Attacks Which Do Not Kill Training Make Adversarial Learning Stronger

Jingfeng Zhang [*] [†] [1] **Xilie Xu** [*] [2] **Bo Han** [3] [4] **Gang Niu** [4] **Lizhen Cui** [5]
Masashi Sugiyama [4] [6] **Mohan Kankanhalli** [1]

Abstract

Adversarial training based on the *minimax* formulation is necessary for obtaining *adversarial robustness* of trained models. However, it is conservative or even pessimistic so that it sometimes hurts the *natural generalization*. In this paper, we raise a fundamental question—do we have to trade off natural generalization for adversarial robustness? We argue that adversarial training is to employ confident adversarial data for updating the current model. We propose a novel formulation of *friendly adversarial training* (FAT): rather than employing most adversarial data maximizing the loss, we search for least adversarial data (i.e., *friendly adversarial data*) minimizing the loss, among the adversarial data that are confidently misclassified. Our novel formulation is easy to implement by just stopping the most adversarial data searching algorithms such as PGD (projected gradient descent) early, which we call *early-stopped PGD*. Theoretically, FAT is justified by an upper bound of the adversarial risk. Empirically, early-stopped PGD allows us to answer the earlier question negatively—adversarial robustness can indeed be achieved without compromising the natural generalization.

1. Introduction

Safety-critical nature of some areas such as medicine (Buch et al., 2018) and automatic driving (Litman, 2017), necessitates the need for deep neural networks (DNNs) to be adversarially robust that generalize well. Recent research focuses on improving their robustness mainly by two defense approaches, i.e., certified defense and empirical defense. Certified defense tries to learn provably robust DNNs against norm-bounded (e.g., ℓ_2 and ℓ_∞) perturbations (Cohen et al., 2019; Wong & Kolter, 2018; Tsuzuku et al., 2018; Lécuyer et al., 2019; Weng et al., 2018; Balunovic & Vechev, 2020; Zhang et al., 2020). Empirical defense incorporates adversarial data into the training process (Goodfellow et al., 2015; Madry et al., 2018; Cai et al., 2018; Zhang et al., 2019b; Wang et al., 2019; 2020). For instance, empirical defense has been used to train Wide ResNet (Zagoruyko & Komodakis, 2016) with natural data and its adversarial variants to make the trained network robust against strong adaptive attacks (Athalye et al., 2018; Carlini & Wagner, 2017). This paper belongs to the empirical defense category.

Existing empirical defense methods formulate the adversarial training as a minimax optimization problem (Section 2.1) (Madry et al., 2018). To conduct this minimax optimization, projected gradient descent (PGD) is a common method to generate the most adversarial data that maximizes the loss, updating the current model. PGD perturbs the natural data for a fixed number of steps with small step size. After each step of perturbation, PGD projects the adversarial data back onto the ϵ-norm ball of the natural data.

However, this minimax formulation is conservative (or even pessimistic), such that it sometimes hurts the natural generalization (Tsipras et al., 2019). For example, the top panels in Figure 1 show that at step #6 to #10 in PGD, the adversarial variants of the natural data significantly cross over the decision boundary and are located at their peer's (natural data) area. Since adversarial training aims to fit natural data and its adversarial variants simultaneously, such the cross-over mixture makes adversarial training extremely difficult. Therefore, the most adversarial data generated by the PGD-10 (i.e., step #10 in top panel of Figure 1) directly "kill" the training, thus rendering the training unsuccessful.

Inspired by philosopher Friedrich Nietzsche's quote "*that which does not kill us makes us stronger*," we propose *friendly adversarial training* (FAT): rather than employing the most adversarial data for updating the current model, we

[*]Equal contribution [†]Preliminary work was done during an internship at RIKEN AIP. [1]School of Computing, National University of Singapore, Singapore [2]Taishan College, Shandong University, Jinan, China [3]Department of Computer Science, Hong Kong Baptist University, Hong Kong, China [4]RIKEN Center for Advanced Intelligence Project (AIP), Tokyo, Japan [5]School of Software & Joint SDU-NTU Centre for Artificial Intelligence Research (C-FAIR), Shandong University, Jinan, China [6]Graduate School of Frontier Sciences, The University of Tokyo, Tokyo, Japan. Correspondence to: Jingfeng Zhang <j-zhang@comp.nus.edu.sg>.

Proceedings of the 37th International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

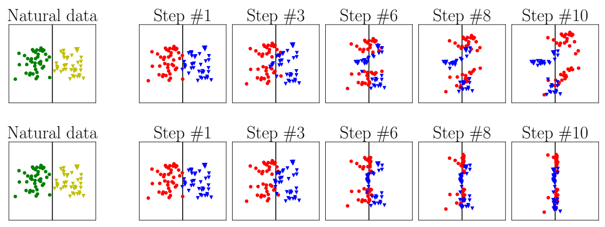


Figure 1. Green circles and yellow triangles are natural data for binary classification. Red circles and blue triangles are adversarial variants of green circles and yellow triangles, respectively. Black solid line is the decision boundary representing the current classifier. Top: the adversarial data generated by PGD. Bottom: the adversarial data generated by early-stopped PGD.

search for the *friendly adversarial data* minimizing the loss. The friendly adversarial data are confidently misclassified by the current model. We design the learning objective of FAT and theoretically justify it by deriving an upper bound of the adversarial risk (Section 3). Essentially, FAT updates the current model using friendly adversarial data. FAT trains a DNN using the wrongly-predicted adversarial data minimizing the loss and the correctly-predicted adversarial data maximizing the loss.

FAT is a reasonable strategy due to two reasons: It removes the existing inconsistency between attack and defense, and it adheres to the spirit of curriculum learning. First, the ways of generating adversarial data by adversarial attackers and adversarial defense methods are inconsistent. Adversarial attacks (Szegedy et al., 2014; Carlini & Wagner, 2017; Athalye et al., 2018) aim to find the adversarial data (not maximizing the loss) to confidently fool the model. On the other hand, existing adversarial defense methods generate the most adversarial data maximizing the loss regardless of the model's predictions. These two should be harmonized. Second, the curriculum learning strategy has been shown to be effective (Bengio et al., 2009). Fitting most adversarial data initially makes the learning extremely difficult, sometimes even killing the training. Instead, FAT learns initially from the least adversarial data and progressively utilizes increasingly adversarial data.

FAT is easy to implement by just early stopping most-adversarial-data searching algorithms such as PGD, which we call early-stopped PGD (Section 4.1). Once adversarial data is misclassified by the current model, we stop the PGD iterations early. Early-stopped PGD has the benefit of alleviating the cross-over mixture problem. For example, as shown in the bottom panels of Figure 1, adversarial data generated by early-stopped PGD will not be located at their peer areas (extensive details in Section 4.2). Thus, it will not hurt the generalization ability much. In addition, FAT based on early-stopped PGD progressively employs stronger and stronger adversarial data (with more PGD steps), engendering increasingly enhanced robustness of the model over the training progression (Section 5.3). This implies that attacks that do not kill the training indeed make the adversarial learning stronger.

A brief overview of our contributions is as follows. We propose a novel formulation for adversarial learning (Section 3.1) and theoretically justify it by an upper bound of the adversarial risk (Section 3.2). Our FAT approximately realizes this formulation by just stopping PGD early. FAT has the following benefits.

- Conventional adversarial training methods, e.g., standard adversarial training (Madry et al., 2018), TRADES (Zhang et al., 2019b) and MART (Wang et al., 2020), can be easily modified to become friendly adversarial training counterparts, i.e., FAT, FAT for TRADES, and FAT for MART (Section 5).
- Compared with conventional adversarial training, FAT has a better standard accuracy for natural data, while keeping a competitively robust accuracy for adversarial data (Sections 5.1 and 6.2).
- FAT is computationally efficient because the early stopped PGD saves a large number of backward propagations for searching adversarial data (Section 5.2).
- FAT can enable larger values of the perturbation bound, i.e., ϵ_{train} (Section 6.1), due to that FAT can alleviate the cross-over mixture problem (Section 4.2).

With these benefits, FAT allows us to answer that adversarial robustness can indeed be achieved without compromising the natural generalization.

2. Standard Adversarial Training

Let $(\mathcal{X}, d_\infty)$ be the input feature space $\mathcal{X}$ with the infinity distance metric $d_{\inf}(x, x') = \|x - x'\|_\infty$, and

$$\mathcal{B}_\epsilon[x] = \{x' \in \mathcal{X} \mid d_{\inf}(x, x') \leq \epsilon\} \tag{1}$$

be the closed ball of radius $\epsilon > 0$ centered at x in $\mathcal{X}$.

2.1. Learning Objective

Given a dataset $S = \{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y} = \{0, 1, ..., C-1\}$, the objective function of standard adversarial training (Madry et al., 2018) is

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \left\{ \max_{\tilde{x} \in \mathcal{B}_\epsilon[x_i]} \ell(f(\tilde{x}), y_i) \right\}, \tag{2}$$

where $\tilde{x}$ is the adversarial data within the ϵ-ball centered at x, $f(\cdot) : \mathcal{X} \to \mathbb{R}^C$ is a score function, and the loss function $\ell : \mathbb{R}^C \times \mathcal{Y} \to \mathbb{R}$ is a composition of a base loss

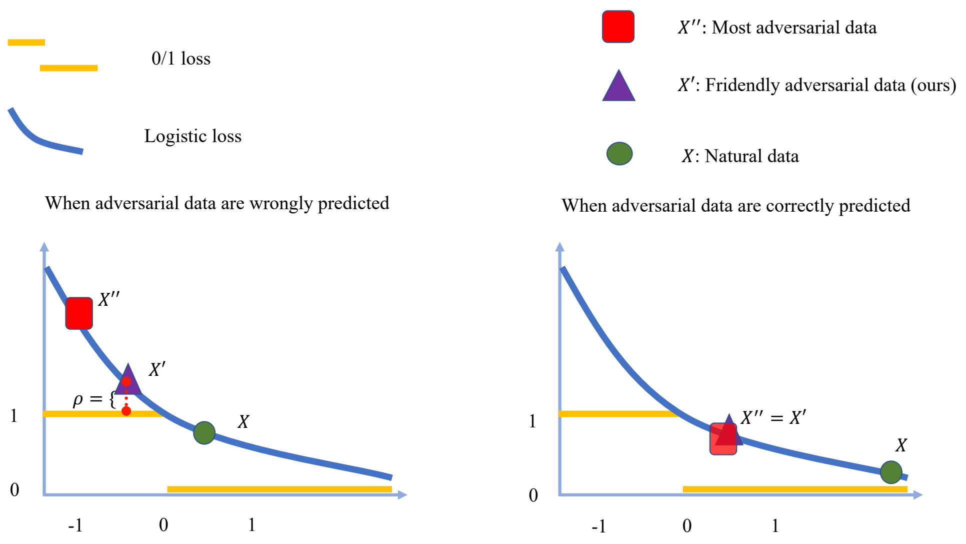

0/1 loss
Logistic loss
X'': Most adversarial data
X': Fridendly adversarial data (ours)
X: Natural data
When adversarial data are wrongly predicted
When adversarial data are correctly predicted
X''
X'
$\rho = \{$
X
$X'' = X'$
1
0
-1
0
1

Figure 2. In adversarial training, the adversarial data is generated within the perturbation ball $\mathcal{B}_\epsilon[X]$ of natural data X given the current model f. The existing adversarial training generates most adversarial data X'' that maximizes the inner loss regardless of their predictions. FAT takes their predictions into account. When the model makes the wrong predictions of adversarial data, our friendly adversarial data X' minimizes the inner loss by a violation of a small constant ρ.

$\ell_{\mathrm{B}} : \Delta^{C-1} \times \mathcal{Y} \to \mathbb{R}$ (e.g., the cross-entropy loss) and an inverse link function $\ell_{\mathrm{L}} : \mathbb{R}^C \to \Delta^{C-1}$ (e.g., the soft-max activation), in which Δ^{C-1} is the corresponding probability simplex—in other words, $\ell(f(\cdot), y) = \ell_{\mathrm{B}}(\ell_{\mathrm{L}}(f(\cdot)), y)$.

For the sake of conceptual consistency, the objective function (i.e., Eq. (2)) can also be re-written as

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(f(\tilde{x}_i), y_i), \tag{3}$$

where

$$\tilde{x}_i = \arg\max\nolimits_{\tilde{x} \in \mathcal{B}_\epsilon[x_i]} \ell(f(\tilde{x}), y_i). \tag{4}$$

It implies the optimization of adversariallly robust network, with one step maximizing loss to find adversarial data and one step minimizing loss on the adversarial data w.r.t. the network parameters θ.

2.2. Projected Gradient Descent (PGD)

To generate adversarial data, standard adversarial training uses PGD to approximately solve the inner maximization of Eq. (4) (Madry et al., 2018).

PGD formulates the problem of finding adversarial data as a constrained optimization problem. Namely, given a starting point $x^{(0)} \in \mathcal{X}$ and step size $\alpha > 0$, PGD works as follows:

$$x^{(t+1)} = \Pi_{\mathcal{B}[x^{(0)}]}\big(x^{(t)} + \alpha \,\mathrm{sign}(\nabla_{x^{(t)}} \ell(f_\theta(x^{(t)}), y))\big), \forall t \geq 0 \tag{5}$$

until a certain stopping criterion is satisfied. For example, the criterion can be a fixed number of iterations K, namely the PGD-K algorithm (Madry et al., 2018; Wang et al., 2020). In Eq. (5), ℓ is the loss function in Eq. (4); $x^{(0)}$ refers to natural data or natural data corrupted by a small Gaussian or uniform random noise; y is the corresponding label for natural data; $x^{(t)}$ is adversarial data at step t; and $\Pi_{\mathcal{B}_\epsilon[x_0]}(\cdot)$ is the projection function that projects the adversarial data back into the ϵ-ball centered at $x^{(0)}$ if necessary.

There are also other ways to generate adversarial data, e.g., the fast gradient signed method (Szegedy et al., 2014; Goodfellow et al., 2015), the CW attack (Carlini & Wagner, 2017), deformation attack (Alaifari et al., 2019; Xiao et al., 2018), and Hamming distance method (Shamir et al., 2019).

