.tw, crlee@cs.nthu.edu.tw`

Abstract. The adversarial training, which augments the training data with adversarial examples, is one of the most effective methods to defend adversarial attacks. However, its robustness degrades for complex models, and the producing of strong adversarial examples is a time-consuming task. In this paper, we proposed methods to improve the robustness and efficiency of the adversarial training. First, we utilized a re-constructor to enforce the classifier to learn the important features under perturbations. Second, we employed the enhanced FGSM to generate adversarial examples effectively. It can detect overfitting and stop training earlier without extra cost. Experiments are conducted on MNIST and CIFAR10 to validate the effectiveness of our methods. We also compared our algorithm with the state-of-the-art defense methods. The results show that our algorithm is 4-5 times faster than the previously fastest training method. For CIFAR-10, our method can achieve above 46% robust accuracy, which is better than most of other methods.

1 Introduction

Deep neural networks have demonstrated a strong capability to solve many challenging computer vision tasks, such as classification [1], object detection [2] and image captioning [3]. These successful achievements not only shift the paradigm of AI researches, but also enable many useful applications, such as self-driving car [4] and medical image analysis [5].

However, the current neural network models for image classification are vulnerable to the adversarial attack, which means slight perturbations in the input data can significantly degrade the accuracy of classification, even though those perturbed images are indistinguishable from the original ones by human's eyes. Adversarial attacks can be a potential threat to real applications, such as the cell-phone camera attack [6] and the road sign attack [7]. Hence, designing models that can have better resistance to adversarial attacks is one of the most important tasks to make practical AI applications.

Adversarial attacks have many different forms. For white-box attacks, such as L-BFGS [8], FGSM [9], and CW attack [10], attackers can access complete information about the target networks, including the architecture, model parameters, or even the training data, from which the adversarial examples can be generated. For black-box attacks, attackers have little knowledge about the target models, but they can still fool the target models, even with one pixel modification [11]. Moreover, the adversarial examples are transferable, which means an adversarial example generated by one model can fool different models. As a result, attackers can make use of this property to generate universal adversarial examples [12].

In this paper, we focus on the defense of the white-box attacks with epsilon bound on L_∞ norm since major attacks of this type generate adversarial examples from the gradient of the loss function [9]. Many attempts of defenses try to eliminate the gradient information during the training, however, the paper [13] showed those types of defenses are unworkable. Another strategy is adversarial training, which adds adversarial examples into training data. Despite its safety and efficacy, adversarial training needs strong enough adversarial examples to avoid the over-fitting to the perturbations. Nevertheless, searching powerful adversarial examples is a time consuming task.

Many defense strategies have been proposed, such as gradient regularization [14], TRADES [15] and feature denoising [16]. Fast training methods have been proposed as well [17–19]. Besides, recent works have investigated how robustness can achieve on large scale datasets [16, 20, 21]. We propose a new training architecture to enhance the robustness of the adversarial training, and design a new method for producing adversarial examples to accelerate the training speed. The proposed training architecture concatenates a re-constructor to the classifier, whose objective is to produce an identical image to the original clean image. The purpose of such architecture is to force the classifier not only to output a correct label for each input image, but also to project adversarial images onto a manifold, on which the inter-class distance can be reduced. With such kind of training model, even a weaker model can learn non-trivial classifiers with good robustness against adversarial attacks.

Our adversarial images generator is called enhanced FGSM (Fast Gradient Sign Method [9]), which can produce strong attacks without computing them from iterative PGD. The signSGD algorithm and its variations have been used for different applications in machine learning, such as distributed learning [22] and the producing of adversarial examples for black-box attacks [23]. FGSM can generate a good initial point for adversarial examples [17], but it suffers the catastrophic overfitting, which can slow down the train process. With the mechanism to check the robust accuracy during training process, we can terminate the training earlier and avoid the catastrophic overfitting.

We evaluated our method using MNIST and CIFAR10 datasets, and compared our methods with the state-of-the-art defense algorithms proposed by [15, 18, 24]. The experimental results show that our methods are about 5 times and 4 times faster than AdvFree [18] for MNIST and CIFAR10 respectively. In addition, for MNIST, our method can achieve better accuracy under projection gradient decent attacks compares to AdvFree. And for CIFAR10, our method can achieve 46.06% robust accuracy. If the enhanced FGSM is combined with TRADES, the robust accuracy of CIFAR10 can be 48.05%.

Our main contributions are summarized as follows:

- We investigate the importance of inter-class distance.
- We propose a new training architecture with a lightweight attack, which reduces heavy computing cost.
- We show that our model's capacity is much smaller than competitor but have similar or better accuracy.

The rest of this paper is organized as follows. Section 2 introduces the necessary background about the adversarial attacks and defense methods. Section 3 presents our algorithms, and model architecture. Section 4 shows the experimental results and the discussions. Conclusion and future work are given in the last section.

2 Preliminary

This section introduces notations and describes mathematical background of adversarial attacks and defenses.

