CAT: Customized Adversarial Training for Improved Robustness

Anonymous Author(s)
Affiliation
Address
email

Abstract

Adversarial training has become one of the most effective methods for improving robustness of neural networks. However, it often suffers from poor generalization on both clean and perturbed data. In this paper, we propose a new algorithm, named Customized Adversarial Training (CAT), which adaptively customizes the perturbation level and the corresponding label for each training sample in adversarial training. We show that the proposed algorithm achieves better clean and robust accuracy than previous adversarial training methods through extensive experiments.

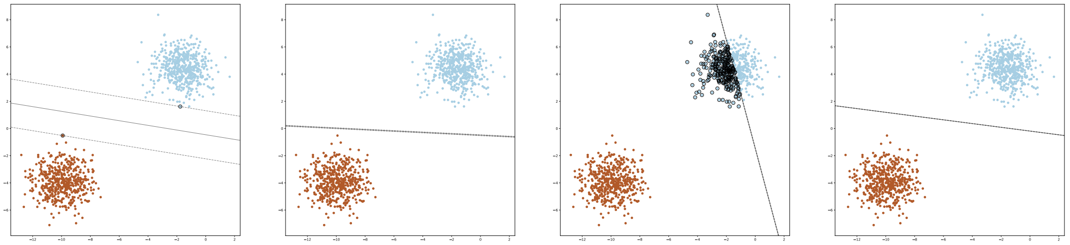
1 Introduction

Deep neural networks (DNNs) have proved their effectiveness on a variety of domains and tasks. However, it has been found that DNNs are highly vulnerable to adversarial examples [25]. To enhance the robustness of DNNs against adversarial examples, adversarial training [15, 20] has become one of the most effective and widely used methods. Given a pre-defined perturbation tolerance, denoted as ϵ, adversarial training aims to minimize the robust loss, defined as the worst-case loss within ϵ-ball around each example, leading to a min-max optimization problem. Madry et al. [20] shows that applying a multi-step projected gradient descent (PGD) attack to approximately solve the inner maximization leads to a robust model, and several recent research has proposed various ways to improve adversarial training [2, 10, 30, 32, 36].

However, standard adversarial training methods still have a hypothetical and possibly problematic assumption: the perturbation tolerance ϵ is a large and fixed constant throughout the training process, which ignores the fact that every data point may have different intrinsic robustness. Intuitively, some examples are naturally closer to the decision boundary, and enforcing large margin on those examples will force the classifier to give up on those examples, leading to a distorted decision surface. This intuition may explain the known issue of the undesirable robustness-accuracy tradeoff in adversarial robustness [24, 28]. Furthermore, with a different perturbation tolerance, it is questionable whether we should still force the model to learn to fit the one-hot label as in the original adversarial training formulation. In the extreme case, if an example is perturbed to the decision boundary, a good classifier yielding the binary class prediction probabilities should output $[0.5, 0.5]$ instead of $[1, 0]$. This aspect becomes crucial when each example is associated with a different level of perturbation. Although some recent papers have started to address the uniform ϵ issue by treating correctly and incorrectly classified examples differently [10] or assigning non-uniform perturbation level [2], none of them have tried to incorporate customized training labels in this process.

Motivated by these ideas, we propose a novel Customized Adversarial Training (CAT) framework that can substantially improve the performance of adversarial training. Throughout the adversarial training process, our algorithm dynamically finds a non-uniform and effective perturbation level and the corresponding customized target label for each example. This leads to better generalization

Submitted to 34th Conference on Neural Information Processing Systems (NeurIPS 2020). Do not distribute.



(a) Standard training (b) Adv-train with $\epsilon = 1$ (c) Adv-train with $\epsilon = 4$ (d) CAT (ours) with $\epsilon_{max} = 4$

Figure 1: Different training methods on a linearly separable binary classification dataset with 1.75 margin for both classes. Adversarial training with small ϵ works fine, but for a large ϵ beyond the true margin, adversarial training would ruin the classifier's classification performance, while our proposed adaptive customized adversarial training method still keeps a good generalization performance.

performance and furthermore, with a careful design on adaptive ϵ tuning, our algorithm has only negligible computational overhead and runs as fast as the original adversarial training algorithm. Furthermore, we theoretically explain why the proposed method could lead to improved generalization performance.

Our method significantly outperforms existing adversarial training methods on the standard CIFAR-10 defense task. With Wide-ResNet structure on CIFAR-10, under $8/255$ ℓ_∞ perturbation, our method achieves 73% robust accuracy under PGD attack and 71% robust accuracy under Carlini and Wagner (C&W) attack [4], while the current best model only achieves 58.6% under PGD attack and 56.8% under C&W attack. Furthermore, our method only degrades the clean accuracy from 95.93% (standard test accuracy) to 93.48%, while other adversarial training methods have clean accuracy below 91.34%.