PGD adversarial training. Besides the standard adversarial training, several improvements to PGD adversarial training have also been proposed, such as Lipschitz regularization (Cisse et al., 2017; Hein & Andriushchenko, 2017; Yan et al., 2018; Farnia et al., 2019), curriculum adversarial training (Cai et al., 2018; Wang et al., 2019), computationally efficient adversarial learning (Shafahi et al., 2019; Zhang et al., 2019a; Wong et al., 2020), ensemble adversarial training (Tramr et al., 2018; Pang et al., 2019), and adversarial training by utilizing unlabeled data (Carmon et al., 2019; Najafi et al., 2019; Alayrac et al., 2019). In addition, TRADES (Zhang et al., 2019b) and MART (Wang et al., 2020) are effective adversarial training methods, which trains on both natural data x and adversarial data $\tilde{x}$ (the learning objectives are reviewed in Appendices D.1 and E.1 respectively). Moreover, there are interesting analyses of PGD adversarial training, such as showing overfitting in PGD-adversarial training (Rice et al., 2019), disentangling robust and non-robust features through PGD-adversarially trained network (Ilyas et al., 2019), showing different feature representations by robust model and non-robust model (Tsipras et al., 2019), and providing a new explanation for the trade-off between robustness and accuracy of PGD-adversarial training (Raghunathan et al., 2020).

3. Friendly Adversarial Training

In this section, we develop a novel learning objective for friendly adversarial training (FAT). Theoretically, we justify FAT by deriving a tight upper bound of the adversarial risk.

3.1. Learning Objective

Let $\rho > 0$ be a margin such that our adversarial data would be misclassified with a certain amount of confidence.

The outer minimization still follows Eq. (3). However, instead of generating $\tilde{x}_i$ via inner maximization, we generate $\tilde{x}_i$ as follows:

$$\tilde{x}_i = \arg\min_{\tilde{x}\in\mathcal{B}_\epsilon[x_i]} \ell(f(\tilde{x}), y_i)$$
$$\text{s.t. } \ell(f(\tilde{x}), y_i) - \min\nolimits_{y\in\mathcal{Y}} \ell(f(\tilde{x}), y) \geq \rho.$$

Note that the operator $\arg\max$ in Eq. (4) is replaced with $\arg\min$ here, and there is a constraint on the margin of loss values (i.e., the misclassification confidence).

The constraint firstly ensures $y_i \neq \arg\min_{y\in\mathcal{Y}} \ell(f(\tilde{x}), y)$ or $\tilde{x}$ is misclassified, and secondly ensures for $\tilde{x}$ the wrong prediction is better than the desired prediction y_i by at least ρ in terms of the loss value. Among all such $\tilde{x}$ satisfying the constraint, we select the one minimizing $\ell(f(\tilde{x}), y_i)$. Namely, we minimize the adversarial loss given that some confident adversarial data has been found. This $\tilde{x}_i$ could be regarded as a "friend" among the adversaries, which is termed *friendly adversarial data*. Figure 2 illustrates the differences between our learning objective and the conventional minimax formulation.

3.2. Upper Bound on Adversarial Risk

In this subsection, we derive a tight upper bound on the adversarial risk, and provide our analysis for adversarial risk minimization. Let X and Y represent random variables. We employ the definition of the adversarial risk given by Zhang et al. (2019b), i.e., $\mathcal{R}_{\text{rob}}(f) := \mathbb{E}_{(X,Y)\sim\mathcal{D}}\mathbb{1}\{\exists X' \in \mathcal{B}_\epsilon[X] : f(X') \neq Y\}$.

Theorem 1. For any classifier f, any non-negative surrogate loss function ℓ which upper bounds the $0/1$ loss, and any probability distribution $\mathcal{D}$, we have

$$\mathcal{R}_{\text{rob}}(f) \leq \underbrace{\mathbb{E}_{(X,Y)\sim\mathcal{D}}\ell(f(X), Y)}_{\text{For standard test accuracy}} + \underbrace{\mathbb{E}_{(X,Y)\sim\mathcal{D}, X'\in\mathcal{B}_\epsilon[X,\epsilon]}\ell^*(f(X'), Y)}_{\text{For robust test accuracy}},$$

where

$$\ell^* = \begin{cases} \min \ell(f(X'), Y) + \rho, & \text{if } f(X') \neq Y, \\ \max \ell(f(X'), Y), & \text{if } f(X') = Y. \end{cases}$$

Note that ρ is the small margin such that friendly adversarial data would be misclassified with a certain amount of confidence. The proof is in Appendix A.2. From Theorem 1, our upper bound on the adversarial risk is tighter than that of conventional adversarial training, e.g.,TRADES (Zhang et al., 2019b), where they maximize the loss regardless of model prediction, i.e., $\ell^* = \max \ell(f(X'), Y)$. By contrast, our bound takes the model prediction into consideration. When the model makes correct prediction on adversarial data X' (i.e., $f(X') = Y$), we still maximize the loss; while the model makes wrong prediction on adversarial data X' (i.e., $f(X') \neq Y$), we minimize the inner loss by violation of a small constant ρ. To better understand the nature of adversarial training and Theorem 1, we provide supporting schematics in Figure 2 and Figure 8 (in Appendix A.2).

Minimizing the adversarial risk based on our upper bound aids in fine-tuning the decision boundary using friendly adversarial data. On one side, the wrongly-predicted adversarial data have a small distance ρ (in term of the loss value) from the decision boundary (e.g., "Step #10 panel" at the bottom series in Figure 1) so that it will not cause the severe issue of cross-over mixture but fine-tunes the decision boundary. On the other side, correctly-predicted adversarial data maintain the largest distance (in term of maximizing the loss value) from their natural data so that the decision boundary is kept far away.

4. Key Component of FAT

To search friendly adversarial data, we design an efficient early-stopped PGD algorithm called PGD-K-τ (Section 4.1), which could alleviate the cross-over mixture problem (Section 4.2) and therefore helps adversarial training. Note that besides PGD-K-τ, there are other ways to search for friendly adversarial data. We show one example in Appendix B.

4.1. PGD-K-τ Algorithm

In Algorithm 1, $\Pi_{\mathcal{B}[x,\epsilon]}$ is the projection operator that projects adversarial data $\tilde{x}$ onto the ϵ-norm ball centered at

Algorithm 1 PGD-K-τ

```
Input: data $x \in \mathcal{X}$, label $y \in \mathcal{Y}$, model $f$, loss function $\ell$, maximum PGD step $K$, step $\tau$, perturbation bound $\epsilon$, step size $\alpha$
Output: $\tilde{x}$
$\tilde{x} \leftarrow x$
while $K > 0$ do
  if $\arg\max_i f(\tilde{x}) \neq y$ and $\tau = 0$ then
    break
  else if $\arg\max_i f(\tilde{x}) \neq y$ then
    $\tau \leftarrow \tau - 1$
  end if
  $\tilde{x} \leftarrow \Pi_{\mathcal{B}[x,\epsilon]}\big(\alpha\,\mathrm{sign}(\nabla_{\tilde{x}}\ell(f(\tilde{x}), y)) + \tilde{x}\big)$
  $K \leftarrow K - 1$
end while
```

x, and $\arg\max_i f(\tilde{x})$ returns the predicted label of adversarial data $\tilde{x}$, where $f(\tilde{x}) = \big(f^i(\tilde{x})\big)^\top_{i=0,\ldots,C-1}$ measures the probabilistic predictions over C classes. Unlike the conventional PGD-K generating adversarial data by maximizing the loss function ℓ regardless of model prediction, our PGD-K-τ generates the adversarial data which takes model prediction into consideration.

Algorithm 1 returns the misclassified adversarial data with small loss values or correctly classified adversarial data with large loss values. Step τ controls the extent of loss minimization when misclassified adversarial data are found. When τ is larger, the misclassified adversarial data with slightly larger loss values are returned, and vice versa. $\tau \times \alpha$ is an approximation to ρ in our learning objective. Note that when $\tau = K$, the conventional PGD-K is the special case of our PGD-K-τ. As τ is an important hyper-parameter of PGD-K-τ for FAT (Section 5), we discuss how to select τ in Sections 5.1 and 5.2 in detail.

4.2. PGD-K-τ Alleviates Cross-over Mixture

In deep neural networks, the cross-over mixture problem may not trivially appear in the original input space, but occur in the output of the intermediate layer. Our proposed PGD-K-τ is an effective solution to overcome this problem, which leads to successful adversarial training.

In Figure 3, we trained an 8-layer convolutional neural network (6 convolutional layers and 2 fully-connected layers, namely, Small CNN) on images of two selected classes in CIFAR-10. We conducted a warm-up training using natural training data, then included their adversarial variants generated by PGD-20 (middle panel) and PGD-20-0 (right panel), where $\tau = 0$ means PGD iterations stop immediately once adversarial data are wrongly predicted by the current network.

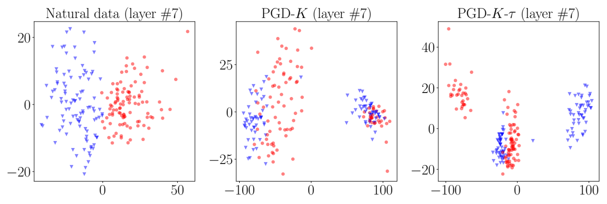


Figure 3. Left: layer #7's output distribution on natural data (not mixed). Middle: layer #7's output distribution on adversarial data generated by PGD-20 (significantly mixed). Right: layer #7's output distribution on friendly adversarial data generated by PGD-20-0 (not significantly mixed).

Figure 3 shows the output distribution of layer #7 by principal component analysis (PCA, (Abdi & Williams, 2010)), which projects high-dimensional output into a two-dimensional subspace. As shown in Figure 3, the output distribution on natural data (left panel) of the intermediate layer is clearly not mixed. By contrast, the conventional PGD-K (middle panel) leads to severe mixing between outputs of adversarial data with different classes. It is more difficult to fit these mixed adversarial data, which leads to inaccurate classifiers. By comparing with PGD-K, our PGD-K-τ (right panel) could greatly overcome the mixture issue of adversarial data. Thus, it helps the training algorithm to return an accurate classifier while ensuring adversarial robustness.

To further justify the above fact, we plot output distributions of other layers, e.g., layer #6 and layer #8. Moreover, we train a Wide ResNet (WRN-40-4) (Zagoruyko & Komodakis, 2016) on 10 classes and randomly select 3 classes for illustrating their output distributions of its intermediate layers. Instead of PCA, we also use a non-linear technique for dimensionality reduction, i.e., t-distributed stochastic neighbor embedding (t-SNE) (Maaten & Hinton, 2008) to visualize output distributions of different classes. All these can be found in Appendix C.

5. Realization of FAT

Based on the proposed PGD-K-τ, we have a new algorithm termed FAT (Algorithm 2). FAT treats the standard adversarial training (Madry et al., 2018) as a special case when we set $\tau = K$ in Algorithm 1. Besides, we also design FAT for TRADES (Appendix D.2) and FAT for MART (Appendix E.2), making two effective adversarial training methods (TRADES (Zhang et al., 2019b) and MART (Wang et al., 2020)) special cases when $\tau = K$. Since the essential component of FAT is PGD-K-τ, we should discuss the effects of step τ w.r.t. standard accuracy and adversarial robustness (Section 5.1) and computational efficiency (Section 5.2). Besides, we should discuss the relation between FAT and curriculum learning (Section 5.3), since FAT is a progressive training strategy. It is worth noting that

Algorithm 2 Friendly Adversarial Training (FAT)

```
Input: network $f_\theta$, training dataset $S = \{(x_i, y_i)\}_{i=1}^n$,
learning rate $\eta$, number of epochs $T$, batch size $m$, number of batches $M$
Output: adversarially robust network $f_\theta$
for epoch $= 1, \dots, T$ do
  for mini-batch $= 1, \dots, M$ do
    Sample a mini-batch $\{(x_i, y_i)\}_{i=1}^m$ from $S$
    for $i = 1, \dots, m$ (in parallel) do
      Obtain adversarial data $\tilde{x}_i$ of $x_i$ by Algorithm 1
    end for
    $\theta \leftarrow \theta - \eta \frac{1}{m} \sum_{i=1}^m \nabla_\theta \ell(f_\theta(\tilde{x}_i), y_i)$
  end for
end for
```

Sitawarin et al. (2020) independently propose adversarial training with early stopping (ATES). Along with our FAT, ATES corroborates the new formulation (Section 3.1) for adversarial training.

5.1. Selection of Step τ

As shown in Section 2.2, the conventional PGD-K is a special case, when step $\tau = K$ in PGD-K-τ. Thus, standard adversarial training is a special case of FAT. Here, we investigate how step τ affects the performance of FAT empirically, and summarize that larger τ may not increase adversarial robustness but hurt the standard test accuracy. Detailed experimental setups of Figure 4 are in Appendix F.1.

Figure 4 shows that, with the increase of τ, the standard test accuracy for natural data decreases significantly; while the robust test accuracy for adversarial data increases at smaller values of τ but reaches its plateau at larger values of τ. For example, when τ is bigger than 2, the standard test accuracy continues to decrease with larger τ. However, the robust test accuracy begins to maintain a plateau for both Small CNN and ResNet-18. Such observation manifests that a larger step τ may not be necessary for adversarial training. Namely, it may not increase adversarial robustness but hurt the standard accuracy. This reflects a trade-off between the standard accuracy and adversarially robust accuracy (Tsipras et al., 2019) and suggests that our step τ helps manage this trade-off.

τ can be treated as a hyper-parameter. Based on the observations in Figure 4, it is enough to select τ from the set $\{0, 1, 2, 3\}$. Note that the size of the set is also influenced by step size α and maximum PGD step K. In Section 6.2, we use τ to fine-tune the performance of FAT.

5.2. Smaller τ is Computationally Efficient

Adversarial training is time-consuming since it needs multiple backward propagations (BPs) to produce adversarial data. The time-consuming factor depends on the number of BPs used for generating adversarial data (Shafahi et al., 2019; Zhang et al., 2019a; Wong et al., 2020; B.S. & Babu, 2020).

Our FAT uses PGD-K-τ to generate adversarial data, and PGD-K-τ is early-stopped. This implies that FAT is computationally efficient, since FAT does not need to compute maximum K BPs on each mini-batch. To illustrate this, we count the number of BPs for generating adversarial data during training. The training setup is the same as the one in Section 5.1, but we only choose τ from $\{0, 1, 2, 3\}$ and train for 100 epochs with learning rate divided by 10 at epochs 60 and 90. In Figure 5, we compare the standard adversarial training (Madry et al., 2018) (dashed line) with our FAT (solid line) and adversarial training TRADES (Zhang et al., 2019b) (dashed line) with our FAT for TRADES (solid line, detailed in Algorithm 4 in Appendix D.2). For each epoch, we compute average BPs over all training data for generating the adversarial counterpart.