Given an input image x, whose value is defined in the domain $[0,1]^{W\times H\times C}$, the classifier outputs one of the indices in a set of labels $\{c_1, c_2, \dots, c_k\}$. The classification process can be represented as a function,

$$F(x)_i = \mathrm{softmax}(Z^o(x)), \tag{1}$$

where $F(x)_i$ represents the probability of that image x is labeled with class i and $Z^o(x) \in R^k$ is the output of the last layer of the network before the softmax layer,

$$\mathrm{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}}, \tag{2}$$

that normalizes the output $Z^o(x)$ to a probability distribution over predicted output classes. The network predicts that x belongs to label t by one of the flowing two functions,

$$\begin{cases} F(x) = \arg\max F(x)_i, \\ H(x) = \arg\max Z^o(x)_i. \end{cases} \tag{3}$$

The training of the classifier can be viewed as a minimization process of the loss function $L(x, y)$ over a given training dataset X, where $x \in X$ and y is the corresponding label for the image x.

2.1 Formulating Adversarial Attack

An adversarial example is an image with indistinguishable perturbation that causes model to make incorrect prediction. An adversarial example is the solution of the following optimization problem:

$$\min ||\delta||_p, \text{ subject to } F(x+\delta) \neq F(x). \tag{4}$$

We can define conditions in Equation (4) as a constraint set Ω,

$$\Omega = \{x | F(x)_t - \arg\max_{i \neq t} F(x)_i \leq 0\}. \tag{5}$$

In [9], authors first proposed a method, called Fast Gradient Sign Method (FGSM), to generate adversarial examples x^{adv} as follows,

$$x^{\text{adv}} = \mathcal{P}_{[0,1]}(x + \alpha \text{sign}(\nabla L(x, y))) \tag{6}$$

where $L(x, y)$ is the loss function, α is the step size, and $\mathcal{P}_{[0,1]}$ is a projection function to ensure that x^{adv} is a valid image. The major idea is to add perturbations along the direction of sign$(\nabla L(x, y))$, such that image could be misclassified.

The attack can be a small perturbation on the original image. One way to characterize this property is to constrain the size of perturbation to a small range, say ϵ. To satisfy the ϵ constrain, the Projection Gradient Decent (PGD) algorithm is usually applied to the generate valid perturbations. PGD runs for T iterations. In each iteration i, it follows the Equation (6):

$$x_{i+1}^{\text{PGD}} = \mathcal{P}_{[0,1]}(x_i^{\text{PGD}} + \alpha \text{sign}(\nabla L(x, y))) \tag{7}$$

and applies projection operator on x_{i+1}^{PGD} such that $\|x_{i+1}^{\text{PGD}} - x\|_\infty < \epsilon$. If a random start is allowed, noise from $-\epsilon$ to ϵ will be added in the beginning or at each step.

For FGSM attack, α in (6) is set to ϵ, so x^{adv} moves to the boundary immediately; for PGD attack, x_{t+1}^{PGD} moves to the boundary by T steps, so α in (7) is ϵ/T which is smaller than ϵ. Therefore, PGD is able to make more powerful adversarial images.

2.2 Linear Approximation and DeepFool

FGSM and PGD discussed in Section 2.1 can find feasible solutions of the Equation (4), but the solutions may not be optimal. The geometric meanings of the optimal δ in Equation (4) is the smallest step toward to the boundary of the constraint set Ω.

In the paper of DeepFool [25], the authors argued that the optimization problem can be linearized by Taylor expansion, by which a better solution with smaller perturbation than that of FGSM and PGD can be found. In DeepFool, the constraint set Ω is replaced with another equivalent set $\hat{\Omega}$,

$$\hat{\Omega} = \{x | Z^o(x)_t - \arg\max_{i \neq t} Z^o(x)_i \leq 0\}. \tag{8}$$

If $Z^o(x)$ is an affine function, problem (8) has a closed-form solution,

$$\min_{i \neq t} \frac{|Z^o(x)_t - Z^o(x)_i|}{||\nabla Z^o(x)_t - \nabla Z^o(x)_i||_q}, \tag{9}$$

where p and q follow Holder's Inequality's constraint $1/p + 1/q = 1$. The detail of the proof can be found in [25].

By linearizing the lost function of a network, we can use the formula in (9) as a solution to (8). The error term depends on how close the initial point is to the decision boundary and the magnitude of the Hessian. Principally, the latter is more important than the former. Nevertheless, the magnitude of the Hessian is difficult to estimate even if function is Lipschitz continuity [26]. A practical strategy is trying to regularize gradient which implicitly reduce the upper bound [14].

2.3 Adversarial Training

The adversarial training, which injects adversarial examples into the training dataset. The state-of-the-art method was proposed by Madry [24], in which they showed that the adversarial training is to solve the following min-max problem:

$$\min_{\theta} \max_{x':D(x',x)<\epsilon} L(x', y; \theta) \tag{10}$$

The maximization problem is to search the strongest adversarial examples; while the minimization problem is to minimize the adversarial loss given by inner attacks. They also argued that adversarial attacks generated by PGD is the strongest first-order attack, which means if a model is robust enough against the PGD attack, it will be defensive against any gradient-based attack. Therefore, PGD attack is a best choice for the inner attack.