2 Background and Motivation

2.1 Preliminaries

Adversarial training can be formulated as a min-max optimization problem. For a K-class classification problem, let $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1,\dots,n}$ denote the set of training samples in the dataset with $\boldsymbol{x}_i \in \mathbb{R}^d, y_i \in \{1, \dots, K\} := [K]$, we consider a classfication model $f_\theta(\boldsymbol{x}) : \mathbb{R}^d \to [K]$ parameterized by θ. We denote by $h_\theta(\boldsymbol{x}) : \mathbb{R}^d \to [0,1]^K$ as the prediction output for each class, i.e., $f_\theta(\boldsymbol{x}) = \text{argmax}_i [h_\theta(\boldsymbol{x})]_i$.

Adversarial training can be formulated as:

$$\min_\theta \frac{1}{n} \sum_{i=1}^{n} \max_{\boldsymbol{x}'_i \in \mathcal{B}_p(\boldsymbol{x}_i, \epsilon)} \ell(f_\theta(\boldsymbol{x}'_i), y_i), \tag{1}$$

where $\mathcal{B}_p(\boldsymbol{x}_i, \epsilon)$ denotes the ℓ_p-norm ball centered at $\boldsymbol{x}_i$ with radius ϵ. The inner maximization problem aims to find an adversarial version of a given data point $\boldsymbol{x}_i$ that achieves the highest loss. In general, one can define $\mathcal{B}_p(\boldsymbol{x}_i, \epsilon)$ based on the threat model, but the ℓ_∞ ball is the most popular choice adopted by recent works [10, 20, 30, 31, 36], which we also use in this paper. For a deep neural network model, the inner maximization does not have a closed form solution, so adversarial training methods typically use a gradient-based iterative solver to approximately solve the inner problem. The most commonly used choice is the multi-step PGD [20] and C&W attack [4].

2.2 Motivation

Intuitively, if adversarial training can always find a model with close-to-zero robust error, one should always use a large ϵ for training because it will automatically imply robustness to any smaller ϵ. Unfortunately, in practice a uniformly large ϵ is often harmful. In the following we empirically explain this problem and use it to motivate our proposed algorithm.

We use a simple linear classification case to demonstrate why a uniformly large ϵ is harmful. In Figure 1a, we generate a synthetic linearly separable dataset with the margin set to be 1.75 for both classes, where the correct linear boundary can be easily obtained by standard training. In Figure 1b, we run adversarial training with $\epsilon = 1$, and since this ϵ is smaller than the margin, the algorithm can still obtain near-optimal results. However, when we use a large $\epsilon = 4$ for adversarial training in Figure 1c, the resulting decision boundary becomes significantly worse. It is because adversarial training cannot correctly fit all the samples with a margin up to 4, so it will sacrifice some data samples, leading to distorted and undesirable decision boundary. This motivates the following two problems:

- We shouldn't set the same large ϵ uniformly for all samples. Some samples are intrinsically closer to the decision boundary and they should use a smaller ϵ. Without doing this, adversarial training will give up on those samples, which leads to worse training and generalization error (see more discussions in Section 3.3 on the generalization bounds).
- The adversarial training loss is trying to force the prediction to match the one-hot label (e.g., $[1, 0]$ in the binary classification case) even after large perturbations. However, if a sample is perturbed, the prediction shouldn't remain one-hot. For instance, if a sample is perturbed to the decision boundary of a binary classification problem, the prediction of a perfect model should be $[0.5, 0.5]$ instead of $[1, 0]$, which also makes adversarial training fail to recover a good decision hyperplane.

Furthermore, we observe that even if adversarial training can obtain close-to-zero training error with large ϵ (e.g., [13] proves that this will happen for overparameterized network with large-enough margin), a uniformly large ϵ will lead to larger generalization gap. This could be partially explained by the theoretical results provided by [34], which shows that the adversarial Rademacher complexity has a lower bound with an explicit dependence on the perturbation tolerance. The empirical results in Table 1 also illustrate this problem. When conducting adversarial training with $\epsilon = 0.3$ on CIFAR10 VGG-16, we found that the model achieves close-to-zero robust training error on all $\epsilon \leq 0.3$, but it suffers larger generalization gap compared to training with smaller ϵ. This also demonstrates that a uniformly large ϵ is harmful even when it achieves perfect training error.

Table 1: The influence of different fixed ϵ values used in adversarial training on the robust accuracy with $\epsilon = 0.01$.

Testing ϵ	Error Type	Training ϵ		
		0.01	0.02	0.03
0.01	Train	99.96%	99.99%	99.16%
	Test	69.79%	69.06%	66.04%

3 CAT (Customized Adversarial Training)

In this section, we propose the Customized Adversarial Training (CAT) framework that improves adversarial training by addressing the above-mentioned problems. First, our algorithm has an auto-tuning method to customize the ϵ used for each training example. Second, instead of forcing the model to fit the original label, we customize the target label for each example based on its own ϵ. In the following we will describe these two components in more detail.