Figure 5 shows that conventional adversarial training uses PGD-K which takes K BPs for generating adversarial data in each mini-batch. By contrast, our adversarial training

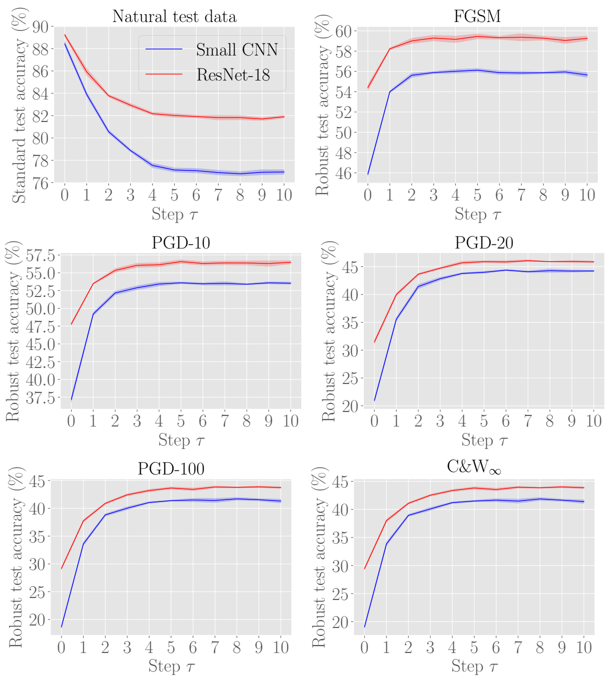


Figure 4. We conduct our adversarial training FAT with various values of step τ on two networks Small CNN (blue line) and ResNet-18 (red line). We evaluate the adversarial training performance according to networks' standard test accuracy for natural test data and robust test accuracy for adversarial test data generated by FGSM, PGD-10, PGD-20, PGD-100 and C&W attack. We report the median test accuracy and its standard deviation as the shaded color over 5 repeated trials of adversarial training.

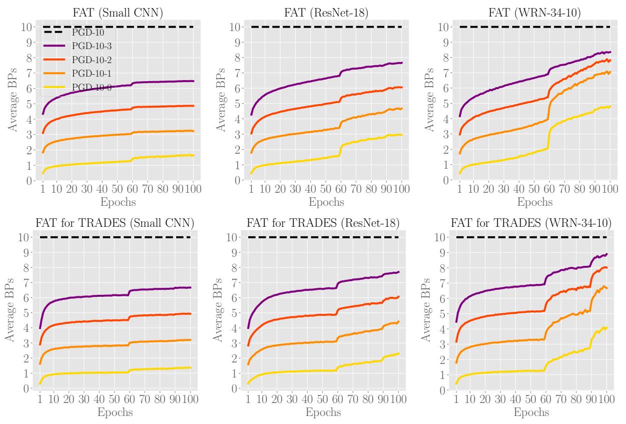


Figure 5. Training statistics on the average number of BPs needed for generating adversarial data. Top three panels are adversarial training FAT on three networks, Small CNN (top left), ResNet-18 (top middle) and WRN-34-10 (top right). Bottom three panels are adversarial training FAT for TRADES on three networks, Small CNN (bottom left), ResNet-18 (bottom middle) and WRN-34-10 (bottom right).

FAT that uses PGD-K-τ significantly reduces the number of required BPs. In addition, with the smaller τ, FAT needs less BPs on average for generating adversarial data. The magnitude of τ controls the number of extra BPs, once misclassified adversarial data is found.

Moreover, there are some interesting phenomena observed in our FAT (or FAT for TRADES). As the training progresses, the number of BPs gradually increases. This shows that more steps are needed by PGD to find misclassified adversarial data. This signifies that it is increasingly difficult to find adversarial data that misclassifies the model. Thus, DNNs become more and more adversarially robust over training epochs. In addition, there is a slight surge in average BPs at epochs 60 and 90, where we divided the learning rate by 10. This means that the robustness of the model gets substantially improved at epochs 60 and 90. It is a common trick to decrease the learning rate over training DNNs for good standard accuracy (He et al., 2016). Figure 5 confirms that it is similarly meaningful to decrease the learning rate during adversarial training.

5.3. Relation to Curriculum Learning

Curriculum learning (Bengio et al., 2009) is a machine learning strategy that gradually makes the learning task more difficult. Curriculum learning is shown effective in improving standard generalization and providing faster convergence (Bengio et al., 2009).

In adversarial training, curriculum learning can also be used to improve adversarial robustness. Namely, DNNs learn from milder adversarial data first, and gradually adapt to stronger adversarial data. There are different ways to determine the hardness of adversarial data. For example, curriculum adversarial training (CAT) uses the perturbation step K of PGD as the hardness measure (Cai et al., 2018). Dynamic adversarial training (DAT) uses their proposed criterion, the first-order stationary condition (FOSC), as the hardness measure (Wang et al., 2019). However, both methods do not have a principled way to decide when the hardness should be increased during training. To increase the hardness at the right time, both methods need domain knowledge to fine-tune the curriculum training sequence. For example, CAT needs to decide when to increase step K in PGD over training epochs; while DAT needs to decide FOSC for generating adversarial data at different training stages.

Our FAT can also be regarded as a type of curriculum training. As shown in Figure 5, as the training progresses, more and more backward propagations (BPs) are needed to generate adversarial data to fool the classifier. Thus, more and more PGD steps are needed to generate adversarial data. Meanwhile, the network gradually and automatically learns from stronger and stronger adversarial data (adversarial data generated by more and more PGD steps). Differently from CAT and DAT, FAT could automatically increase the hardness of friendly adversarial data based on the model's predictions. As a result, Table 1 in Section 6.2 shows that empirical results of FAT can outperform the best results in CAT (Cai et al., 2018), DAT (Wang et al., 2019) and standard Madry's adversarial training (not a curriculum learning) (Madry et al., 2018). Thus, attacks which do not kill training indeed make adversarial learning stronger.

6. Experiments

To evaluate the efficacy of FAT, we firstly use CIFAR-10 (Krizhevsky, 2009) and SVHN (Netzer et al., 2011) datasets to verify that FAT can help achieve a larger perturbation bound ϵ_{train}. Then, we train Wide ResNet (Zagoruyko & Komodakis, 2016) on the CIFAR-10 dataset to achieve state-of-the-art results.

6.1. FAT can Enable Larger Perturbation Bound ϵ_{train}

All images of CIFAR-10 and SVHN are normalized into $[0, 1]$. We compare our FAT ($\tau = 0, 1, 3$) and standard adversarial training (Madry) on ResNet-18 with different perturbation bounds ϵ_{train}, i.e., $\epsilon_{train} \in [0.03, 0.15]$ for CIFAR-10 in Figure 6 and $\epsilon_{train} \in [0.01, 0.06]$ for SVHN in Figure 7. The maximum PGD step $K = 10$, step size $\alpha = \epsilon/10$. DNNs were trained using SGD with 0.9 momentum for 80 epochs with the initial learning rate of 0.01 divided by 10 at epoch 60.

In addition, we have experiments of training a different DNN, e.g., Small CNN. We also set maximum PGD step

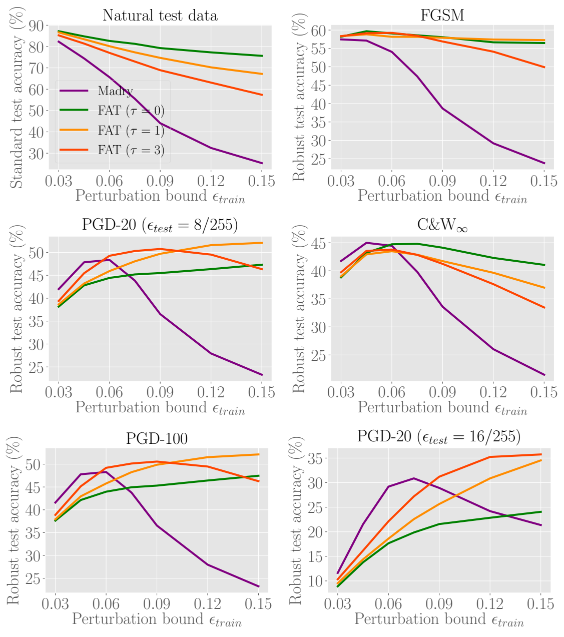


Figure 6. Comparisons on standard test accuracy and robust test accuracy of the deep model (ResNet-18) trained by standard adversarial training (Madry) and our friendly adversarial training ($\tau = 0, 1, 3$) respectively on CIFAR-10 dataset.

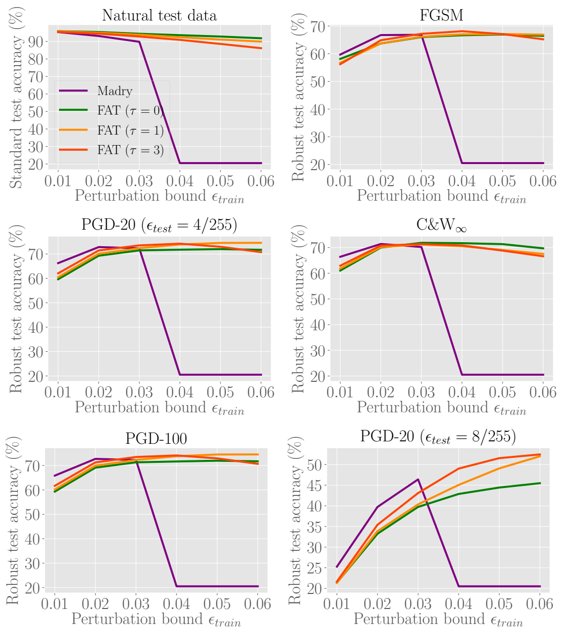


Figure 7. Comparisons on standard test accuracy and robust test accuracy of deep model (ResNet-18) trained by standard adversarial training (Madry) and our friendly adversarial training ($\tau = 0, 1, 3$) respectively on SVHN dataset.

$K = 20$. Besides, we compare our FAT for TRADES and TRADES (Zhang et al., 2019b) with different values of ϵ_{train}. These extensive results are presented in Appendix G.

Figures 6 and 7 show the performance of FAT ($\tau = 0, 1, 3$) and standard adversarial training w.r.t. standard test accuracy and adversarially robust test accuracy of the DNNs. We obtain standard test accuracy for natural test data and robust test accuracy for adversarial test data. The adversarial test data are bounded by L_∞ perturbations with $\epsilon_{test} = 8/255$ for CIFAR-10 and $\epsilon_{test} = 4/255$ for SVHN, which are generated by FGSM, PGD-20, PGD-100 and C&W$_\infty$ (L_∞ version of C&W optimized by PGD-30 (Carlini & Wagner, 2017)). Moreover, we also evaluate the robust DNNs using stronger adversarial test data generated by PGD-20 with a larger perturbation bound $\epsilon_{test} = 16/255$ for CIFAR-10 and $\epsilon_{test} = 8/255$ for SVHN (bottom right panels in Figures 6 and 7). All PGD attacks have random start, i.e, the uniformly random perturbation of $[-\epsilon_{test}, \epsilon_{test}]$ added to the natural test data before PGD perturbations. Step size α of PGD is fixed to 2/255.

From top-left panels of Figures 6 and 7, DNNs trained by FAT ($\tau = 0, 1, 3$) have higher standard test accuracy compared with those trained by standard adversarial training (i.e., Madry). This gap significantly widens as perturbation bound ϵ_{train} increases. Larger ϵ_{train} will allow the generated adversarial data deviate more from natural data. In standard adversarial training, natural generalization is significantly hurt with larger ϵ_{train} due to the cross-over mixture issue. In contrast, DNNs trained by FAT could have better standard generalization, which is less affected by an increasing perturbation bound ϵ_{train}.

In addition, with the increase of the perturbation bound ϵ_{train}, robust test accuracy (e.g., PGD and C&W) of DNNs trained by standard adversarial training (Madry) gets a slight increase first but is followed by a sharp drop. For a larger ϵ_{train} (e.g., $\epsilon_{train} > 0.06$ in CIFAR-10 and $\epsilon_{train} > 0.03$ in SVHN), standard adversarial training (Madry) basically fails, and thus its robust test accuracy drops sharply. Without early-stopped PGD, the generated adversarial data has a severe cross-over mixture problem, which makes the adversarial learning extremely difficult and sometimes even "kills" the learning.

However, it is still meaningful to enable a stronger defense over a weaker attack, i.e., ϵ_{train} in adversarial training should be larger than ϵ_{test} in adversarial attack. The right bottom panels in Figures 6 and 7 show for the stronger attack ($\epsilon_{test} = 16/255$ for CIFAR-10 and $\epsilon_{test} = 8/255$ for SVHN), it is meaningful to have larger ϵ_{train} to attain a better robustness. Our FAT is able to achieve larger ϵ_{train}. Figures 6 and 7 show our FAT ($\tau = 0, 1, 3$) maintain higher robust accuracy with larger ϵ_{train}.

Note that the performance of FAT with PGD-10-3 ($\tau = 3$, red lines) in Figure 6 drops with $\epsilon_{train} > 0.09$. We believe that FAT with larger τ could also have the issue of cross-

Table 1. Evaluations (test accuracy) of deep models (WRN-32-10) on CIFAR-10 dataset

Defense	Natural	FGSM	PGD-20	C&W$_\infty$	PGD-100
Madry	87.30	56.10	45.80	46.80	-
CAT	77.43	57.17	46.06	42.28	-
DAT	85.03	63.53	48.70	47.27	-
FAT ($\epsilon_{train} = 8/255$)	**89.34** $\pm$ 0.221	65.52 $\pm$ 0.355	46.13 $\pm$ 0.409	46.82 $\pm$ 0.517	45.31 $\pm$ 0.531
FAT ($\epsilon_{train} = 16/255$)	87.00 $\pm$ 0.203	**65.94** $\pm$ 0.244	**49.86** $\pm$ 0.328	**48.65** $\pm$ 0.176	**49.56** $\pm$ 0.255

Results of Madry, CAT and DAT are reported in (Wang et al., 2019). FAT has the same evaluations.

Table 2. Evaluations (test accuracy) of deep models (WRN-34-10) on CIFAR-10 dataset

Defense	Natural	FGSM	PGD-20	C&W$_\infty$	PGD-100
TRADES ($\beta = 1.0$)	88.64	56.38	49.14	-	-
FAT for TRADES ($\epsilon_{train} = 8/255$)	**89.94** $\pm$ 0.303	**61.00** $\pm$ 0.418	**49.70** $\pm$ 0.653	49.35 $\pm$ 0.363	48.35 $\pm$ 0.240
TRADES ($\beta = 6.0$)	84.92	61.06	56.61	**54.47**	55.47
FAT for TRADES ($\epsilon_{train} = 8/255$)	**86.60** $\pm$ 0.548	**61.97** $\pm$ 0.570	55.98 $\pm$ 0.209	54.29 $\pm$ 0.173	55.34 $\pm$ 0.291
FAT for TRADES ($\epsilon_{train} = 16/255$)	84.39 $\pm$ 0.030	61.73 $\pm$ 0.131	**57.12** $\pm$ 0.233	54.36 $\pm$ 0.177	**56.07** $\pm$ 0.155

Results of TRADES ($\beta = 1.0$ and 6.0) are reported in (Zhang et al., 2019b). FAT for TRADES has the same evaluations.

over mixture, which is detrimental to adversarial learning. In addition, both standard adversarial training and friendly adversarial training do not perform well under C&W attack with larger ϵ_{train} (e.g., $\epsilon_{train} > 0.075$ in Figure 6), we believe it is due to the mismatch between PGD-adversarial training and C&W attack. We discuss the reasons in detail in Appendix G.4.