To reduce the heavy computation of adversarial training, Adversarial training for Free (AdvFree) [18] calculates mini-batch's one-step adversarial images with step size ϵ immediately during backward phase, re-uses adversarial images in the next forward phase, and repeats such loop m times. Compared with the standard adversarial training, which requires m times forward and backward phases to generate adversarial images, AdvFree needs no extra cost. However, AdvFree is not exactly equal to PGD attack [24], because it chooses ϵ as step size instead of $\frac{\epsilon}{T}$ and moves to the boundary in one step. If ϵ is large, AdvFree cannot compute an accurate adversarial images during iterations. On the other hand, if AdvFree chooses a smaller step size, attacks at each step would not be strong enough, except the last step. As a result, AdvFree takes longer time to converge.

2.4 Gradient Masking

Since most attacks are based on the gradient information of target networks, the gradient masking methods defend the attacks by making the gradient indeterminable. However, the gradient masking methods are vulnerable to other types of attacks other than the gradient based ones [13]. On the other hand, if a model can defend the gradient based attacks without using gradient masking methods, it can resist almost all kinds of attacks, as shown in [13]. Therefore, it is important to ensure that no gradient masking is used explicitly or implicitly in the design of the defending method.

In [13], authors enumerated some rules to judge whether the gradients information of a model is hidden or not. If a model satisfies one or more properties listed below, it may use the gradient masking methods.

1. The random sampling method can find adversarial examples but gradient based attacks cannot.
2. One-step PGD attacks have better performance than iterative PGD attacks.
3. Increasing the distortion bound does not decrease robust accuracy.
4. Unbounded PGD attacks do not reach 100% success.
5. Black-box attacks are better than white-box attacks.

We are going to scrutinize that the above phenomena do not occur on our trained model.

3 Model and Algorithm

This section introduces the training architecture and the adversarial image generator.

3.1 Adversarial Image Generator

Our adversarial image generator is based on the FGSM method, because of its computational advantages over the optimization based algorithms. However, it is well-known that models trained by FGSM attacks can easily overfit to adversarial images, especially for a large ϵ [6, 24]. This phenomenon is called catastrophic over-fitting, where the model's robust accuracy drops to 0% suddenly during training phase.

The root cause of the overfitting problem is that the adversarial examples generated by FGSM cannot fully represent the ϵ-ball attacks [24]. There are two major problems of FGSM. First, the direction of FGSM x^{FGSM} is obtained from $\nabla_x L(x)$, along which the loss function $L(x)$ increases most. However, such direction is not the solution to the problem (5), because it does not consider the inter-class relation. Second, the step size of the FGSM is a fixed value, which cannot represent the attacks of smaller sizes.

In this paper, we proposed a fast adversarial image generator, called enhanced FGSM (eFGSM), which modifies FGSM to generate a better solution for problem (5). eFGSM has two steps. First, we will use the direction generated by FGSM as an initial point, because as shown in [17], FGSM still gives an approximate solution to the problem (5).

$$x^{\text{eFGSM}} = \mathcal{P}_{[0,1]}(x + \mathcal{P}_{[-\epsilon,\epsilon]}(\kappa \ \text{sign}(\mathit{\Gamma}))), \tag{11}$$

where κ and $\mathit{\Gamma}$ are the magnitude and the estimated attack respectively. Second, a more accurate adversarial example x^{adv} is computed with linear approximation $\mathit{\Delta}$ and x^{eFGSM},

$$\begin{aligned} \mathit{\Delta} &= \min_{i \neq t} \frac{|Z^o(x^{\text{eFGSM}})_t - Z^o(x^{\text{eFGSM}})_i|}{||\nabla Z^o(x^{\text{eFGSM}})_t - \nabla Z^o(x^{\text{eFGSM}})_i||_1} \\ x^{\text{adv}} &= \mathcal{P}_{[0,1]}(x^{\text{eFGSM}} + \text{sign}(\nabla Z^o(x^{\text{eFGSM}}))\mathit{\Delta}). \end{aligned} \tag{12}$$

This is because the direction of linear approximation can solve the problem (5), as shown in Section 2.2.

For each image, eFGSM generates a quick perturbation with the majority vote version of FGSM [22]. Equation (11) is similar to the original FGSM, except two modifications. First, Γ is calculated from the gradient direction obtained in each epoch,

$$\Gamma^{j+1} = 0.95\Gamma^{j} + \text{sign}(x^{adv} - x). \tag{13}$$

where the Γ^{j+1} at $j+1$th epoch is a weighted accumulated sum of gradients. Compared with the single gradient direction used in FGSM, Γ will stick on the weighted gradient direction of each pixel and hence produce stronger attacks. Second, κ is obtained from the Gaussian distribution with the mean equal to $\text{sign}(\Gamma)\epsilon$,

$$\kappa = \mathcal{N}(\text{sign}(\Gamma)\epsilon, \sigma^2). \tag{14}$$

And the variance σ is a hyperparameter to be decided. By doing so, the generated adversarial examples can better represent the attacks than those generated by FGSM.