3.1 Auto-tuning ϵ for adversarial training

The first component of our algorithm is an ϵ auto-tuning method which adaptively assigns a suitable ϵ for each example during the adversarial training procedure. Let ϵ_i be the perturbation level assigned to example i. Based on the intuition mentioned in Section 2.2, we do not want to further increase ϵ if we find the classifier does not have capacity to robustly classify the example, which means we should set

$$\epsilon_i = \operatorname*{argmin}_{\epsilon}\{ \max_{\boldsymbol{x}_i' \in \mathcal{B}_p(\boldsymbol{x}_i, \epsilon)} f_\theta(\boldsymbol{x}_i') \neq y_i \} \tag{2}$$

and the adversarial training objective becomes

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \max_{\boldsymbol{x}_i' \in \mathcal{B}_p(\boldsymbol{x}_i, \epsilon_i)} \ell(f_\theta(\boldsymbol{x}_i'), y_i). \tag{3}$$

Note that ϵ_i in (2) depends on θ, while θ in (3) also depends on ϵ_i. We thus propose an alternative update scheme — conducting one SGD update on θ, and then updating the ϵ_i in the current batch. However, finding ϵ_i exactly requires brute-force search for every possible value, which adds significant computational overhead to adversarial training.

Therefore, we only conduct a simplified update rule on ϵ_i as follows. Starting from an initial perturbation level of zero, at each iteration we conduct adversarial attack (e.g., PGD attack) with perturbation tolerance $\epsilon_i + \eta$ where η is a constant. If the attack is successful, then we reset current ϵ_i to 0 to encourage model learning a more robust classifier towards those examples. While if the attack is unsuccessful, which means an attacker still cannot find an adversarial example that satisfies $\max_{\boldsymbol{x}_i' \in \mathcal{B}_p(\boldsymbol{x}_i, \epsilon_i+\eta)} f_\theta(\boldsymbol{x}_i') \neq y_i$, then we increase $\epsilon_i = \epsilon_i + \eta$. The attack results will also be used to update the model parameter θ, so this adaptive scheme does not require any additional cost. In practice, we also have an upper bound on the final perturbation to ensure that ϵ_i remains bounded for each i.

3.2 Adaptive label uncertainty for adversarial training

As mentioned in Section 2.2, the standard adversarial training loss is trying to enforce a sample being classified as the original one-hot label after ϵ perturbation. However, this may not be ideal. In the extreme case, if a sample is perturbed to the decision boundary, the prediction must be far away from one-hot. This problem is more severe when using non-uniform ϵ_i, since each different ϵ_i will introduce a different bias to the loss, and that may be one of the reasons that purely adaptive ϵ-scheduling does not work well (see our ablation study in Section 5 and also the results reported in [2]).

In the following, we propose an adaptive label smoothing approach to reflect different perturbation tolerance on each example. Szegedy et al. [26] introduced label smoothing that converts one-hot label vectors into one-warm vectors representing low-confidence classification, in order to prevent the model from making over-confident predictions. Specifically, with a one-hot encoded label y, the smoothed version is

$$\tilde{y} = (1-\alpha)y + \alpha u,$$

where $\alpha \in [0, 1]$ is the hyperparameter to control the smoothing level. In the adaptive setting, we set $\alpha = c\epsilon_i$ so that a larger perturbation tolerance would receive a higher label uncertainty and c is a hyperparameter. A common choice of u is $u = \frac{1}{K}$. However, this strict requirement tries to enforce every other labels having the same probability, which may not make sense in practice. On the other hand, as shown Section 2.2, adversarial training is easy to overfit and generate a large generalization gap. To better address these issues, we sample from a distribution instead. Specifically, we use $u = \text{Dirichlet}(\boldsymbol{\beta})$ where Dirichlet$(\cdot)$ refers to the Dirichlet distribution and $\boldsymbol{\beta} \in \mathbb{R}^K$ is concentration hyperparamter. With different perturbation tolerance, the adaptive version of label smoothing is

$$\tilde{y_i} = (1 - c\epsilon_i)y_i + c\epsilon_i \text{Dirichlet}(\boldsymbol{\beta}). \tag{4}$$

The final objective function: Combining the two aforementioned techniques, our Customized Adversarial Training (CAT) method attempts to minimize the following objective:

$$\min_\theta \ \frac{1}{n}\sum_{i=1}^{n} \max_{\boldsymbol{x}_i' \in \mathcal{B}_p(\boldsymbol{x}_i, \epsilon_i)} \ell(f_\theta(\boldsymbol{x}_i'), \tilde{y_i}) \qquad \textbf{s.t. } \epsilon_i = \operatorname*{argmin}_{\epsilon} \{ \max_{\boldsymbol{x}_i' \in \mathcal{B}_p(\boldsymbol{x}_i, \epsilon)} f_\theta(\boldsymbol{x}_i') \neq y_i \}, \tag{5}$$

where $\tilde{y_i}$ is defined in (4). As described in Section 3.1, we approximately minimize this objective with an alternative update scheme, which incurs almost no additional cost compared to the original adversarial training algorithm. The detailed algorithm is presented in Algorithm 1.