To sum up, deep models by FAT with $\tau = 0$ (green line) have higher standard test accuracy, but lower robust test accuracy. By increasing τ to 1, deep models have slightly reduced standard test accuracy but have the increased adversarial robustness. This sheds light on the importance of τ, which handles the trade-off between robustness and standard accuracy. In order not to "kill" the training at the initial stage, we could vary τ from a smaller value to a larger value over training epochs. In addition, due to benefits that our FAT could enable larger ϵ_{train}, we could also make ϵ_{train} larger over training. Those tricks echo our paper's philosophy of "attacks which do not kill training make adversarial training stronger". In the next subsection, we unleash the full power of FAT (and FAT for TRADES) and show its superior performance over the state-of-the-art methods.

6.2. Performance Evaluations on Wide ResNet

To manifest the full power of friendly adversarial training, we adversarially train Wide ResNet (Zagoruyko & Komodakis, 2016) to achieve the state-of-the-art performance on CIFAR-10. Similar to (Wang et al., 2019; Zhang et al., 2019b), we employ WRN-32-10 (Table 1) and WRN-34-10 (Table 2) as our deep models.

In Table 1, we compare FAT with standard adversarial training (Madry) (Madry et al., 2018), CAT (Cai et al., 2018) and DAT (Wang et al., 2019) on WRN-32-10. Training and evaluation details are in Appendix H.1. The performance evaluations are done exactly as in DAT (Wang et al., 2019). In Table 2, we compare FAT for TRADES with TRADES (Zhang et al., 2019b) on WRN-34-10. Training and evaluation details are in Appendix H.2. The performance evaluations are done exactly as in TRADES (Zhang et al., 2019b).

Moreover, Nakkiran (2019) states that robust classification needs more complex classifiers (exponentially more complex). We employ FAT for TRADES on even larger WRN-58-10, the performance gets further improved over WRN-34-10 (Appendix H.2). Moreover, we also apply the early-stopped PGD to MART, namely, FAT for MART (in Appendix E.2). As a result, the performance gets improved (detailed in Appendix H.3).

Tables 1 and 2 and results in Appendix H justify the efficacy of friendly adversarial training - adversarial robustness can indeed be achieved without compromising the natural generalization. In addition, we are even able to attain the state-of-the-art robustness.

7. Conclusion

This paper has proposed a novel formulation for adversarial training. Friendly adversarial training (FAT) approximately realizes this formulation by stopping the PGD early. FAT is computationally efficient and adheres to the spirit of curriculum training. In addition, FAT helps to relieve the problem of cross-over mixture. As a result, FAT can train deep models with larger perturbation bounds ϵ_{train}. Finally, FAT can achieve competitive performance on the large capacity networks. Further research includes (a) how to choose optimal step τ in FAT algorithm, (b) besides PGD-K-τ, how to search for friendly adversarial data effectively, and (c) theoretically studying adversarially robust generalization (Yin et al., 2019), e.g., through the lens of Rademacher complexity (Bartlett & Mendelson, 2002).

Acknowledgments

This research is supported by the National Research Foundation, Singapore under its Strategic Capability Research Centres Funding Initiative (MK, JZ), JST AIP Acceleration Research Grant Number JPMJCR20U3, Japan (GN, MS), National Key R&D Program No.2017YFB1400100, the NSFC No.91846205, the Shandong Key R&D Program No.2018YFJH0506 (LC), the Early Career Scheme (ECS) through the Research Grants Council of Hong Kong under Grant No.22200720 (BH), HKBU Tier-1 Start-up Grant (BH) and HKBU CSD Start-up Grant (BH). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.

References

Abdi, H. and Williams, L. J. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459, 2010.

Alaifari, R., Alberti, G. S., and Gauksson, T. Adef: an iterative algorithm to construct adversarial deformations. In *ICLR*, 2019.

Alayrac, J., Uesato, J., Huang, P., Fawzi, A., Stanforth, R., and Kohli, P. Are labels required for improving adversarial robustness? In *NeurIPS*, 2019.

Athalye, A., Carlini, N., and Wagner, D. A. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018.

Balunovic, M. and Vechev, M. Adversarial training and provable defenses: Bridging the gap. In *ICLR*, 2020.

Bartlett, P. L. and Mendelson, S. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. Curriculum learning. In *ICML*, 2009.

B.S., V. and Babu, R. V. Single-step adversarial training with dropout scheduling. In *CVPR*, 2020.

Buch, V. H., Ahmed, I., and Maruthappu, M. Artificial intelligence in medicine: current trends and future possibilities. *Br J Gen Pract.*, 68(668):143–144, 2018.

Cai, Q., Liu, C., and Song, D. Curriculum adversarial training. In *IJCAI*, 2018.

Carlini, N. and Wagner, D. A. Towards evaluating the robustness of neural networks. In *Symposium on Security and Privacy (SP)*, 2017.

Carmon, Y., Raghunathan, A., Schmidt, L., Liang, P., and Duchi, J. C. Unlabeled data improves adversarial robustness. In *NeurIPS*, 2019.

Cisse, M., Bojanowski, P., Grave, E., Dauphin, Y., and Usunier, N. Parseval networks: Improving robustness to adversarial examples. In *ICML*, 2017.

Cohen, J. M., Rosenfeld, E., and Kolter, J. Z. Certified adversarial robustness via randomized smoothing. In *ICML*, 2019.

Farnia, F., Zhang, J. M., and Tse, D. Generalizable adversarial training via spectral normalization. In *ICLR*, 2019.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, 2016.

Hein, M. and Andriushchenko, M. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *NeurIPS*, 2017.

Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. Adversarial examples are not bugs, they are features. In *NeurIPS*. 2019.

Krizhevsky, A. Learning multiple layers of features from tiny images. Technical report, 2009.

Lécuyer, M., Atlidakis, V., Geambasu, R., Hsu, D., and Jana, S. Certified robustness to adversarial examples with differential privacy. In *Symposium on Security and Privacy (SP)*, 2019.

Litman, T. *Autonomous vehicle implementation predictions*. Victoria Transport Policy Institute Victoria, Canada, 2017.

Maaten, L. v. d. and Hinton, G. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov): 2579–2605, 2008.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Miyato, T., Maeda, S., Koyama, M., Nakae, K., and Ishii, S. Distributional smoothing by virtual adversarial examples. In *ICLR*, 2016.

Najafi, A., Maeda, S., Koyama, M., and Miyato, T. Robustness to adversarial perturbations in learning from incomplete data. In *NeurIPS*, 2019.

Nakkiran, P. Adversarial robustness may be at odds with simplicity. *arXiv:1901.00532*, 2019.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. In *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Pang, T., Xu, K., Du, C., Chen, N., and Zhu, J. Improving adversarial robustness via promoting ensemble diversity. In *ICML*, 2019.

Raghunathan, A., Xie, S. M., Yang, F., Duchi, J., and Liang, P. Understanding and mitigating the tradeoff between robustness and accuracy. In *ICML*, 2020.

Rice, L., Wong, E., and Kolter, J. Z. Overfitting in adversarially robust deep learning. In *ICML*. 2019.

Shafahi, A., Najibi, M., Ghiasi, M. A., Xu, Z., Dickerson, J., Studer, C., Davis, L. S., Taylor, G., and Goldstein, T. Adversarial training for free! In *NeurIPS*. 2019.

Shamir, A., Safran, I., Ronen, E., and Dunkelman, O. A simple explanation for the existence of adversarial examples with small hamming distance. *arXiv:1901.10861*, 2019.

Sitawarin, C., Chakraborty, S., and Wagner, D. Improving adversarial robustness through progressive hardening. *arXiv:2003.09347*, 2020.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. In *ICLR*, 2014.

Tramr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. Ensemble adversarial training: Attacks and defenses. In *ICLR*, 2018.

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. Robustness may be at odds with accuracy. In *ICLR*, 2019.

Tsuzuku, Y., Sato, I., and Sugiyama, M. Lipschitz-Margin training: Scalable certification of perturbation invariance for deep neural networks. In *NeurIPS*, 2018.

Wang, Y., Ma, X., Bailey, J., Yi, J., Zhou, B., and Gu, Q. On the convergence and robustness of adversarial training. In *ICML*, 2019.

Wang, Y., Zou, D., Yi, J., Bailey, J., Ma, X., and Gu, Q. Improving adversarial robustness requires revisiting misclassified examples. In *ICLR*, 2020.

Weng, T., Zhang, H., Chen, P., Yi, J., Su, D., Gao, Y., Hsieh, C., and Daniel, L. Evaluating the robustness of neural networks: An extreme value theory approach. In *ICLR*, 2018.

Wong, E. and Kolter, J. Z. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *ICML*, 2018.

Wong, E., Rice, L., and Kolter, J. Z. Fast is better than free: Revisiting adversarial training. In *ICLR*, 2020.

Xiao, C., Zhu, J., Li, B., He, W., Liu, M., and Song, D. Spatially transformed adversarial examples. In *ICLR*, 2018.

Yan, Z., Guo, Y., and Zhang, C. Deep defense: Training dnns with improved adversarial robustness. In *NeurIPS*, 2018.

Yin, D., Ramchandran, K., and Bartlett, P. L. Rademacher complexity for adversarially robust generalization. In *ICML*, 2019.

Zagoruyko, S. and Komodakis, N. Wide residual networks. *arXiv:1605.07146*, 2016.

Zhang, D., Zhang, T., Lu, Y., Zhu, Z., and Dong, B. You only propagate once: Accelerating adversarial training via maximal principle. In *NeurIPS*. 2019a.

Zhang, H., Yu, Y., Jiao, J., Xing, E. P., Ghaoui, L. E., and Jordan, M. I. Theoretically principled trade-off between robustness and accuracy. In *ICML*, 2019b.

Zhang, H., Chen, H., Xiao, C., Gowal, S., Stanforth, R., Li, B., Boning, D., and Hsieh, C.-J. Towards stable and efficient training of verifiably robust neural networks. In *ICLR*, 2020.

A. Friendly Adversarial Training

For completeness, besides the learning objective by loss value, we also give the learning objective of FAT by class probability. Then, based on the learning objective by loss value, we give the proof of Theorem 1, which theoretically justifies FAT.

A.1. Learning Objective

Case 1 (by loss value, restated). The outer minimization still follows Eq. (3). However, instead of generating $\tilde{x}_i$ via inner maximization, we generate $\tilde{x}_i$ as follows:

$$\tilde{x}_i = \mathop{\arg\min}_{\tilde{x}\in\mathcal{B}_\epsilon[x_i]} \ell(f(\tilde{x}), y_i)$$
$$\text{s.t. } \ell(f(\tilde{x}), y_i) - \min\nolimits_{y\in\mathcal{Y}} \ell(f(\tilde{x}), y) \geq \rho.$$

Note that the operator $\arg\max$ in Eq. (4) is replaced with $\arg\min$ here, and there is a constraint on the margin of loss values (i.e., the mis-classification confidence).

The constraint firstly ensures $y_i \neq \arg\min_{y\in\mathcal{Y}} \ell(f(\tilde{x}), y)$ or $\tilde{x}$ is mis-classified, and secondly ensures for $\tilde{x}$ the wrong prediction is better than the desired prediction y_i by at least ρ in terms of the loss value. Among all such $\tilde{x}$ satisfying the constraint, we select the one minimizing $\ell(f(\tilde{x}), y_i)$. Namely, we minimize the adversarial loss given that a confident adversarial data has been found. This $\tilde{x}_i$ could be regarded as a friend among the adversaries, which is termed as friendly adversarial data.

Case 2 (by class probability). We redefine the above objective from the loss point of view (above) to the class probability point of view. The objective is still Eq. (3), in which $\ell(f(\tilde{x}), y) = \ell_\mathrm{B}(\ell_\mathrm{L}(f(\tilde{x})), y)$. Hence, $\ell_\mathrm{L}(f(\tilde{x}))$ is an estimate of the class-posterior probability $p(y \mid \tilde{x})$, and for convenience, denote by $p_f(y \mid \tilde{x})$ the y-th element of the vector $\ell_\mathrm{L}(f(\tilde{x}))$.

$$\tilde{x}_i = \mathop{\arg\max}_{\tilde{x}\in\mathcal{B}_\epsilon[x_i]} p_f(y_i \mid \tilde{x})$$
$$\text{s.t. } \max\nolimits_{y\in\mathcal{Y}} p_f(y \mid \tilde{x}) - p_f(y_i \mid \tilde{x}) \geq \rho.$$

The constraint ensures $\tilde{x}$ is misclassified by at least ρ, but here the margin ρ is applied to the class probability instead of the loss value. Hence, $\tilde{x}_i$ should usually be different from the one according to the loss value.

A.2. Proofs

We derive a tight upper bound on adversarially robust risk (adversarial risk), and provide our theoretical analysis for the adversarial risk minimization. X and Y represent random variables. Adversarial risk $\mathcal{R}_\mathrm{rob}(f) := \mathbb{E}_{(X,Y)\sim\mathcal{D}}\mathbb{1}\{\exists X' \in \mathcal{B}_\epsilon[X] : f(X') \neq Y\}$. $\mathcal{R}_\mathrm{rob}(f)$ can be decomposed, i.e., $\mathcal{R}_\mathrm{rob}(f) = \mathcal{R}_{nat}(f) + \mathcal{R}_\mathrm{bdy}(f)$, where natural risk $\mathcal{R}_\mathrm{nat}(f) = \mathbb{E}_{(X,Y)\sim\mathcal{D}}\mathbb{1}\{f(X) \neq Y\}$ and boundary risk $\mathcal{R}_\mathrm{bdy}(f) = \mathbb{E}_{(X,Y)\sim\mathcal{D}}\mathbb{1}\{X \in \mathcal{B}_\epsilon[DB(f)], f(X) = Y\}$. Note that $\mathcal{B}_\epsilon[DB(f)]$ is the set denoting the decision boundary of f, i.e., $\{x \in \mathcal{X} : \exists x' \in \mathcal{B}_\epsilon[x] \quad \text{s.t.} \quad f(x) \neq f(x')\}$.