The formulation of Γ can be considered as Bagging [27], which produces more accurate decisions than those produced by a single targeted model, even though each attack is weak and noisy. However, we need not generate multiple models for each epoch. Instead, like the idea proposed in [18], we only compute the gradient of the trained model during backward phase, and store the cumulative signs of gradients, so that Γ can be updated in each epoch. The updated Γ can be used to generate the adversarial images in the next epoch. Comparing to AdvFree [18], our method is more flexible and converges faster.

The design of κ is a heuristic strategy. Comparing with Madry's PGD with random start, which selects a random point in ϵ-ball as initial point. Instead, Γ gives us a hint of gradient information. To start from a random point without losing much gradient information and keeping diversity, we suggest that κ is generated by Gaussian distribution with mean equal to $\text{sign}(\Gamma)\epsilon$.

3.2 Inter-Class Distance

We argue that network's vulnerabilities come from one-hot encoding. In categorical classification problems, labels are usually encoded as one-hot vectors, and cross entropy loss encourages network to fit one-hot encoding labels. Thus, it forces each class to be mutually orthogonal to each other. The disadvantage of the one-hot encoding is ignoring the relationship among classes. For example, image 0 and image 6 on MNIST have similar stroke, and there are ambiguous images between those two classes. On the contrary, the difference between image 0 and 7 on MNIST is significant.

To solve this issue, the label smoothing technique was proposed [28], which is a regularization technique avoiding classifier predicting a too confident result. In [28], authors suggested that this skill helps networks' pre-softmax layer to get better representation.

Equation (9) suggests inter-class distance should also be taken into consideration in the design of a robust classifier. If $|\Delta|$ is greater than ϵ, no feasible adversarial examples exist for the ϵ-ball attack. In other words, the perturbation reaches the boundary of ϵ-ball or physical constraint($x \in [0, 1]$) before reaching decision boundary. To enlarge Δ as much as possible, we can either increase the magnitude of numerator or decrease the magnitude of denominator of Equation (9). In the L_2 norm attack for an affine classifier, the denominator is

$$\begin{aligned}&\arg\min_{i \neq t} ||W_t - W_i||_2 ||\nabla Z^{o-1}(x)||_2 = \\ &\arg\min_{i \neq t} (|W_t|^2 + |W_i|^2 - 2|W_t||W_i|)||\nabla Z^{o-1}(x)||_2,\end{aligned} \tag{15}$$

where W is the last layer's weights and Z^{o-1} is the output of the second last layer. If two classes are orthogonal, $-2|W_t||W_i|$ will be 0 and Δ will be minimum. Moreover, this result can be extended to L_∞ norm attack as well; for L_1 norm attack, denominator is $\| \cdot \|_\infty$, whose magnitude cannot be reduced efficiently. Thus, defending against L_1 norm attacks is still problematic [29].

Inter-class distance provides another benefits to check the occurrence of catastrophic overfitting. We can roughly estimate the distance between x^{adv} and the decision boundaries during the training phase without extra cost. That means we can quickly check the robust accuracy without the PGD-k attack. An image x is considered as robust if the minimal distance is always greater than ϵ. Catastrophic overfitting can be detected once robust accuracy at this epoch is tremendously decreased. If such case happens, the training procedure should be terminated immediately to avoid the catastrophic overfitting.

3.3 Training Model

In a seminal paper [24], authors had pointed out the characters of the defense models for effective adversarial training. One of them is "*weak models may fail to learn non-trivial classifiers*". However, as mentioned in Section 3.2, no strong evidences suggest that model's capacity is related to robustness. The inter-class distance should be the key point. To show that, we employ the idea of denoising auto-encoder [30] to design the training architecture. A denoising auto-encoder concatenates an encoder with a re-constructor, whose objective is to force the encoder to project the noisy data back onto a lower dimension manifold where the clean data reside. Because on the manifold, the noisy data are more clustered to the clean data, a weak model can still learn non-trivial classifiers.

The proposed architecture is presented in Figure 1, which consists of two networks: a classifier C and a re-constructor R. The network has three inputs: $\hat{x}$ is an adversarial example, x is the original image, and y is the ground-truth label with label smoothing. When receiving an input image $\hat{x}$, the classifier outputs a label $\hat{y}$, which is a vector showing the probability of each category for the input image. The output $\hat{y}$ is then fed into the re-constructor R, whose goal is to produce an identical image to x. The loss function of the network contains two

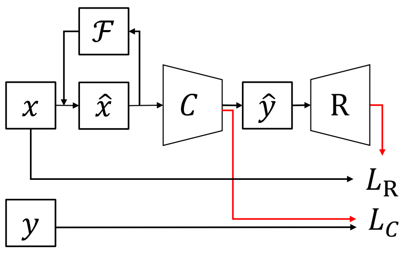

Fig. 1. The network architecture for training. y is ground-label with label smoothing, $\hat{x}$ is adversarial image generated by input image x and $\mathcal{F}$ is adversarial image generator which collecting gradient infomation from $\hat{x}$. For inference, only the classifier C is used.

terms: L_G and L_C, which are reconstruction loss and categorical loss respectively. Loss function is formulated as follows:

$$\min_{\theta} L_R(\hat{x}, x; \theta) + \lambda \cdot L_C(\hat{x}, y; \theta). \tag{16}$$

The goal is to find parameters θ that minimizes the risk of misclassification. The categorical loss L_C, which is also considered as sparse regularization, helps the classifier to output the correct label for the input image $\hat{x}$, even with noises. The reconstruction loss L_R enlarges the misclassified images' penalty. Because the latent code $\hat{y}$ is totally different from y, the re-constructor R is not able to generate the images which belong to the same class. On the other hand, it is allowable to classifier predict wrong class but is adjacent to the the ground-truth label. The reason is those classes share similar representations in hidden layers.