Choice of loss function. In general, our framework can be used with any loss function $\ell(\cdot)$. In the previous works, cross entropy loss is commonly used for ℓ. However, the model trained by smoothing techniques tends to have a smaller logit gap between true label and other labels. Therefore, in order to encourage model to generate a larger logit gap, we propose a mixed loss to enhance the defense performance towards C&W$_\infty$ attack. That is,

$$\text{CE}(f_\theta(\boldsymbol{x}_i'), \tilde{y_i}) + \max\{\max_{j \neq y_0}\{[Z(\boldsymbol{x}_i')]_j - [Z(\boldsymbol{x}_i')]_{y_0}\}, -\kappa\}, \tag{6}$$

where $Z(\boldsymbol{x}) \in \mathbb{R}^K$ is the final (logit) layer output, and $[Z(\boldsymbol{x})]_i$ is the prediction score for the i-th class and y_0 is the original label. The parameter κ encourages the adversary to find higher confident adversarial examples in training.

Algorithm 1 CAT algorithm

```
Input: Training dataset $(X, Y)$, cross entropy loss or mix loss $\ell$, scheduling parameter $\eta$, weighting factor $c$, perturbation upperbound $\epsilon_{max}$
Initial every sample's $\epsilon_i$ with 0
for epoch=$1, \ldots, N$ do
  for i=$1, \ldots, B$ do
    $\tilde{y_i} \leftarrow (1 - c\epsilon_i)y_i + c\epsilon_i$Dirichlet($\boldsymbol{\beta}$)
    $\epsilon_i \leftarrow \epsilon_i + \eta$
    $\delta_i \leftarrow 0$
    for $j = 1 \ldots m$ do
      $\delta_i \leftarrow \delta_i + \alpha \cdot sign(\nabla_\delta \ell(f_\theta(\boldsymbol{x}_i + \delta_i), \tilde{y_i})$
      $\delta_i \leftarrow \max(\min(\delta_i, \epsilon_i), -\epsilon_i)$
    end for
    if $f_\theta(\boldsymbol{x}_i + \delta_i) \neq y_i$ then
      $\epsilon_i \leftarrow 0$
    end if
    $\epsilon_i \leftarrow \min(\epsilon_{max}, \epsilon_i)$
    $\tilde{y_i} \leftarrow (1 - c\epsilon_i)y_i + (1 - c\epsilon_i)$Dirichlet($\boldsymbol{\beta}$)
    $\theta \leftarrow \theta - \gamma_\theta \nabla_\theta \ell(f_\theta(\boldsymbol{x}_i + \delta_i), \tilde{y_i})$
  end for
end for
return $\theta$
```

3.3 Theoretical Analysis

To better understand how our scheme improves generalization, we now provide some theoretical analysis. Recall we denote by $h_\theta(\boldsymbol{x}) : \mathbb{R}^d \to [0, 1]^K$ as the prediction probability for the K classes. We define the bilateral margin that our paper is essentially maximizing over as follows.

Definition 3.1 (Bilateral margin) *We define the bilateral perturbed network output by $H_\theta(\boldsymbol{x}, \boldsymbol{\delta}^i, \boldsymbol{\delta}^o)$:*

$$H_\theta(\boldsymbol{x}, \boldsymbol{\delta}^i, \boldsymbol{\delta}^o) := h_\theta\left(\boldsymbol{x} + \boldsymbol{\delta}^i \|\boldsymbol{x}\|\right) + \left\|\boldsymbol{x} + \boldsymbol{\delta}^i \|\boldsymbol{x}\|\right\| \cdot \boldsymbol{\delta}^o.$$

The bilateral margin is now defined as the minimum norm of $(\boldsymbol{\delta}^i, \boldsymbol{\delta}^o)$ required to cause the classifier to make false predictions:

$$m_F(\boldsymbol{x}, y) := \min_{\boldsymbol{\delta}^i, \boldsymbol{\delta}^o} \sqrt{\|\boldsymbol{\delta}^i\|^2 + \|\boldsymbol{\delta}^o\|^2} \quad s.t. \quad \max_{y'} H_\theta(\boldsymbol{x}, \boldsymbol{\delta}^i, \boldsymbol{\delta}^o)_{y'} \neq y. \tag{7}$$

This margin captures both the relative perturbation on the input layer $\boldsymbol{\delta}^i$ and the soft-max output $\boldsymbol{\delta}^o$.

Theorem 3.2 *Suppose the parameter space Θ we optimize over has covering number that scales as $\log \mathcal{N}_{\|\cdot\|_{op}}(\eta, \Theta) \leq \lfloor \mathcal{C}^2 / \eta^2 \rfloor$ for some complexity $\mathcal{C}$. Then with probability $1 - \delta$ over the draw of the training data, any classifer $f_\theta, \theta \in \Theta$ which achieves training error zero satisfies:*

$$\mathbb{E}[f_\theta(\boldsymbol{x}) = y] \lesssim \frac{\mathcal{C} \log^2 n}{\sqrt{n}} \sqrt{\frac{1}{n} \sum_{i=1}^{n} \frac{1}{m_F(\boldsymbol{x}_i, y_i)}} + \zeta,$$

where ζ is of small order $O\left(\frac{1}{n}\log(1/\delta)\right)$.