Lemma 1. For any classifier $f : \mathcal{X} \to \mathcal{Y}$, any probability distribution $\mathcal{D}$ on $\mathcal{X} \times \mathcal{Y}$, we have

$$\mathcal{R}_\mathrm{rob}(f) = \mathcal{R}_\mathrm{nat}(f) + \mathbb{E}_{(X,Y)\sim\mathcal{D}}\mathbb{1}\{\exists X' \in \mathcal{B}_\epsilon[X] : f(X) \neq f(X')\} \cdot \mathbb{1}\{f(X) = Y\}.$$

Proof. By the equation $\mathcal{R}_\mathrm{rob}(f) = \mathcal{R}_\mathrm{nat}(f) + \mathcal{R}_\mathrm{bdy}(f)$,

$$\begin{aligned}
\mathcal{R}_\mathrm{rob}(f) &= \mathcal{R}_\mathrm{nat}(f) + \mathcal{R}_\mathrm{bdy}(f) \\
&= \mathcal{R}_\mathrm{nat}(f) + \mathbb{E}_{(X,Y)\sim\mathcal{D}}\mathbb{1}\{X' \in \mathcal{B}_\epsilon[DB(f)], f(X) = Y\} \\
&= \mathcal{R}_\mathrm{nat}(f) + \Pr[X' \in \mathcal{B}_\epsilon[DB(f)], f(X) = Y] \\
&= \mathcal{R}_\mathrm{nat}(f) + \Pr[f(X) \neq f(X'), f(X) = Y] \\
&= \mathcal{R}_\mathrm{nat}(f) + \mathbb{E}_{(X,Y)\sim\mathcal{D}}\mathbb{1}\{\exists X' \in \mathcal{B}_\epsilon[X] : f(X) \neq f(X'), f(X) = Y)\} \\
&= \mathcal{R}_\mathrm{nat}(f) + \mathbb{E}_{(X,Y)\sim\mathcal{D}}\mathbb{1}\{\exists X' \in \mathcal{B}_\epsilon[X] : f(X) \neq f(X')\} \cdot \mathbb{1}\{f(X) = Y\}
\end{aligned}$$

The fourth equality comes from the definition of decision boundary of f, i.e., $\mathcal{B}_\epsilon[DB(f)] = \{x \in \mathcal{X} : \exists x' \in \mathcal{B}_\epsilon[x] \quad \text{s.t.} \quad f(x) \neq f(x')\}$. □

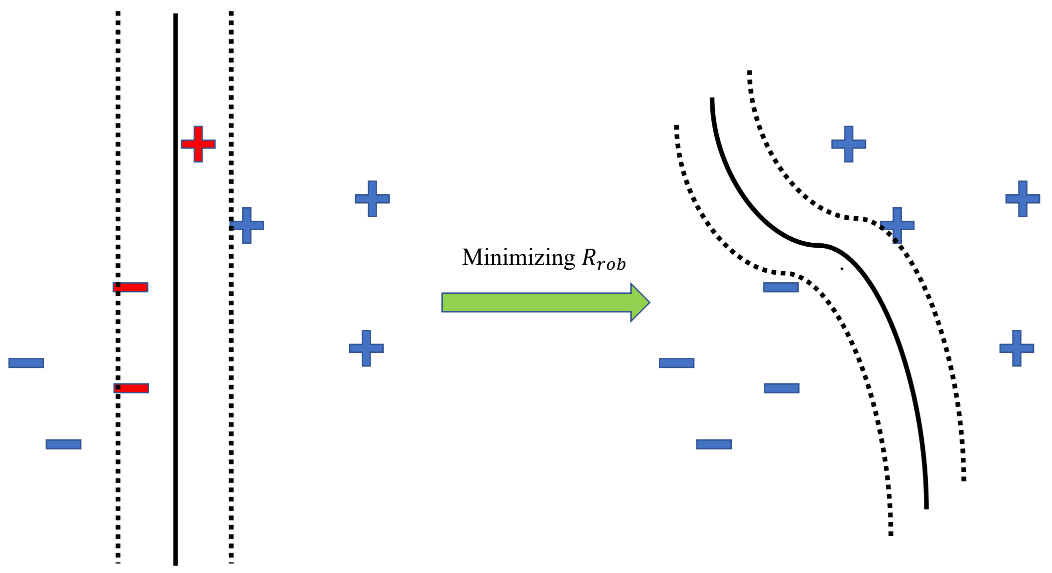


Figure 8. Solid line is the classifier. The area between dashed line is decision boundary of the classifier. Minimizing robust risk R_{rob} is to find a classifier, where data is less likely located within decision boundary of the classifier.

Figure 8 illustrates the message of Lemma 1. Minimizing robust risk R_{rob} encourages the learning algorithm to find a classifier whose decision boundary contains less training data. Meanwhile, the classifier should correctly separate data from different classes. Finding such a classifier is hard. As we can see in Figure 8, the hypothesis set of hyperplanes is enough for minimizing the natural risk (the left figure). However, a robust classifier (the right figure) has much curvatures, which is more complicated. Nakkiran (2019) states that robust classification needs more complex classifiers (exponentially more complex, in some examples). This implies that in order to learn a robust classifier, our learning algorithm needs (a) setting a large hypothesis set and (b) fine-tuning the decision boundary.

Theorem 1 (restated). For any classifier f, any non-negative surrogate loss function ℓ which upper bounds $0/1$ loss, and any probability distribution $\mathcal{D}$, we have

$$\mathcal{R}_{\mathrm{rob}}(f) \leq \underbrace{\mathbb{E}_{(X,Y)\sim\mathcal{D}}\ell(f(X),Y)}_{\text{For standard test accuracy}} + \underbrace{\mathbb{E}_{(X,Y)\sim\mathcal{D},X'\in\mathcal{B}_\epsilon[X,\epsilon]}\ell^*(f(X'),Y)}_{\text{For robust test accuracy}},$$

where

$$\ell^* = \begin{cases} \min \ell(f(X'),Y) + \rho, & \text{if } f(X') \neq Y; \\ \max \ell(f(X'),Y), & \text{if } f(X') = Y. \end{cases}$$

ρ is the small constant.

Proof.

$$\begin{aligned}
\mathcal{R}_{\text{rob}}(f) &= \mathcal{R}_{\text{nat}}(f) + \mathcal{R}_{\text{bdy}}(f) \\
&\leq \mathbb{E}_{(X,Y)\sim\mathcal{D}}\ell(f(X),Y) + \mathcal{R}_{\text{bdy}}(f) \\
&= \mathbb{E}_{(X,Y)\sim\mathcal{D}}\ell(f(X),Y) + \mathbb{E}_{(X,Y)\sim\mathcal{D}}\mathbb{1}\{X \in \mathcal{B}_\epsilon[DB(f)], f(X)=Y\} \\
&= \mathbb{E}_{(X,Y)\sim\mathcal{D}}\ell(f(X),Y) + \Pr[X \in \mathcal{B}_\epsilon[DB(f)], f(X)=Y] \\
&= \mathbb{E}_{(X,Y)\sim\mathcal{D}}\ell(f(X),Y) + \Pr[f(X) \neq f(X'), f(X)=Y] \\
&\leq \mathbb{E}_{(X,Y)\sim\mathcal{D}}\ell(f(X),Y) + \Pr[f(X') \neq Y] \\
&= \mathbb{E}_{(X,Y)\sim\mathcal{D}}\ell(f(X),Y) + \mathbb{E}_{(X,Y)\sim\mathcal{D}}\mathbb{1}\{\exists X' \in \mathcal{B}_\epsilon[X] : f(X') \neq Y\} \\
&\leq \mathbb{E}_{(X,Y)\sim\mathcal{D}}\ell(f(X),Y) + \mathbb{E}_{(X,Y)\sim\mathcal{D}, X' \in \mathcal{B}_\epsilon[X,\epsilon]}\ell^*(f(X'),Y)
\end{aligned}$$

The first inequality comes from the assumption that surrogate loss function ℓ upper bound $0/1$ loss function. The second inequality comes from the fact that there exists misclassified natural data within the decision boundary set. Therefore, $\Pr[f(X) \neq f(X'), f(X)=Y] \cup \Pr[f(X) \neq f(X'), f(X) \neq Y] = \Pr[f(X') \neq Y]$. The third inequality comes from the assumption that surrogate loss function ℓ upper bound $0/1$ loss function, i.e., in Figure 2, the adversarial data X' (purple triangle) is on line of logistic loss (blue line), which is always above the 0/1 loss (yellow line). □

Our Theorem 1 informs our strategy to fine tune the decision boundary. To fine-tune the decision boundary, the data "near" the classifier plays an important role. Those data are easily wrongly predicted with small perturbations. As we show in the Figure 2, when adversarial data are wrongly predicted, our adversarial data (purple triangle) increases to minimize the loss by a violation of a small constant ρ. Thus, our adversarial data can help fine-tune the decision boundary "bit by bit" over the training.

B. Alternative Adversarial Data Searching Algorithm

In this section, we give an alternative adversarial data searching algorithm to generate friendly adversarial data via modifying the method to update $\tilde{x}$ in Algorithm 1. We remove the constraint of ϵ-ball projection in Eq. (5), i.e.,

$$x^{(t+1)} = x^{(t)} + \alpha \operatorname{sign}(\nabla_{x^{(t)}} \ell(f_\theta(x^{(t)}), y)), \forall t \geq 0 \tag{6}$$

where x^0 is a natural data and $\alpha > 0$ is step size.

We employ Small CNN and ResNet-18 to show the performance of deep models against FGSM, PGD-20, PGD-100 and C&W$_\infty$ in Figure 9 and Figure 10. Deep models are trained using SGD with 0.9 momentum for 80 epochs with the initial learning rate 0.01 divided by 10 at 60 epoch. We compare FAT combined with the alternative adversarial data searching algorithm ($\tau = 0, 1, 3$) and standard adversarial training (Madry) with different step size α, i.e., $\alpha \in [0.003, 0.015]$. The maximum PGD step K is fixed to 10. All the testing settings are the same as those are stated in Section 6.1.

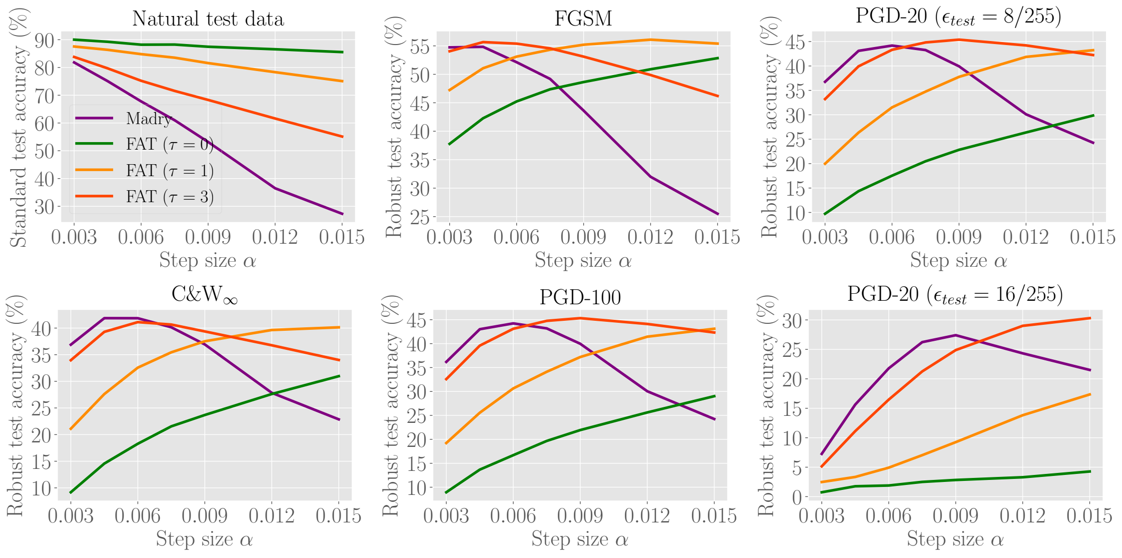


Figure 9. Test accuracy of Small CNN trained under different step size α on CIFAR-10

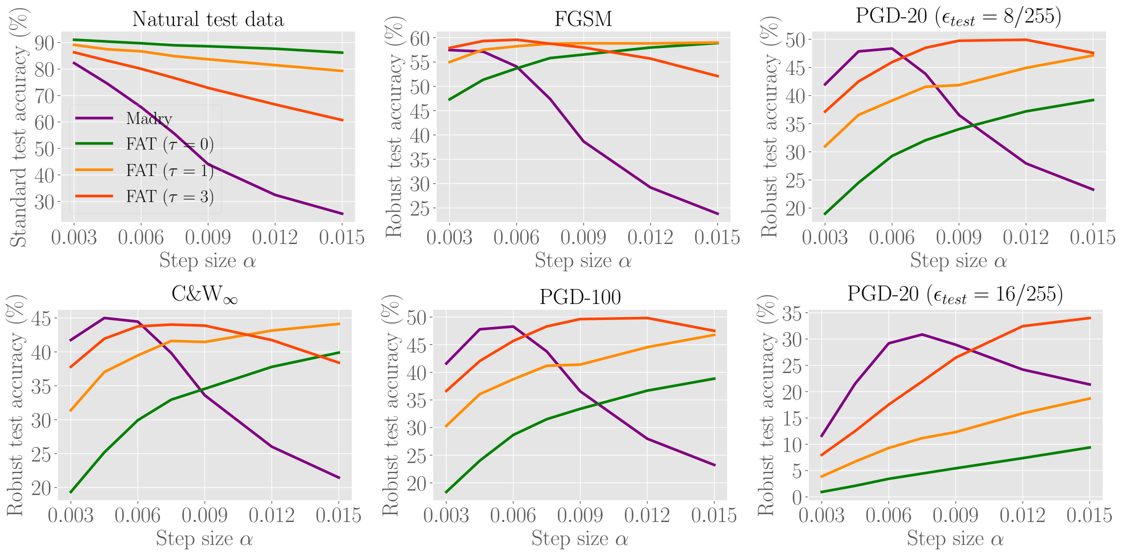


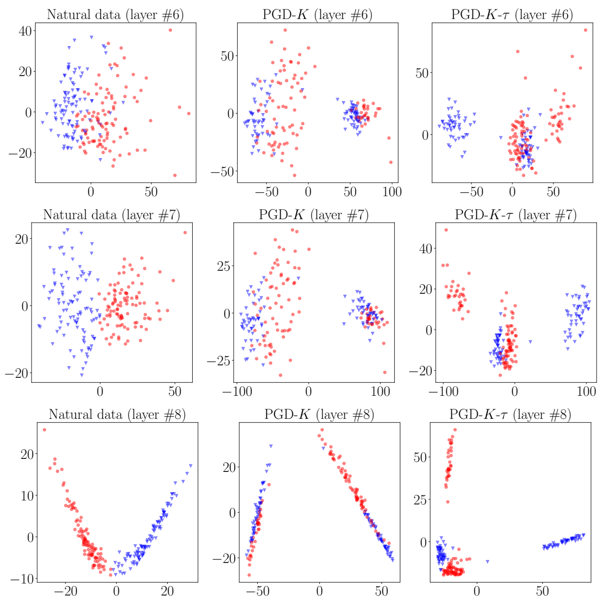
Figure 10. Test accuracy of ResNet-18 trained under different step size α on CIFAR-10

C. Mixture Alleviation

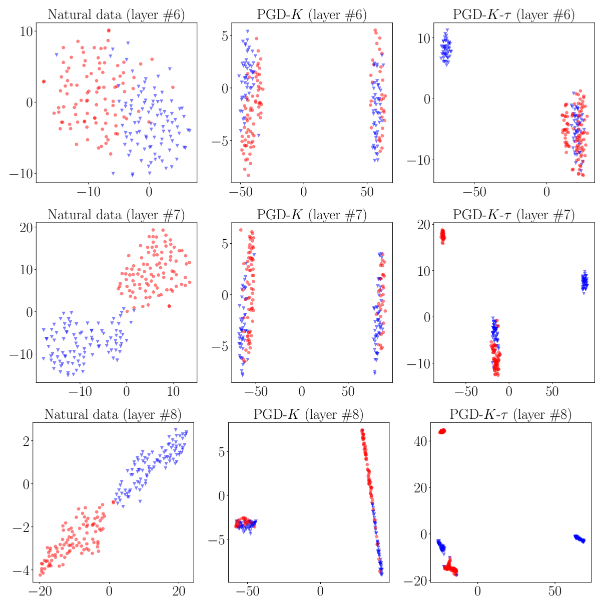
C.1. Output Distributions of Small CNN's Intermediate Layers

In Figure 3 in Section 4.2, we only visualize layer #7s output distribution by Small CNN (8-layer convolutional neural network with 6 convolutional layers and 2 fully connected layers). For completeness, we visualize the output distributions of layers #7 and #8.