To refine the distribution of latent space, the label smoothing technique is applied, which adds small perturbations in the ground-truth labels. Without label smoothing, $\hat{y}$ is almost a one-hot vector. It is because feeding a one-hot vector into re-constructor is equivalent to solving a one-to-many mapping problem, which means one-hot vector represents the images of the same class with arbitrary shapes or textures. This is not what we want.

The network R is used to improve inter-class distance. It is not used in the inference time. The hyper-parameter λ balances the functions of those two terms. We will justify its optimal value in the experiments.

3.4 Models for MNIST and CIFAR10

The specific model designs for MNIST and CIFAR10 are given below. For MNIST, all classifiers use CNN architecture. Our classifier requires smaller filter size and fewer unit of fully connected layer than others, which makes our model smaller and faster. More specifically, the number of parameters in our model is only 0.2 million, while it is 0.76 millions in TRADES and 3.3 millions in Madry's model. The architecture of re-constructor is a simple model with two dense layers and three transpose convolutional layers, whose number of parameters is 0.26 millions.

For CIFAR10, the architecture of the classifier is a variation of the wide residual networks WRN22-5. Comparing to AdvFree, TRADES and Mardy's model, they use WRN34-10 [31] as the classifier model. Our WRN-22-5 model has 6.7 millions parameters, and WRB34-10 model has near 46 millions. Our re-constructor uses the upsamping version of WRN22-3 model, whose number of parameters is 5.1 millions.

4 Experiments

This section presents three sets of experiments. The first set of experiments compare the accuracy under attacks and the training performance of our model with others. The second set of experiments perform ablation study on various factors, including hyperparameter λ in Equation (16), label smoothing, and the effectiveness of adversarial training and the re-constructor model. The third set of experiments justify that our model does not belong to the gradient masking methods, and therefore can resist most kinds of attacks.

4.1 Performance and Robustness

We evaluated our method on MNIST and CIFAR10 datasets. For both datasets, we followed the instructions in [32] and reported robust accuracy with adaptive PGD (PGD-ADP). For ablation tests, we used the standard PGD with cross-entropy loss (PGD_{CE}) and PGD with CW loss (PGD_{CW}) against the defense models [24]. For MNIST, we set $\epsilon = 0.3/1.0$ on L_∞ norm. Standard attacks iterates 100 steps and the step size is 0.01. For CIFAR10, we set $\epsilon = 8.0/255$ on L_∞ norm. Attacks iterates 20 steps and the step size is 2.0/255. Adaptive attacks iterate 100 steps and search the optimal step size at each step.

We also compared our model with other competitive models. For MNIST, Madry's model uses adversarial training with PGD_{CE}-40 on training set; TRADES sets β to 1; and AdvFree sets m to 15 and step size to 0.01. For CIFAR10, Madry's model uses adversarial training with PGD_{CE}-10 on the training set; TRADES sets β to 6; and AdvFree sets m to 8.

Table 1 summarizes the experimental results. The first column is the total time for training; the second column is the elapsed time of each epoch; the

Table 1. Comparison of different models.

MNIST	T_{train}[s]	T_{epoch}[s]	$\text{acc}_{\text{nat.}}$	$\text{acc}_{\text{adv.}}$
Ours	268	6.7	99.01	92.37
Madry's	69,993	77.0	97.26	88.45
TRADES	8,000	80.0	99.48	95.41
AdvFree	1,430	2.6	99.40	87.62

CIFAR10	T_{train}[s]	T_{epoch}[s]	$\text{acc}_{\text{nat.}}$	$\text{acc}_{\text{adv.}}$
Ours	7,095	141.9	88.01	46.06
Madry's	240,177	1351.0	87.25	44.04
TRADES	118,560	1526.0	84.92	52.76
AdvFree	29,120	183.0	86.07	43.80

third column is accuracy of natural data, and the forth column is accuracy under adversarial attacks. As can be seen, our model is the fastest model for both datasets, although our training has a little extra computational cost for re-constructor. The reason is our model is much smaller than others. Also, we have the fastest rate of convergence. For CIFAR10, our model achieves 46.06% robust accuracy, which is similar to Madry's method. For MNIST, TRADES's natural and robust accuracy is slightly higher than our model. MNIST is a good example for large ϵ. If AdvFree's step size is increased, it would fail to converge.