We defer the proof to the Appendix, which is adapted from Theorem 2.1 of [33]. We observe the population risk is bounded by two key factors, the average of $\frac{1}{m_F(\boldsymbol{x}_i, y_i)}$ and $\mathcal{C}$, the covering number of the parameter space. On one side, the average of $\frac{1}{m_F(\boldsymbol{x}_i, y_i)}$ is dominated by the samples with the smallest margin. Therefore when we do adversarial training, it is important that we not only achieve higher overall accuracy, but also make sure the samples closer to the decision boundary have large enough margin. This can not be achieved by simply using constant and large ϵ that will maintain a large margin for most samples but sacrifice the accuracy of a small portion of data. On the other hand, the covering number of the network's parameter space can be roughly captured by a bound of product of all layers' weight norms. We hypothesize that with more flexibility in choosing ϵ, our algorithm will converge faster than using larger constant ϵ and will have more implicit regularization effect. To testify this hypothesis, we roughly measure the model complexity $\mathcal{C}$ by the product of the weight norms of different models. In comparison to our model, when training with constant $\epsilon = 0.01, 0.02$ and 0.03, it respectively yields $\mathcal{C}$ as large as $2.54, 3.53$ and 1.39 times of that of our model, which means our model indeed has more implicit regularization effect among others.

4 Related Work

Adversarial attack. Finding adversarial examples, also known as adversarial attacks, can be formulated as an optimization problem — the goal is to find the perturbation δ to maximize the (robust) loss, while constraining δ to have small norm (e.g., ℓ_p norm). Specifically, gradient-based algorithms have been widely used, such as fast gradient sign method (FGSM) [15], C&W attack [4] and PGD attack [20]. In addition to white-box attacks, it has been also found that adversarial attacks can be generated also in the soft-label black box setting [5, 16] and hard-label black box setting [3, 6, 7], and with similar quality to white-box attacks. Moreover, physical attacks have been proposed to generate adversarial examples in the real world [12]. Therefore, with the existence of these powerful adversarial attacks, enhancing the robustness of neural network models has become an important issue in many real world applications.

Adversarial training To enhance the adversarial robustness of a neural network model, a natural idea is to iteratively generate adversarial examples, add them back to the training data, and retrain the model. For example, Goodfellow et al. [15] use adversarial examples generated by FGSM to augment the data, and Kurakin et al. [18] propose to use a multiple-step FGSM to further improve the performance. Madry et al. [20] show that adversarial training can be formulated as a min-max optimization problem, and propose to use PGD attack (similar to multi-step FGSM) to find adversarial examples for each batch. After that, many defense algorithms are based on a similar min-max framework. However, each of them uses slightly different loss functions. We summarize the loss functions used by recent adversarial training methods in Table 2.

We see that except for natural training which directly minimizes the cross entropy loss (denoted as CE), all training techniques involve the use of the min-max framework. TRADES and MMA use the unperturbed data's cross entropy loss as an additional regularization term to achieve a better trade-off between clean and robust error.

Similar to our method, both MMA and IAAT have sample-wise adaptive ϵ during training. They also utilize the adaptive ϵ to find the largest possible ϵ_i for every sample $\boldsymbol{x}_i$. However, they do not consider the adaptive label technique mentioned in Section 3.2. As a result, they can only achieve better clean accuracy while the improvements in robust accuracy are limited. Our CAT algorithm (CAT with CE loss) is more general than IAAT and MMA. CAT reduce to IAAT when we set $c = 0$ in adaptive label smoothing. Moreover, MMA could be treated as a special case of CAT when we use a line search scheme to find the ϵ_i and $c = 0$. Also, in Section 5, we will show the importance of the adaptive label uncertainty step in CAT. Recently, a concurrent work [23] combines label smoothing with adversarial training. However, in the adversarial training process, they still use the same ϵ for all the examples, which is quite different from our instance-wise auto-tuning ϵ. Since their model jointly performs detection (dropping low confident examples) and prediction, the results and formulation are not directly comparable to other adversarial training methods.

Table 2: Summary of several robust training methods amd their corresponding loss function. Dirichlet($\mathbf{b}$) indicates the Dirichlet distribution parameterized by $\mathbf{b}$.