We conduct warm-up training using natural training data of two randomly selected classes (bird and deer) in CIFAR-10, then involve its adversarial variants generated by PGD-20 with step size $\alpha = 0.007$ and maximum perturbation $\epsilon = 0.031$. We show output distributions of layer #6, #7 and #8 by PCA in Figure 11(a) and t-SNE in Figure 11(b).



(a) Output distributions visualized by PCA

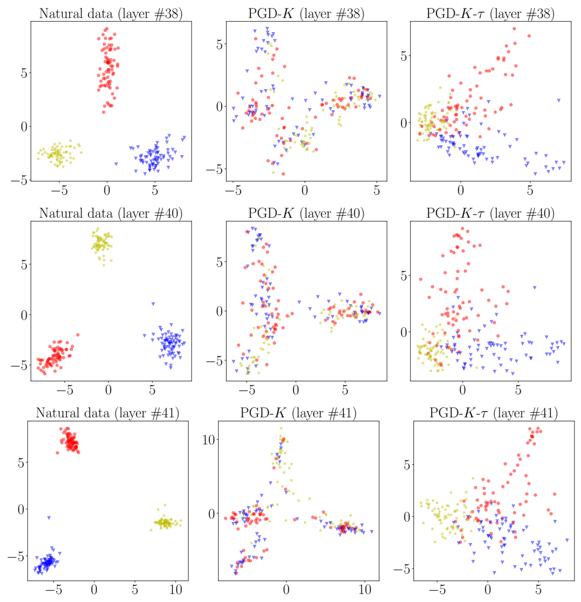


(b) Output distributions visualized by t-SNE

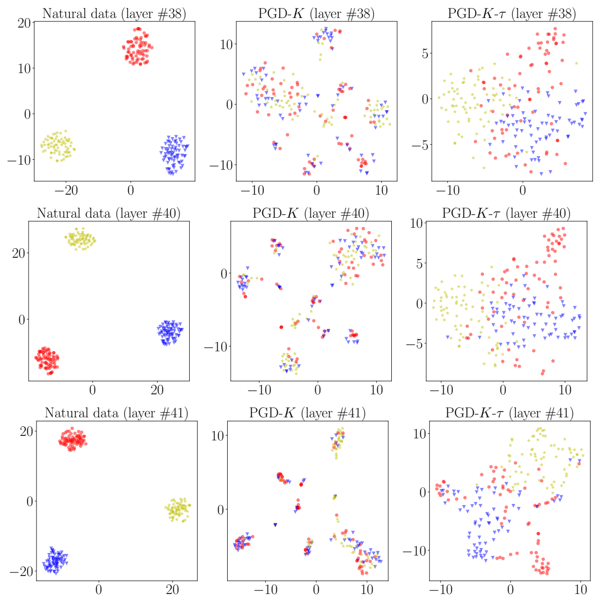
Figure 11. Output distributions of Small CNN's intermediate layers. Left column: Intermediate layers' output distributions on natural data (not mixed). Middle column: Intermediate layers' output distributions on adversarial data generated by PGD-20 (significantly mixed). Right column: Intermediate layers' output distributions on friendly adversarial data generated by PGD-20-0 (no significantly mixed).

C.2. Output distributions of WRN-40-4's intermediate layers

We train a Wide ResNet (WRN-40-4, totally 41 layers) using natural data on 10 classes in CIFAR-10 and then include adversarial variants. We randomly select 3 classes (deer, horse and truck) for illustrating output distributions of WRN-40-4's intermediate layers. Adversarial data are generated by PGD-20 with step size $\alpha = 0.007$ and maximum perturbation $\epsilon = 0.031$ on WRN-40-4. We show output distributions by layer #38, #40 and #41 by PCA in Figure 12(a) and t-SNE in Figure 12(b).



(a) Output distributions visualized by PCA



(b) Output distributions visualized by t-SNE

Figure 12. Output distributions of WRN-40-4's intermediate layers. Left column: Intermediate layers' output distributions on natural data (not mixed). Middle column: Intermediate layers' output distributions on adversarial data generated by PGD-20 (significantly mixed). Right column: Intermediate layers' output distributions on friendly adversarial data generated by PGD-20-0 (no significantly mixed).

D. FAT for TRADES

D.1. Learning Objective of TRADES

Besides the standard adversarial training, TRADES is another effective adversarial training method (Zhang et al., 2019b), which trains on both natural data x and adversarial data $\tilde{x}$.

Similar to virtual adversarial training (VAT) adding a regularization term to the loss function (Miyato et al., 2016), which regularizes the output distribution by its local sensitivity of the output w.r.t. input, the objective of TRADES is

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \left\{ \ell(f(x_i), y_i) + \beta \ell_{KL}(f(\tilde{x}_i), f(x_i)) \right\}, \tag{7}$$

where $\beta > 0$ is a regularization parameter, which controls the trade-off between standard accuracy and robustness accuracy, i.e., as β increases, standard accuracy will decease while robustness accuracy will increase, and vice visa. Meanwhile, $\tilde{x}_i$ in TRADES is dynamically generated by

$$\tilde{x}_i = \arg\max\nolimits_{\tilde{x} \in \mathcal{B}_\epsilon[x_i]} \ell_{KL}(f(\tilde{x}), f(x)), \tag{8}$$

and ℓ_{KL} is Kullback-Leibler loss that is calculated by

$$\ell_{KL}(f(\tilde{x}), f(x)) = \sum_{i=1}^{C} \ell_{\mathrm{L}}^{i}(f(x)) \log \left(\frac{\ell_{\mathrm{L}}^{i}(f(x))}{\ell_{\mathrm{L}}^{i}(f(\tilde{x}))} \right).$$

D.2. FAT for TRADES - Realization

Algorithm 3 PGD-K-τ (Early Stopped PGD for TRADES)

```
Input: data $x \in \mathcal{X}$, label $y \in \mathcal{Y}$, model $f$, loss function $\ell_{KL}$, maximum PGD step $K$, step $\tau$, perturbation bound $\epsilon$, step size $\alpha$
Output: $\tilde{x}$
$\tilde{x} \leftarrow x + \xi \mathcal{N}(\mathbf{0}, \mathbf{I})$
while $K > 0$ do
  if $\arg\max_i f(\tilde{x}) \neq y$ and $\tau = 0$ then
    break
  else if $\arg\max_i f(\tilde{x}) \neq y$ then
    $\tau \leftarrow \tau - 1$
  end if
  $\tilde{x} \leftarrow \Pi_{\mathcal{B}[x,\epsilon]}\big(\alpha\, \mathrm{sign}(\nabla_{\tilde{x}} \ell_{KL}(f(\tilde{x}), f(x)) + \tilde{x}\big)$
  $K \leftarrow K - 1$
end while
```

In Algorithm 3, $\mathcal{N}(\mathbf{0}, \mathbf{I})$ generates a random unit vector of d dimension. ξ is a small constant. ℓ_{KL} is Kullback-Leibler loss.

Given a dataset $S = \{(x_i, y_i)\}_{i=1}^{n}$, where $x_i \in \mathcal{R}^d$ and $y_i \in \{0, 1, ..., C-1\}$, adversarial training (TRADES) with early stopped PGD-K-τ returns a classifier θ^*:

$$\theta^* = \arg\min_{\theta} \sum_{i=1}^{n} \left\{ \ell_{CE}(f_\theta(x_i), y_i) + \beta \ell_{KL}(f_\theta(\tilde{x}_i), f_\theta(x_i)) \right\} \tag{9}$$

where $f_\theta : \mathcal{R}^d \to \mathcal{R}^C$ is DNN classification function, $f_\theta(\cdot)$ outputs predicted probability over C classes, The adversarial data $\tilde{x}_i$ of x_i is dynamically generated according to Algorithm 3, $\beta > 0$ is a regularization parameter, ℓ_{CE} is cross-entropy loss, ℓ_{KL} is Kullback-Leibler loss.

Based on our early stopped PGD-K-τ for TRADES in Algorithm 3, our friendly adversarial training for TRADES (FAT for TRADES) is

Algorithm 4 Friendly Adversarial Training for TRADES (FAT for TRADES)

```
**Input:** network $f_\theta$, training dataset $S = \{(x_i, y_i)\}_{i=1}^n$, learning rate $\eta$, number of epochs $T$, batch size $m$, number of batches $M$
**Output:** adversarially robust network $f_\theta$
for epoch $= 1, \dots, T$ do
  for mini-batch $= 1, \dots, M$ do
    Sample a mini-batch $\{(x_i, y_i)\}_{i=1}^m$ from $S$
    for $i = 1, \dots, m$ (in parallel) do
      Obtain adversarial data $\tilde{x}_i$of $x_i$ by Algorithm 3
    end for
    $\theta \leftarrow \theta - \eta \frac{1}{m} \sum_{i-1}^{m} \nabla_\theta \left[ \ell_{CE}(f_\theta(\tilde{x}_i), y_i) + \beta \ell_{KL}(f_\theta(\tilde{x}_i), f_\theta(x_i)) \right]$
  end for
end for
```

E. FAT for MART

E.1. Learning Objective of MART

MART (Wang et al., 2020) emphasizes the importance of misclassified natural data on the adversarial robustness. Wang et al. (2020) propose a regularized adversarial learning objective which contains an explicit differentiation of misclassified data as the regularizer. The learning objective of MART is

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \left\{ \ell_{BCE}(f(\tilde{x}_i), y_i) + \beta \cdot \ell_{KL}(f(\tilde{x}_i), f(x_i)) \cdot (1 - \ell_L^{y_i}(f(x_i))) \right\}, \tag{10}$$

where $\beta > 0$ is a regularization parameter which balances the two parts of the final loss. ℓ_{KL} is Kullback-Leibler loss. ℓ_L^k stands for the k-th element of the soft-max output and $\tilde{x}_i$ in MART is dynamically generated according to Eq. 4 that is realized by PGD-K. The first part ℓ_{BCE} is the proposed BCE loss in MART that is calculated by

$$\ell_{BCE}(f(\tilde{x}_i), y_i) = -\log(\ell_L^{y_i}(f(\tilde{x}_i))) - \log(1 - \max_{k \neq y_i} \ell_L^k(f(\tilde{x}_i)))$$

where the first term $-\log(\ell_L^{y_i}(f(\tilde{x}_i)))$ is the cross-entropy loss and the second term $-\log(1 - \max_{k \neq y_i} \ell_L^k(f(\tilde{x}_i)))$ is a margin term used to increase the distance between $\ell_L^{y_i}(f(\tilde{x}_i))$ and $\max_{k \neq y_i} \ell_L^k(f(\tilde{x}_i))$. This is a similar to C&W (Carlini & Wagner, 2017) attack that is to improve attack strength. For the second part, they combine Kullback-Leibler loss (Zhang et al., 2019b) and emphases on misclassified examples. This part of loss will be large for misclassified examples and small for correctly classified examples.

E.2. FAT for MART - Realization

Given a dataset $S = \{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathcal{R}^d$ and $y_i \in \{0, 1, ..., C-1\}$, FAT for MART returns a classifier θ^*:

$$\theta^* = \arg\min_{\theta} \sum_{i=1}^{n} \left\{ \ell_{BCE}(f_\theta(\tilde{x}_i), y_i) + \beta \cdot \ell_{KL}(f_\theta(\tilde{x}_i), f_\theta(x_i)) \cdot (1 - \ell_L^{y_i}(f_\theta(x_i))) \right\} \tag{11}$$

where $f_\theta : \mathcal{R}^d \to \mathcal{R}^C$ is DNN classification function, $f_\theta(\cdot)$ outputs predicted probability over C classes. The adversarial data $\tilde{x}_i$ of x_i is dynamically generated according to Algorithm 1, ℓ_L is the soft-max activation, $\beta > 0$ is a regularization parameter, ℓ_{BCE} is the proposed BCE loss in MART and ℓ_{KL} is Kullback-Leibler loss. Based on our early stopped PGD-K-τ in Algorithm 1, FAT for MART Algorithm 5.

F. Experimental Setup

F.1. Selection of Step τ

Figure 4 presents empirical results on CIFAR-10 via our FAT algorithm,where we train 8-layer convolutional neural network (Small CNN, blue line) and 18-layer residual neural network (ResNet-18, red line) (He et al., 2016). The maximum step

Algorithm 5 Friendly Adversarial Training for MART (FAT for MART)

```
Input: network $f_\theta$, training dataset $S = \{(x_i, y_i)\}_{i=1}^n$, learning rate $\eta$, number of epochs $T$, batch size $m$, number of batches $M$
Output: adversarially robust network $f_\theta$
for epoch $= 1, \dots, T$ do
  for mini-batch $= 1, \dots, M$ do
    Sample a mini-batch $\{(x_i, y_i)\}_{i=1}^m$ from $S$
    for $i = 1, \dots, m$ (in parallel) do
      Obtain adversarial data $\tilde{x}_i$of $x_i$ by Algorithm 1
    end for
    $\theta \leftarrow \theta - \eta \frac{1}{m} \sum_{i-1}^{m} \nabla_\theta \left[ \ell_{BCE}(f_\theta(\tilde{x}_i), y_i) + \beta \cdot \ell_{KL}(f_\theta(\tilde{x}_i), f_\theta(x_i)) \cdot (1 - \ell_L^{y_i}(f_\theta(x_i))) \right]$
  end for
end for
```

$K = 10$, $\epsilon_{train} = 8/255$, step size $\alpha = 0.007$, and step $\tau \in \{0, 1, \dots, 10\}$. We train deep networks for 80 epochs using SGD with 0.9 momentum, where learning rate starts at 0.1 and divided by 10 at 60 epoch.