4.2 Ablation Study

Hyperparameter $\boldsymbol{\lambda}$ This experiment verifies how the hyperparameter λ in Equation (16) affects the robust accuracy of our model on MNIST and CIFAR10. Table 2 lists the model accuracy for MNIST and CIFAR10 under different attacks respectively. The result shows the best choice of λ for MNIST is near 0.1, and the best λ for CIFAR10 is 0.05. If a smaller λ is chosen, the categorical loss in Equation (16) will be neglected, which means the model is like to solve an unsupervised clustering problem.

Label Smoothing This experiment investigates the influence of the label smoothing for the model accuracy. Table 3 shows the results of the model trained without using the label smoothing technique. Comparing to the results shown in Table 2, one can find that the model's robust accuracy increases significantly with label smoothing technique.

Adversarial Training and Re-constructor This experiment compares the effectiveness of two techniques used in our method: adversarial training with enhanced FGSM attack and re-constructor model. For each dataset, we tried three different combinations:

- AdvTrain: using adversarial training with enhanced FGSM attack only.
- REC: using re-constructor model only. The input images are unmodified.
- Combined: using both techniques.

The results, which are the model accuracy under different kinds of attacks, are shown in Table 4 for MNIST and CIFAR10 respectively. AdvTrain successfully generates good quality of adversarial images. The robust accuracy for PGD_{CE}

Table 2. Influence of λ on the model accuracy (%). Left: MNIST. Right CIFAR10.

λ	1.0	0.5	0.1	0.01	0.001
Natural	99.22	99.13	99.02	96.72	53.19
PGD-ADP	91.24	91.54	92.37	86.66	26.08
PGD_{CE}-100	96.75	96.49	96.55	92.10	36.08
PGD_{CW}-100	96.58	96.47	96.33	91.27	30.50

λ	1.0	0.5	0.1	0.05	0.01
Natural	88.00	87.97	87.76	88.01	83.19
PGD-ADP	36.89	39.82	44.03	46.06	38.51
PGD_{CE}-20	59.47	59.73	64.65	65.36	54.02
PGD_{CW}-20	54.84	59.20	64.48	65.21	55.17

Table 3. Accuracy (%) of the model trained without using label smoothing technique. Left: MNIST. Right CIFAR10.

λ	1.0	0.5	0.1	0.01	0.001
Natural	99.31	99.31	99.17	98.46	70.91
PGD-ADP	87.70	88.33	90.63	89.31	54.58
PGD_{CE}-20	94.60	94.58	95.68	95.15	62.72
PGD_{CW}-20	94.79	94.68	95.72	94.81	59.22

λ	1.0	0.5	0.1	0.05	0.01
Natural	86.82	86.85	86.18	86.67	84.88
PGD-ADP	22.63	22.90	26.91	31.43	28.23
PGD_{CE}-20	24.85	25.69	35.22	41.39	56.32
PGD_{CW}-20	25.49	27.00	39.22	43.61	57.04

Table 4. Accuracy (%) of models trained by different techniques. Left: MNIST. Right: CIFAR10

Techniques	AdvTrain	REC	Combined
Natural	99.02	98.98	99.01
PGD-ADP	90.05	0.00	92.37
PGD_{CE}-100	96.06	0.18	96.27
PGD_{CW}-100	95.24	0.00	96.23

Techniques	AdvTrain	REC	Combined
Natural	87.05	90.05	88.01
PGD-ADP	40.21	0.07	46.06
PGD_{CE}-20	62.22	41.20	65.36
PGD_{CW}-20	56.32	43.59	65.21

is similar to Madrey's model for both dataset, but here is a gap between the robust accuracy for PGD_{CE} and robust accuracy for PGD_{CW}. It implies that AdvTrain may not to defense against arbitrary attacks.

The goal of re-constructor is to make the classifier converge faster and more robust. Without any adversarial training data, the re-constructor is useless, because it does not help the original classifier to capture more features. Hence, re-constructor model only method fails completely for MNIST. On the other hand, the images in CIFAR10 have a large variation even for the objects in the same class. Moreover, the image background are not unified. Thus, the re-constructor alone can enhance the ability of classifier to learn better features. The images augmented by sampled noise just enlarge the variation of training data, whose function is not significant.

Early Stopping Estimation This experiment evaluates the robust accuracy by Equation (9). For MNIST, we get 14.7% high risk data; for CIFAR10, we get 43.12% high risk data. That means the distance between those data and the decision boundary is less than ϵ. Comparing with Table 4, our method gives a better bound than that of the standard PGD attacks.

TRADES With Enhanced FGSM The enhanced FGSM can be applied not only to our model, but also others. We combined eFGSM with TRADES, and evaluated its performance. The robust accuracy of such combination for CIFAR10 is 48.05%, which is less than that of TRADES (54.02%), but is slightly better than that of our original method (46.07%). However, its average epoch time is about 250s, which is much faster than that of TRADES (1526s). Therefore, for fast training, eFGSM has its value to achieve good descent accuracy with significant acceleration.