Methods	Loss Function
Natural	$\mathrm{CE}(f_\theta(\boldsymbol{x}), y)$
Adversarial training [20]	$\max_{\boldsymbol{x}' \in \mathcal{B}_p(\boldsymbol{x}, \epsilon)} \mathrm{CE}(f_\theta(\boldsymbol{x}'), y)$
TRADES [36]	$\mathrm{CE}(f_\theta(\boldsymbol{x}), y) + \max_{\boldsymbol{x}' \in \mathcal{B}_p(\boldsymbol{x}, \epsilon)} \mathrm{KL}(f_\theta(\boldsymbol{x}'), f_\theta(\boldsymbol{x}))$
Bilateral Adv Training [30]	$\max_{\boldsymbol{x}' \in \mathcal{B}_p(\boldsymbol{x}, \epsilon), y' \in \Delta} \mathrm{CE}(f_\theta(\boldsymbol{x}'), y')$
MMA [10]	$\mathrm{CE}(f_\theta(\boldsymbol{x}))\mathbf{1}(f_\theta(\boldsymbol{x}) \neq y) + (\max_{\boldsymbol{x}' \in \mathcal{B}_p(\boldsymbol{x}, \epsilon)} \mathrm{CE}(f_\theta(\boldsymbol{x}'), y))\mathbf{1}(f_\theta(\boldsymbol{x}) = y)$
MART [31]	$\max_{\boldsymbol{x}' \in \mathcal{B}_p(\boldsymbol{x}, \epsilon)} \mathrm{BCE}(f_\theta(\boldsymbol{x}'), y) + \mathrm{KL}(f_\theta(\boldsymbol{x}'), f_\theta(\boldsymbol{x})) \cdot (1 - f_\theta(\boldsymbol{x}))$
IAAT [2]	$\max_{\boldsymbol{x}_i' \in \mathcal{B}_p(\boldsymbol{x}_i, \epsilon_i)} \mathrm{CE}(f_\theta(\boldsymbol{x}_i'), y_i)$
CAT (ours)	$\max_{\boldsymbol{x}_i' \in \mathcal{B}_p(\boldsymbol{x}_i, \epsilon_i)} \mathrm{CE}(f_\theta(\boldsymbol{x}_i'), (1 - c\epsilon_i)y_i + c\epsilon_i \mathrm{Dirichlet}(\boldsymbol{\beta})) + \max_{j \neq y_0} [Z(\boldsymbol{x}_i')]_j - [Z(\boldsymbol{x}_i')]_{y_0}$

Table 3: The clean and robust accuracy of VGG-16 models trained by various defense methods. All robust accuracy results use $\epsilon = 8/255$ ℓ_∞ ball. $^{(X)}$ denotes using a X-step PGD attack. X random denotes X times random restart.

Methods	No attack	Deepfool	PGD^{100}	C&W^{100}	20 PGD^{1000}	20 C&W^{1000}
Natural train	**93.34%**	16.39%	0.6%	0.0%	0.0%	0.0%
Adv train [20]	80.32%	44.65%	36.36%	37.89%	36.12%	36.8%
TRADES [36]	84.85%	48.37%	38.81%	39.49%	37.95%	38.94%
CAT (ours)	85.44%	**70.19%**	**75.54%**	**51.81%**	**75.17%**	**50.08%**

Table 4: The clean and robust accuracy of Wide Resnet models trained by various defense methods. All robust accuracy results use $\epsilon = 8/255$ ℓ_∞ ball. We reported the best performance listed in the papers. $^{(*)}$ denotes random-restart is applied in the testing attack. $^{(X)}$ denotes using a X-step PGD attack. ✗ denotes not reported.

Methods	Clean accuracy	PGD accuracy	C&W accuracy
Natural training	**95.93%**	0%	0%
Adversarial training [20]	87.30%	52.68%	50.73%
Dynamic adversarial training [32]	84.51%	55.03%	51.98%
TRADES [36]	84.22%	56.40%$^{(20)}$	51.98%
Bilateral Adv Training [30]	91.00%	57.5%$^{(*20)}$	56.2%$^{(*20)}$
MMA [10]	84.36%	47.18%	✗
MART [31]	84.17%	58.56%$^{(20)}$	54.58%
IAAT [2]	91.34%	48.53%$^{(*10)}$	56.80%
CAT (ours)	89.61%	73.16%$^{(*20)}$	**71.67%**$^{(*20)}$

Other adversarial defenses In addition to adversarial training based methods, a wide range of defense methods have been proposed such as Gaussian data augmentation [35], randomized smoothing [8, 19], Mixup [37] and its variants [27, 29], and Label smoothing [14, 21]. Shafahi et al. [21] find that it could achieve similar robust accuracy with adversarial training when combining Gaussian data augmentation and label-smoothing. However, some of the aforementioned methods have been shown to cause obfuscated gradients instead of enhanced robustness [1], while adversarial training based methods are still shown to be robust under different kinds of adversarial attacks.

5 Performance Evaluation

In this section, we conduct extensive experiments to show that CAT achieves a strong result on both clean and robust accuracy. We include the following methods into our comparison:

- Customized Adversarial Training (CAT): Our proposed method.
- Adversarial training: The adversarial training method proposed in [20] where they use a K-step PGD attack as adversary.
- TRADES: TRADES [36] improves adversarial training by an additional loss on the clean examples and achieves the state-of-art performance on robust accuracy.
- Natural: the natural training which only minimizes the cross entropy loss.

Furthermore, since many recently proposed adversarial training methods have considered CIFAR-10 with Wide-ResNet structure as the standard setting for reporting their numbers, we also compare our performance with 7 previous methods on this specific setting.