For each τ, we take five trials, where each trial will obtain standard test accuracy evaluated on natural test data and robust test accuracy evaluated on adversarial test data that are generated by attacks FGSM (Goodfellow et al., 2015), PGD-10 and PGD-20, PGD-100 (Madry et al., 2018) and C&W attack (Carlini & Wagner, 2017) respectively. All those attacks are white box attacks, which are constrained by the same perturbation bound $\epsilon_{test} = 8/255$. Following Zhang et al. (2019b), all attacks have the random start, and the step size α in PGD-10, PGD-20, PGD-100 and C&W is fixed to 0.003.

G. Supplementary Experiments - FAT Enabling Larger ϵ_{train}

In this section, we provide extensive experimental results. The test settings are the same as those are stated in Section 6.1. In Section G.1, instead of using ResNet-18, we conduct adversarial training on the deep model of Small CNN. In Section G.2, instead of applying FAT, we compare our FAT for TRADES and TRADE (Zhang et al., 2019b) under different values of perturbation bound ϵ_{train} on the deep models ResNet-18 and Small CNN. In Section G.3, we set maximum PGD steps $K = 20$ and report results of FAT and FAT for TRADES over existing methods with larger perturbation bound ϵ_{train}. To sum up, all those extensive results verify that FAT and FAT for TRADES can enable deep models trained under larger values of perturbation bound ϵ_{train}.

G.1. A Different Deep Model - Small CNN

We train Small CNN on CIFAR-10 and SVHN using the same settings as those stated in Section 6.1. We show standard and robust test accuracy of deep model (Small CNN) on CIFAR-10 dataset (Figure 13) and SVHN dataset (Figure 14).

G.2. FAT for TRADES

We apply FAT for TRADES(Algorithm 4) to Small CNN and ResNet-18 on CIFAR-10 dataset. All training settings are the same as those are stated in Section 6.1. Regularization parameter $\beta = 6$. We present standard and robust test results of Small CNN (Figure 15) and ResNet-18 (Figure 16).

G.3. Maximum PGD Step $K = 20$

By setting maximum PGD step $K = 20$, we conduct more experiments on Small CNN and ResNet-18 using FAT and FAT for TRADES. Except maximum PGD steps $K = 20$, training settings are the same as those are stated in Section 6.1. Test results of robust deep models are shown in Figures 17, 18, 19 and 20.

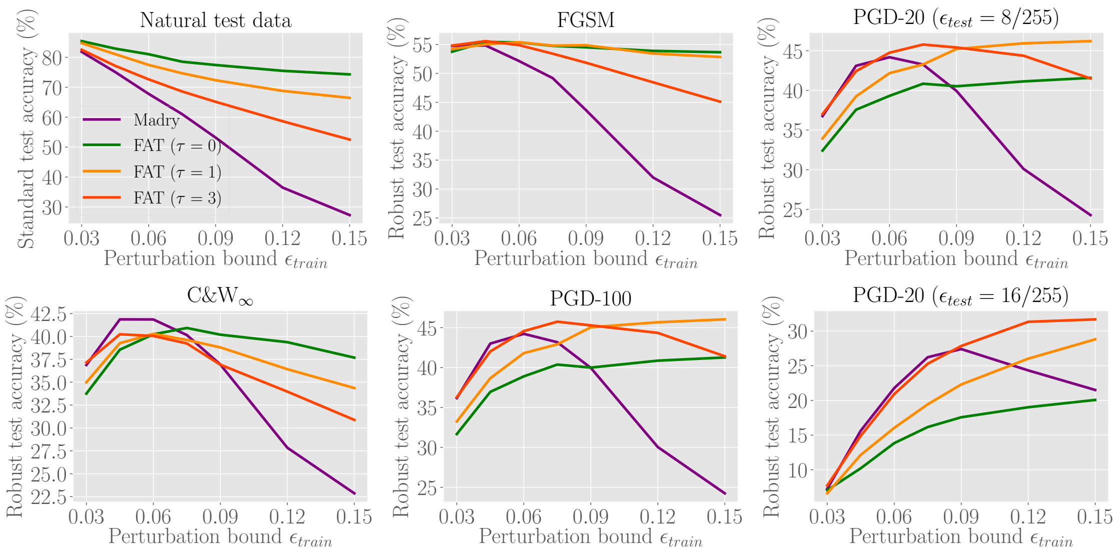


Figure 13. Test accuracy of Small CNN trained under different values of ϵ_{train} on CIFAR-10 dataset.

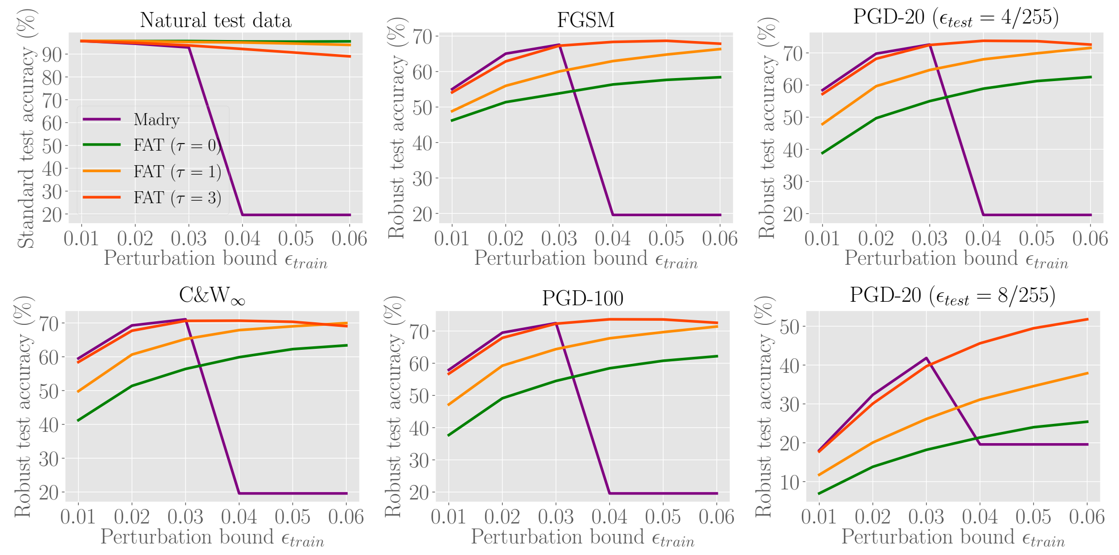


Figure 14. Test accuracy of Small CNN trained under different values of ϵ_{train} on SVHN dataset.

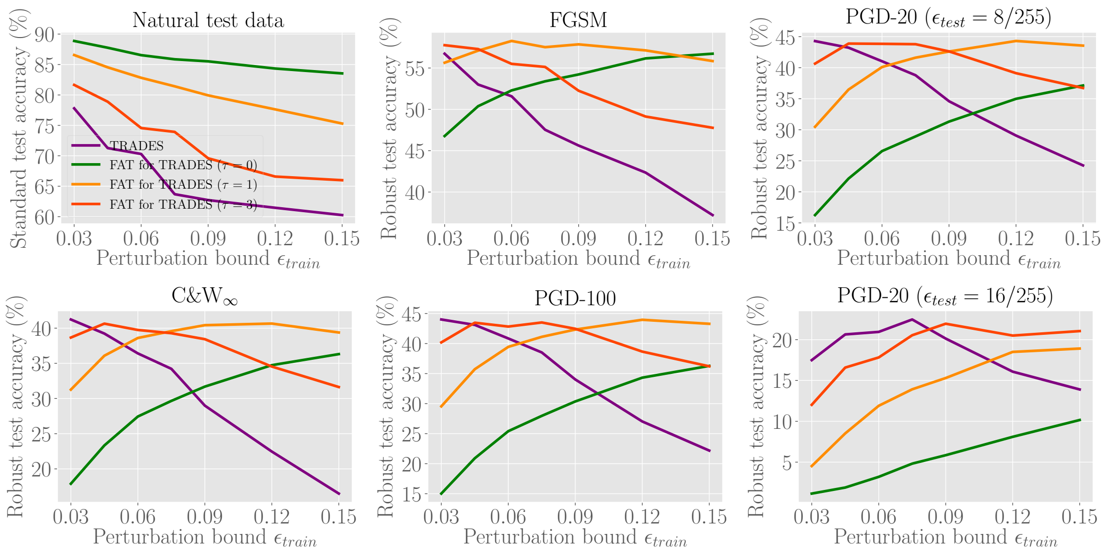


Figure 15. Test accuracy of Small CNN trained by FAT for TRADES ($\tau = 0, 1, 3$) and TRADES under different values of ϵ_{train} on CIFAR-10 dataset.

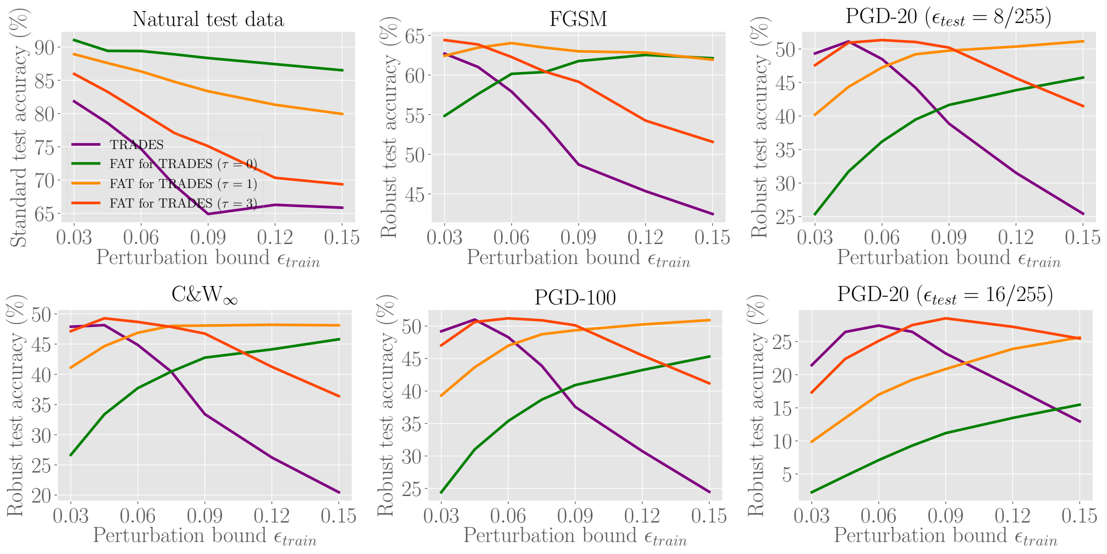


Figure 16. Test accuracy of ResNet-18 trained by FAT for TRADES ($\tau = 0, 1, 3$) and TRADES under different values of ϵ_{train} on CIFAR-10 dataset.

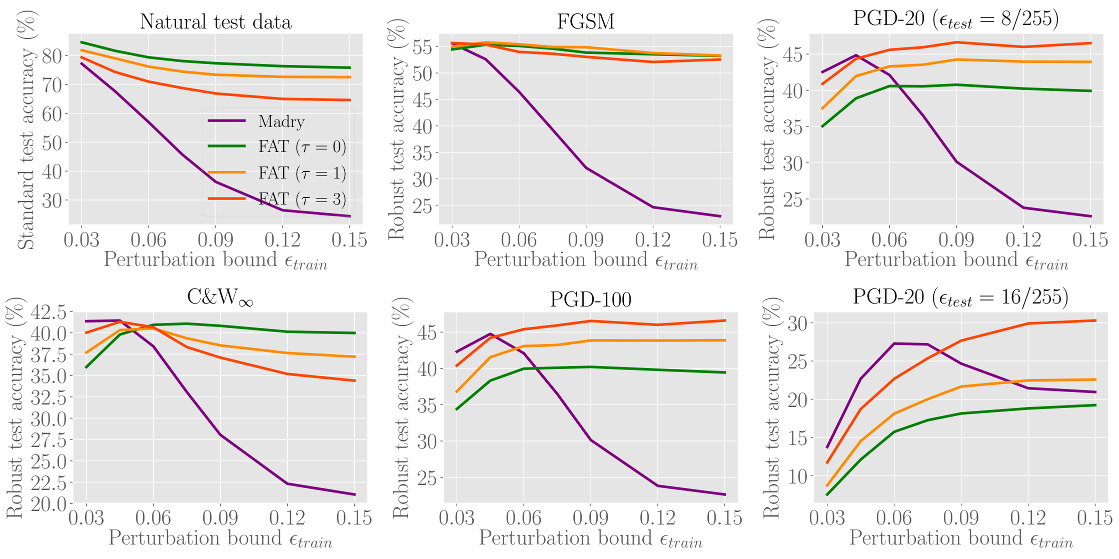


Figure 17. Test accuracy of Small CNN trained by FAT and standard adversarial training (Madry) with maximum PGD step $K = 20$ under different values of ϵ_{train} on CIFAR-10 dataset.