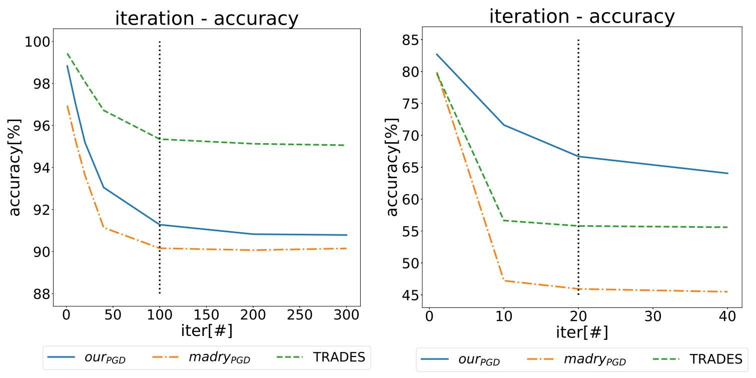

Fig. 2. Robust accuracy decreases when PGD iteration increases. Left: $\epsilon = 0.3/1.0$ on MNIST. Right: $\epsilon = 8/255$ on CIFAR10.

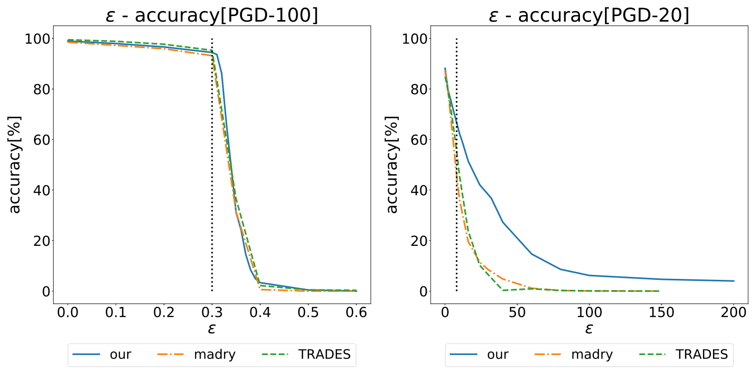

Fig. 3. Robust accuracy decreases when ϵ increases. Left: MNIST. Right CIFAR10.

4.3 Justification of Gradient Masking Issues

This set of experiments checked whether our model shows some properties of gradient masking with three experiments.

– Increase the number of iterations for PGD attack (checking property 2).
– Increase the size of ϵ (checking property 3 and 4).
– Attack the models using other methods (checking property 1 and 5).

The result of the first experiment is shown in Figure 2. As can be seen, the accuracy converges when the number of iterations of PGD increases, which means the attacks are stronger.

Figure 3 shows the robust accuracy for MNIST and CIFAR10 when the size of ϵ increases. First, the accuracy are decreased monotonically as the size of ϵ grows. Second, when ϵ is large enough, the accuracy drops to 0. For CIFAR10, the degradation of our method is slower than that of others and the curve does not reach to 0% when ϵ is larger than 200.

Table 5. Robust accuracy for black-box PGD_{CE}-20 attack on our model

	Conv-4	ResNet-56	Madry's
accuracy[%]	85.03	83.42	72.0

We want to emphasize that our method seems to perform better result than others but this does not mean that our method is quite robust than others. We selected standard PGD as attacker since we get quite suspicious result in ablation study. This phenomenon is reasonable because the purpose of those sets of experiments is verifying gradient masking issue. Once we strengthen the power of PGD with several times with random restarts, robust accuracy will be zero eventually. From the experimental results, we believe that obfuscating gradient does not occur in our training method.

The last experiment uses the transferability of attacks on CIFAR10 to show that other kinds of attacks are weaker than the PGD based method for our model. We used the attacks generated for other three models: (1) a four layer convolution neural network (Conv-4), (2) Resnet-56 [33], and (3) Madry's adversarial trained model. The accuracy for our model under the attacks transferred from other models are shown in Table 5. As can be seem the values are between the accuracy under PGD attacks (46.06%) and the natural accuracy (88.01%). Attacks generated from Madry's model are stronger than others because our model is similar to Madry's. These experiments confirmed that the gradient masking does not occur in our model.

5 Conclusion

In this paper, we presented two methods to improve the robustness and the performance of adversarial training. The first one is a training architecture, which leverages a re-constructor to make the model learn the classifier on a lower dimension manifold, on which the inter-class distance is shrunken, so the trained model can be more robust and the model size can be reduced. The second method is a fast way to generate the adversarial examples. We enhanced FGSM so that it can generate more reprensentable attacks without solving the optimization problems. Experimental results show that the models generated by our methods for MNIST and CIFAR10 require less parameters and run faster than other training methods.

In the future, we have several research directions to explore. First, we want to try our methods on more complicated models and datasets, such as ImageNet. Second, for the training architecture, how to build a suitable re-constructor to help the training of more robust classifiers requires further investigation. Third, for adversarial example generator, more computationally economic ways still need to research. Last, we would like to extend our methods to other applications, besides image classification, to resist different kinds of adversarial attacks.