Dataset and model structure. We use two popular dataset CIFAR-10 [17] and Restricted-ImageNet [9] for performance evaluation. For CIFAR-10, we use both standard VGG-16 [22] and Wide ResNet that is used in both vanilla adversarial training [20] and TRADES [36]. For VGG-16, we implement adversarial training with the standard hyper-parameters and train TRADES with the official implementation. For Wide ResNet, since the model has become standard for testing adversarial training methods, we use exactly the same model structure provided by [20, 36]. We use the models' checkpoint released by TRADES official repository and implement the Madry's adversarial training using the standard hyper-parameters. For Restricted-ImageNet, we use ResNet-50. All our experiments were implemented in Pytorch-1.4.

Implementation details. We set the number of iterations in adversarial attack to be 10 for all methods during training. Adversarial training and TRADES are trained on PGD attacks setting $\epsilon = 8/255$

with cross entropy loss (CE). All the models are trained using SGD with momentum 0.9, weight decay 5×10^{-4}. For VGG-16/Wide ResNet models, we use the initial learning rate of 0.01/0.1, and we decay the learning rate by 90% at the 80th, 140th, and 180th epoch. For CAT, we set epsilon scheduling parameter $\eta = 0.005$, $\epsilon_{max} = 8/255$ and weighting parameter $c = 10$. We set $\boldsymbol{\beta} = \mathbf{1}$ for the distribution Dirichlet($\boldsymbol{\beta}$), which is equal to a uniform distribution. Also, we set $\kappa = 10$.

White-box attacks results. For CIFAR10, we evaluate all the models under white-box $\epsilon = 8/255$ ℓ_∞-norm bounded non-targeted PGD and C&W attack. Specifically, we use both PGD^X (X-step PGD with step size $\epsilon/5$) and C&W$_\infty$. As suggested, we test our model under different steps PGD and multiple random restarts.

The experimental results are shown in Table 3, where we can easily see that CAT clearly outperforms other methods. CAT achieves a significant better robust accuracy at the standard $8/255$ perturbation threshold considered in the literature, and also have better clean accuracy. We also test the performance of CAT under attacks with 20 restarts and 1,000 iterations to confirm the robustness of the model. Futhermore, we visualize the loss landscape and perform PGD attack with different strength in the appendix.

Wide-ResNet has become a standard structure for comparing adversarial training methods, and it's standard to train and evaluate with $8/255$ ℓ_∞ norm perturbation. For this setting, we collect the reported accuracy from 7 other adversarial training methods, with several of them published very recently, to have a detailed full comparison. As shown in Table 4, our method achieves state-of-art robust accuracy while maintaining a high clean accuracy. Due to the page limit, we put the Restricted ImageNet result in the appendix.

Black-box transfer attacks results. We follow the criterion of evaluating transfer attacks as suggested by [1] to inspect whether the models trained by CAT will cause the issue of obfuscated gradients and give a false sense of model robustness. We generate 10,000 adversarial examples of CIFAR-10 from natural models with $\epsilon = 8/255$ and evaluate their attack performance on the target model. Table 5 shows that CAT achieves the best accuracy compared with adversarial training and TRADES, suggesting the effectiveness of CAT in defending both white-box and transfer attacks.

Table 5: Robust accuracy under transfer attack on CIFAR-10

Method	VGG 16	Wide ResNet
Adv train	79.13%	85.84%
TRADES	83.53%	83.90%
CAT	**86.58%**	**88.66 %**

Table 6: Ablation study on CAT by changing the loss function and removing Label Adaption (LA). All robust accuracy results use $\epsilon = 8/255$ ℓ_∞ ball.

Methods	Clean acc	PGD acc
Adv train	80.32%	36.63%
Adv+LS	80.25%	43.0%
Adp-Adv	87.91%	38.59%
CAT	**84.22%**	**75.54%**

The importance of adaptive label uncertainty. Here we discuss and perform an ablation study using VGG-16 and CIFAR-10 on the importance of adaptive label uncertainty and adaptive instance-wise ϵ. In Table 6, Adv train denotes the original adversarial training, Adv+LS denotes adversarial training with label smoothing (setting y by Eq (4)), Adp-Adv denotes adversarial training with adaptive instance-wise ϵ, and CAT is the proposed method which is a combination of these two tricks. We found that only applying adaptive instance-wise ϵ or label smoothing cannot significantly boost the robust accuracy over standard adversarial training, but the proposed method, by nicely combining these two ideas, can significantly improve the performance. This explains why CAT significantly outperforms some instance adaptive ϵ methods like IAAT and MMA.

6 Conclusions

In this paper, we propose CAT, a customized adversarial training method that is designed to have better generalization for both clean and robust performance. We also provide a theoretical analysis that explains the performance of our algorithm. Experimental results show that CAT has achieved state-of-art robust accuracy and a high clean accuracy while keeping similar running time as standard adversarial training. The success of CAT indicates that it is crucial to customize the perturbation level on both data sample side and its label in adversarial training.