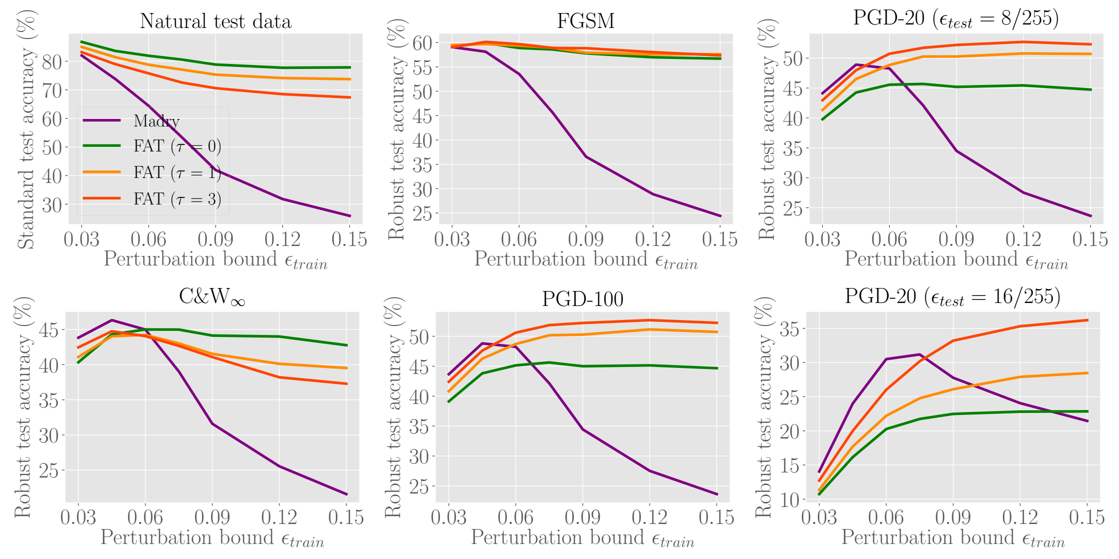


Figure 18. Test accuracy of ResNet-18 trained by FAT standard adversarial training (Madry) with maximum PGD step $K = 20$ under different values of ϵ_{train} on CIFAR-10 dataset.

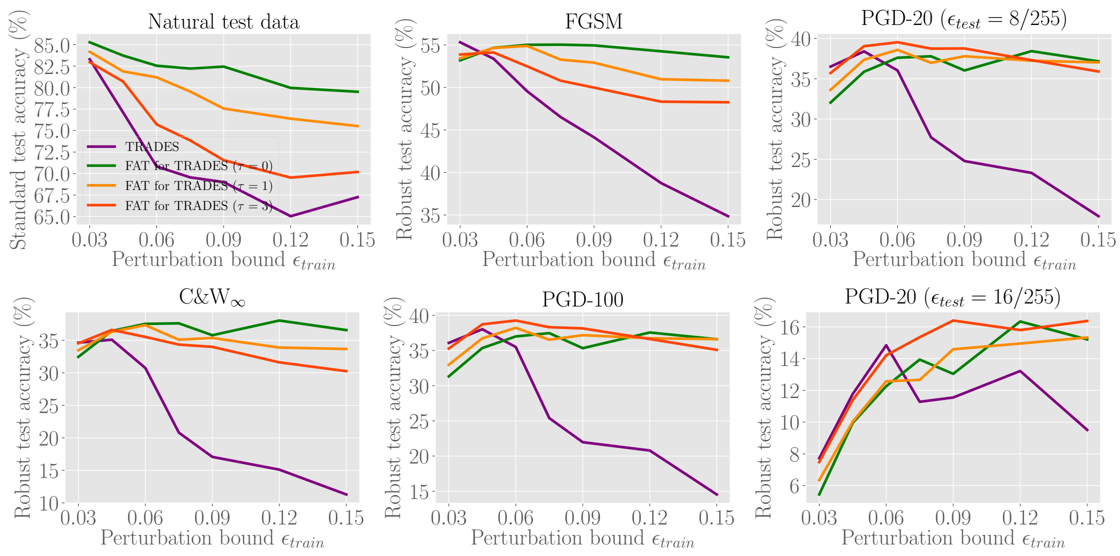


Figure 19. Test accuracy of Small CNN trained by FAT for TRADES and TRADES with maximum PGD step $K = 20$ under different values of ϵ_{train} on CIFAR-10 dataset.

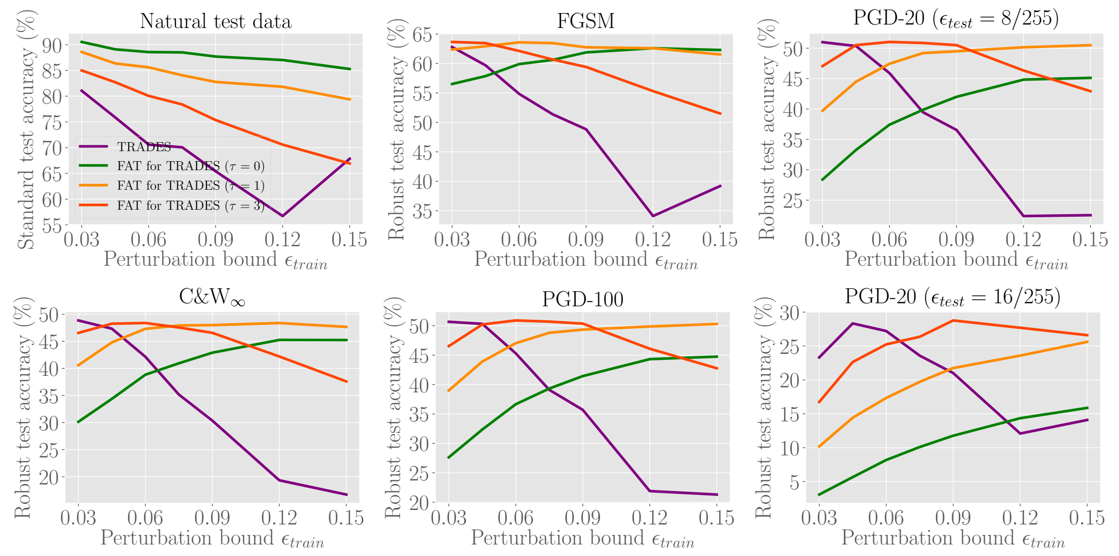


Figure 20. Test accuracy of ResNet-18 trained by FAT for TRADES and TRADES with maximum PGD step $K = 20$ under different values of ϵ_{train} on CIFAR-10 dataset.

G.4. C&W Attack Analysis

As is shown in Figure 6 along with Figure 13 in Section G.1, both standard adversarial training and friendly adversarial training do not perform well under C&W (Carlini & Wagner, 2017) attack with larger ϵ_{train} (e.g., $\epsilon_{train} > 0.075$). We discuss the reasons for these phenomena.

C&W attack. Given x, we choose a target class t and then search for adversarial data $\tilde{x}$ under C&W attack in the L_p metric by solving

$$\text{minimize} \quad \|\tilde{x} - x\|_p + c \cdot h(\tilde{x})$$

with h defined as

$$h(\tilde{x}) = \max(\max_{i \neq t} f(\tilde{x})_i - f(\tilde{x})_t, -\kappa).$$

The parameter $c > 0$ balances two parts of loss. $\kappa > 0$ encourages the solver to find adversarial data $\tilde{x}$ that will be classified as class t with high confidence. Note that this paper follows the implementation of C&W$_\infty$ attack in (Cai et al., 2018)[1] and (Wang et al., 2019)[2] where they replace the cross-entropy loss with $h(\tilde{x})$ in PGD, i.e.,

$$\tilde{x}_i = \arg\max_{\tilde{x} \in \mathcal{B}_\epsilon[x_i]} (\max_{i \neq y_i} f(\tilde{x})_i - f(\tilde{x})_{y_i} - \kappa). \tag{12}$$

Analysis. In Figure 6, with larger ϵ_{train}, the performance evaluated by PGD attacks increases, while performance evaluated by C&W attack decreases. The reason is that C&W and PGD have different ways of generating adversarial data according to Eq. (12) and Eq. (4) respectively. The two interactive methods search adversarial data in different directions due to gradients w.r.t. different loss. Therefore, the distributions of C&W and PGD adversarial data are inconsistent. As perturbation bound ϵ_{train} increases, there are more PGD adversarial data generated within ϵ_{train}-ball. A DNN learned from more PGD adversarial data becomes more defensive to PGD attacks, but this deep model may not effectively defend C&W adversarial data.

[1] curriculum adversarial training GitHub

[2] dynamic adversarial training GitHub

H. Extensive State-of-the-art Results on Wide ResNet

H.1. Training Details of FAT on WRN-32-10

In Table 1, we compare our FAT with standard adversarial training (Madry), CAT (Cai et al., 2018) and DAT (Wang et al., 2019).

We use FAT ($\epsilon_{train} = 8/255$ and $16/255$ respectively) to train WRN-32-10 for 120 epochs using SGD with 0.9 momentum, and weight decay is 0.0002. Maximum PGD step is 10 and step size is fixed to 0.007. The initial learning rate is 0.1 reduced to 0.01, 0.001 and 0.0005 at epoch 60, 90 and 110. We set step $\tau = 0$ initially and increase τ by one at epoch 50 and 90 respectively. The maximum step $K = 10$. We report performance of the deep model at the last epoch. For fair comparison, in Table 1 we use the same test settings as those in DAT (Wang et al., 2019). Performance of robust deep model is evaluated standard test accuracy for natural data and robust test accuracy for adversarial data, that are generated by FGSM, PGD-20 (20-steps PGD with random start), PGD-100 and C&W$_\infty$(L$_\infty$ version of C&W optimized by PGD-30).

All attacks have the same perturbation bound $\epsilon_{test} = 0.031$ and step size in PGD is $\alpha = \epsilon_{test}/4$. The same as DAT (Wang et al., 2019), there is random start in PGD attack, i.e., uniformly random perturbations ($[-\epsilon_{test}, +\epsilon_{test}]$) added to natural data before PGD perturbations. We report the median test accuracy and its standard deviation over 5 repeated trails of adversarial training in Table 1.

H.2. Training details of FAT for TRADES on Wide ResNet

In Table 2, we use FAT for TRADES ($\epsilon_{train} = 8/255$ and $16/255$ respectively) train WRN-34-10 by FAT for TRADES for 85 epochs using SGD with 0.9 momentum and 0.0002 weight decay. Maximum PGD step $K = 10$ and step size $\alpha = 0.007$. The initial learning rate is 0.1 and divided 10 at epoch 75. We set step $\tau = 0$ initially and increased by one at epoch 30, 50 and 70. Since TRADES has a trade-off parameter β, for fair comparison, our FAT for TRADES use the same β. In Table 2, we set $\beta = 1$ and 6 separately, which are endorsed by (Zhang et al., 2019b).

For fair comparison, we use the same test settings as those are stated in TRADES (Zhang et al., 2019b). All attacks have the same perturbation bound $\epsilon_{test} = 0.031$ (without random start), and step size $\alpha = 0.003$, which is the same as stated in the paper (Zhang et al., 2019b). Performance of robust deep model is evaluated standard test accuracy for natural data and robust test accuracy for adversarial data, that are generated by FGSM, PGD-20, PGD-100 and C&W$_\infty$(L$_\infty$ version of C&W optimized by PGD-30). We report the median test accuracy and its standard deviation of the deep model at the last epoch over 3 repeated trials of adversarial training in Table 2.

Fair comparison based on TRADES's experimental setting. However, in TRADES's experimental testing[3], they use random start before PGD perturbation that is deviated from the statements in the paper (Zhang et al., 2019b). For fair comparison, we also retest the robust deep models under PGD attacks with random start. We evaluate their publicly released robust deep model[4] WRN-34-10 and compare it with ours trained by FAT for TRADES. The test results are reported in Table 3.

FAT for TRADES on larger WRN-58-10. We employ Wide ResNet with larger capacity, i.e., WRN-58-10 to show our superior performance achieved by FAT for TRADES in Table 3. All the training settings are the same as details on WRN-34-10 in this section. The regularization parameter β is fixed to 6.0. All attacks have the same perturbation bound $\epsilon_{test} = 0.031$ and step size $\alpha = 0.003$, which is the same as TRADES's experimental setting. Robustness against FGSM, PGD-20(20-steps PGD with random start) and C&W$_\infty$ is reported in Table 3.

H.3. FAT for MART on Wide ResNet

We train WRN-34-10 by FAT for MART ($\epsilon_{train} = 8/255$ and $16/255$ respectively) using SGD with 0.9 momentum and 0.0002 weight decay. Maximum PGD step $K = 10$ and step size $\alpha = 0.007$. The initial learning rate is 0.1 and divided 10 at epoch 60 and 90 respectively. We set step $\tau = 0$ initially and increase τ by one at epoch 20, 40, 60 and 80. The regularization parameter β is fixed to 6.0. The maximum step size $K = 10$.

[3]TRADES GitHub
[4]TRADES's pre-trained model

Table 3. Robustness (test accuracy) of deep models on CIFAR-10 dataset (evaluated with random start)

Model	Defense	Natural	FGSM	PGD-20	C&W$_\infty$
WRN-34-10	TRADES ($\beta = 6.0$)	84.92	67.00	57.18	54.72
	FAT for TRADES ($\epsilon_{train} = 8/255$)	86.38 $\pm$ 0.548	67.64 $\pm$ 0.572	56.65 $\pm$ 0.262	54.51 $\pm$ 0.299
	FAT for TRADES ($\epsilon_{train} = 16/255$)	84.39 $\pm$ 0.030	67.38 $\pm$ 0.370	57.67 $\pm$ 0.198	54.62 $\pm$ 0.140
WRN-58-10	FAT for TRADES ($\epsilon_{train} = 8/255$)	**87.09**	**68.7**	57.17	55.43
	FAT for TRADES ($\epsilon_{train} = 16/255$)	85.28	68.08	**58.39**	**55.89**

Fair comparison based on MART's experimental setting. For fair comparison, all attacks have the same perturbation bound $\epsilon_{test} = 8/255$ and step size $\alpha = \epsilon_{test}/10$, which is the same setting in MART (Wang et al., 2020). White-box robustness of the deep model against attacks such as FGSM, PGD-20 (20-steps PGD with random start) and C&W$_\infty$ (L$_\infty$ version of C&W optimized by PGD-30) is reported. We evaluate Wang et al. (2020) publicly released robust deep model[5] WRN-34-10 and compare it with ours trained by FAT for MART. In Table 4, we report the median test accuracy and its standard deviation over 3 repeated trails of FAT for MART on WRN-34-10.

FAT for MART on larger WRN-58-10. In Table 4, we also employ WRN-58-10 to show the performance achieved by FAT for MART. All the training and testing settings are the same as those on WRN-34-10.

Table 4. Robustness (test accuracy) of deep models on CIFAR-10 dataset

Model	Defense	Natural	FGSM	PGD-20	C&W$_\infty$
WRN-34-10	MART ($\beta = 6.0$)	83.62	67.38	58.24	**53.67**
	FAT for MART ($\epsilon_{train} = 8/255$)	86.40 $\pm$ 0.071	68.94 $\pm$ 0.195	57.89 $\pm$ 0.144	52.28 $\pm$ 0.110
	FAT for MART ($\epsilon_{train} = 16/255$)	84.39 $\pm$ 0.390	68.52 $\pm$ 0.297	59.13 $\pm$ 0.180	52.85 $\pm$ 0.459
WRN-58-10	FAT for MART ($\epsilon_{train} = 8/255$)	**87.10**	**69.52**	58.57	52.73
	FAT for MART ($\epsilon_{train} = 16/255$)	85.19	69.00	**59.82**	53.01

[5]MART's pre-trained model