References

1. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. (2012) 1097–1105
2. Redmon, J., Farhadi, A.: Yolo9000: better, faster, stronger. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2017) 7263–7271
3. Hossain, M., Sohel, F., Shiratuddin, M.F., Laga, H.: A comprehensive survey of deep learning for image captioning. ACM Computing Surveys (CSUR) **51** (2019) 118
4. Huval, B., Wang, T., Tandon, S., Kiske, J., Song, W., Pazhayampallil, J., Andriluka, M., Rajpurkar, P., Migimatsu, T., Cheng-Yue, R., et al.: An empirical evaluation of deep learning on highway driving. arXiv preprint arXiv:1504.01716 (2015)
5. Litjens, G., Kooi, T., Bejnordi, B.E., Setio, A.A.A., Ciompi, F., Ghafoorian, M., Van Der Laak, J.A., Van Ginneken, B., Sánchez, C.I.: A survey on deep learning in medical image analysis. Medical image analysis **42** (2017) 60–88
6. Kurakin, A., Goodfellow, I., Bengio, S.: Adversarial examples in the physical world. arXiv preprint arXiv:1607.02533 (2016)
7. Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T., Song, D.: Robust physical-world attacks on deep learning models. arXiv preprint arXiv:1707.08945 (2017)
8. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 (2013)
9. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014)
10. Carlini, N., Wagner, D.: Towards evaluating the robustness of neural networks. In: 2017 IEEE Symposium on Security and Privacy (SP), IEEE (2017) 39–57
11. Su, J., Vargas, D.V., Sakurai, K.: One pixel attack for fooling deep neural networks. IEEE Transactions on Evolutionary Computation (2019)
12. Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z.B., Swami, A.: Practical black-box attacks against machine learning. In: Proceedings of the 2017 ACM on Asia conference on computer and communications security, ACM (2017) 506–519
13. Athalye, A., Carlini, N., Wagner, D.: Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. arXiv preprint arXiv:1802.00420 (2018)
14. Ross, A.S., Doshi-Velez, F.: Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In: Thirty-second AAAI conference on artificial intelligence. (2018)
15. Zhang, H., Yu, Y., Jiao, J., Xing, E.P., Ghaoui, L.E., Jordan, M.I.: Theoretically principled trade-off between robustness and accuracy. arXiv preprint arXiv:1901.08573 (2019)
16. Xie, C., Wu, Y., Maaten, L.v.d., Yuille, A.L., He, K.: Feature denoising for improving adversarial robustness. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2019) 501–509
17. Wong, E., Rice, L., Kolter, J.Z.: Fast is better than free: Revisiting adversarial training. arXiv preprint arXiv:2001.03994 (2020)

18. Shafahi, A., Najibi, M., Ghiasi, M.A., Xu, Z., Dickerson, J., Studer, C., Davis, L.S., Taylor, G., Goldstein, T.: Adversarial training for free! In: Advances in Neural Information Processing Systems. (2019) 3353–3364
19. Zhang, D., Zhang, T., Lu, Y., Zhu, Z., Dong, B.: You only propagate once: Accelerating adversarial training via maximal principle. In: Advances in Neural Information Processing Systems. (2019) 227–238
20. Raff, E., Sylvester, J., Forsyth, S., McLean, M.: Barrage of random transforms for adversarially robust defense. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2019) 6528–6537
21. Kannan, H., Kurakin, A., Goodfellow, I.: Adversarial logit pairing. arXiv preprint arXiv:1803.06373 (2018)
22. Bernstein, J., Zhao, J., Azizzadenesheli, K., Anandkumar, A.: signsgd with majority vote is communication efficient and fault tolerant. arXiv preprint arXiv:1810.05291 (2018)
23. Liu, S., Chen, P.Y., Chen, X., Hong, M.: signsgd via zeroth-order oracle. In: International Conference on Learning Representations. (2018)
24. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083 (2017)
25. Moosavi-Dezfooli, S.M., Fawzi, A., Frossard, P.: Deepfool: a simple and accurate method to fool deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2016) 2574–2582
26. Virmaux, A., Scaman, K.: Lipschitz regularity of deep neural networks: analysis and efficient estimation. In: Advances in Neural Information Processing Systems. (2018) 3835–3844
27. Opitz, D., Maclin, R.: Popular ensemble methods: An empirical study. Journal of artificial intelligence research **11** (1999) 169–198
28. Müller, R., Kornblith, S., Hinton, G.E.: When does label smoothing help? In: Advances in Neural Information Processing Systems. (2019) 4694–4703
29. Sharma, Y., Chen, P.Y.: Attacking the madry defense model with l_1-based adversarial examples. arXiv preprint arXiv:1710.10733 (2017)
30. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A.: Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. J. Mach. Learn. Res. **11** (2010) 3371–3408
31. Zagoruyko, S., Komodakis, N.: Wide residual networks. arXiv preprint arXiv:1605.07146 (2016)
32. Carlini, N., Athalye, A., Papernot, N., Brendel, W., Rauber, J., Tsipras, D., Goodfellow, I., Madry, A., Kurakin, A.: On evaluating adversarial robustness. arXiv preprint arXiv:1902.06705 (2019)
33. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2016) 770–778