Broader Impact

While more and more deep learning models are being deployed in the everyday life, people find they are highly vulnerable to adversarial examples, which could be easily generated and transferable among different models. Therefore, it becomes a huge security risk as attackers can generate adversarial examples and use them for attacking. This puts a whole host of main stream or soon to be main stream applications like facial recognition, self-driving cars, biometric recognition etc that leverage ML based computer vision models at risk. Therefore, it is crucial to develop a robust model to counter such attacks. On the other hand, understanding how the adversarial attacks and defends is helpful for the interpretability, which could also help to learn a better model in the future.

References

[1] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *International Coference on International Conference on Machine Learning*, 2018.

[2] Yogesh Balaji, Tom Goldstein, and Judy Hoffman. Instance adaptive adversarial training: Improved accuracy tradeoffs in neural nets. *arXiv preprint arXiv:1910.08051*, 2019.

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

[4] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy*, pages 39–57, 2017.

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

[6] Minhao Cheng, Thong Le, Pin-Yu Chen, Jinfeng Yi, Huan Zhang, and Cho-Jui Hsieh. Query-efficient hard-label black-box attack: An optimization-based approach. *arXiv preprint arXiv:1807.04457*, 2018.

[7] Minhao Cheng, Simranjit Singh, Patrick Chen, Pin-Yu Chen, Sijia Liu, and Cho-Jui Hsieh. Sign-opt: A query-efficient hard-label adversarial attack. In *ICLR*, 2020.

[8] Jeremy M Cohen, Elan Rosenfeld, and J Zico Kolter. Certified adversarial robustness via randomized smoothing. *International Conference on Machine Learning*, 2019.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[10] Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. Max-margin adversarial (mma) training: Direct input space margin maximization through adversarial training. *arXiv preprint arXiv:1812.02637*, 2018.

[11] Logan Engstrom, Andrew Ilyas, and Anish Athalye. Evaluating and understanding the robustness of adversarial logit pairing. *arXiv preprint arXiv:1807.10272*, 2018.

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

[13] Ruiqi Gao, Tianle Cai, Haochuan Li, Cho-Jui Hsieh, Liwei Wang, and Jason D Lee. Convergence of adversarial training in overparametrized neural networks. In *Advances in Neural Information Processing Systems*, pages 13009–13020, 2019.

[14] Morgane Goibert and Elvis Dohmatob. Adversarial robustness via adversarial label-smoothing. *arXiv preprint arXiv:1906.11567*, 2019.

[15] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *International Conference on Learning Representations*, 2015.

[16] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. *arXiv preprint arXiv:1804.08598*, 2018.

[17] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). URL `http://www.cs.toronto.edu/~kriz/cifar.html`.

[18] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *International Conference on Learning Representations*, 2017.

[19] Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. Towards robust neural networks via random self-ensemble. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 369–385, 2018.

[20] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *International Conference on Learning Representations*, 2018.

[21] Ali Shafahi, Amin Ghiasi, Furong Huang, and Tom Goldstein. Label smoothing and logit squeezing: A replacement for adversarial training? 2018.

[22] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations*, 2015.

[23] David Stutz, Matthias Hein, and Bernt Schiele. Confidence-calibrated adversarial training: Towards robust models generalizing beyond the attack used during training. *arXiv preprint arXiv:1910.06259*, 2019.

[24] Dong Su, Huan Zhang, Hongge Chen, Jinfeng Yi, Pin-Yu Chen, and Yupeng Gao. Is robustness the cost of accuracy?–a comprehensive study on the robustness of 18 deep image classification models. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 631–648, 2018.

[25] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *International Conference on Learning Representations*, 2014.

[26] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[27] Sunil Thulasidasan, Gopinath Chennupati, Jeff Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. *Neural Information Processing Systems*, 2019.

[28] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2019.

[29] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, Aaron Courville, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. *International Conference on Machine Learning*, 2018.

[30] Jianyu Wang. Bilateral adversarial training: Towards fast training of more robust models against adversarial attacks. *International Conference on Computer Vision*, 2019.

[31] Yi Bailey Ma Gu Wang, Zou. Improving adversarial robustness requires revisiting misclassified examples. *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=rklOg6EFwS`.

[32] Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, and Quanquan Gu. On the convergence and robustness of adversarial training. In *International Conference on Machine Learning*, pages 6586–6595, 2019.

[33] Colin Wei and Tengyu Ma. Improved sample complexities for deep networks and robust classification via an all-layer margin. *arXiv preprint arXiv:1910.04284*, 2019.

[34] Dong Yin, Kannan Ramchandran, and Peter Bartlett. Rademacher complexity for adversarially robust generalization. *arXiv preprint arXiv:1810.11914*, 2018.

[35] Valentina Zantedeschi, Maria-Irina Nicolae, and Ambrish Rawat. Efficient defenses against adversarial attacks. In *ACM Workshop on Artificial Intelligence and Security*, pages 39–49, 2017.

[36] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P Xing, Laurent El Ghaoui, and Michael I Jordan. Theoretically principled trade-off between robustness and accuracy. *International Conference on Machine Learning*, 2019.

[37] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *International Conference on Learning Representations*, 2018.