# Adversarial training for image classification

Amirhossein Sohrabbeig and Masooma Nazari

University of Alberta

**Abstract.** Adversarial training and its various versions boost deep network resiliency significantly, albeit at the expense of standard accuracy. Furthermore, because the training procedure is time-consuming, it is impracticable to fully investigate the trade-off between accuracy and robustness. This study tries to find an answer for this question: how can a trained model be swiftly calibrated in-situ to investigate the achievable trade-offs between its standard and robust accuracies without having to (re-)train it several times? A newly proposed method, called Once-for-all Adversarial Training (OAT), is based on a novel model-conditional training framework with a controlling hyper-parameter as the input. At testing time, the trained model may be modified "for free" between multiple standard and robust accuracies. Experiments reveal that, in various configurations, OAT achieve equivalent or even higher performance to dedicatedly trained models without any re-training or ensembling. Our codes can be found at:
https://github.com/Sohrabbeig/ECE740/tree/main/Project

## 1 Introduction

Deep neural networks (DNNs) have grown in popularity and success in a variety of machine learning tasks. They've had a lot of success in diverse recognition challenges in the realms of photos, graphs, text, and audio. They can distinguish objects with near-human accuracy in the image recognition domain[20][12]. They're also utilised for speech recognition [13], natural language processing [14], and gaming [31].

Deep learning algorithms are now used in safety-critical tasks as a result of these achievements. Deep convolutional neural networks (CNNs) are employed in autonomous cars to detect traffic signs, for example [5]. The machine learning approach utilized here must be extremely precise, steady, and dependable. But what if the CNN model fails to identify the roadside "STOP" sign and the car continues on its way? This is going to be a perilous scenario. Companies regularly utilize graph convolutional networks (GCNs)[18] in financial fraud detection systems to determine whether or not their clients are trustworthy. If fraudsters hide their personal identifying information to avoid discovery by the firm, the company will suffer a significant loss. As a result, deep neural network security has become a key problem.

Many recent studies [12][35][10] have demonstrated that DNN models are sensitive to adversarial examples, which are described as "inputs to machine

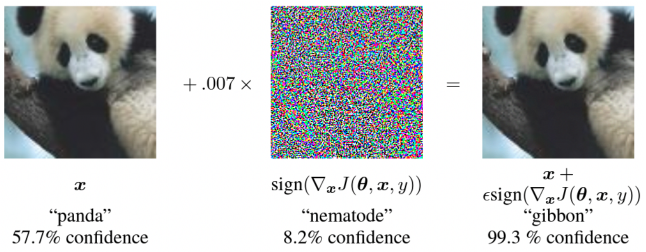
$\boldsymbol{x}$
"panda"
57.7% confidence
$+ .007 \times$
$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"nematode"
8.2% confidence
$=$
$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"gibbon"
99.3 % confidence

**Fig. 1.** "Panda" is classed as "gibbon" by introducing an imperceptible disturbance. source: Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." arXiv preprint arXiv:1412.6572 (2014).

learning models that an attacker purposely constructed to induce the model to make mistakes." In the image classification domain, adversarial examples are purposely manufactured pictures that seem very identical to the original images (see Fig.1) but can lead to incorrect prediction outputs from the classifier. Almost 80% of the digit samples may be attacked by an unnoticeable perturbation introduced to the original picture for a well-trained DNN image classifier on the MNIST dataset. Similar adversarial attacking strategies exist in different application areas employing graphs, text, or audio to confound deep learning algorithms. For example, changing only a few edges in a graph can deceive graph neural networks [46], while adding typos to a phrase can deceive text categorization or conversation systems [7]. As a result, the presence of hostile instances across all application sectors has prompted academics to advise against using DNNs in safety-critical machine learning activities.

To solve the above-mentioned problem, Adversarial training (AT) has been proposed. This approach is a powerful strategy for improving model robustness by including adversarial data in model training, and it typically outperforms other adversarial defensive strategies [45][22][4].

While adversarial defensive strategies are rising in popularity and attention in safety/security-critical applications, they do have certain drawbacks. To begin with, most adversarial defensive approaches, such as adversarial training, come at the expense of standard accuracy [8]. Many investigations [45] [29] [34] have proved both theoretically and practically that there is an inherent accuracy-robustness trade-off. In practise, most defensive mechanisms use a predetermined hyper-parameter to establish their accuracy-robustness trade-off. In adversarial training, the training goal is frequently a weighted summation of a conventional classification loss and a robustness loss, with the trade-off coefficient set to an empirical value by default. Following that, several models are trained in the same environment to compare their standard and robust accuracies.

However, in a real-world machine learning system, the criteria for standard and robust accuracies may have different thresholds that aren't always reached by "default" settings. As a result, practitioners seek simple ways to investigate and adjust the accuracy-robustness trade-off. Unfortunately, most defence mechanisms need extensive training, making it difficult or impossible to naively train a variety of configurations before selecting the best.

Motivated by the foregoing, the primary issue we pose and answer in this study is: *How can a trained model be swiftly calibrated to investigate the feasible trade-offs between its standard and robust accuracies without having to (re-)train it multiple times?*

In this study we examine a revolutionary Once-for-All Adversarial Training (OAT) [38] approach that fulfils this aim. OAT is built on top of adversarial training, but with a novel model-conditional training technique added on top. The robust loss term's weight hyper-parameter is seen by OAT as a user-specified input to the model. It samples not just data points, but also model instances from the objective family, which are parameterized by distinct loss term weights, during training. consequently, the model learns to base its behaviour and output on the hyper-parameter in question. As a result, we could switch between standard and robust accuracies "for free" in the same model at testing time by simply swapping the hyper-parameters as inputs.

## 2 Previous Works

At the moment, adversarial training is largely regarded as the most successful strategy for improving the adversarial robustness of deep learning models in practice[1]. However, adversarial training still has a long way to go before completely handling adversarial attacks. On MNIST, the current adversarial training methods can build a robust model with a worst-case accuracy of roughly 90%[22]. Adversarial training only reaches around 45 percent and 40 percent on SVHN for slightly more complex datasets, such as CIFAR-10, which is far from sufficient. In addition, adversarial training reduces the ability of deep learning models to generalize[3].

[10] is the first to mention adversarial regularisation. A regularization term based on FGSM and defined as $\mathcal{L}(\theta, x+\epsilon sign(\nabla_x \mathcal{L}(\theta, x, y))$ was added to the objective function in addition to cross-entropy loss. [21] enhanced this FGSM-based regularization term by regulating the ratio of adversarial instances in batches so that it may scale up to ImageNet. Because they think the linearity of neural networks is attributable to the availability of adversarial instances [10], their approaches are proven on single-step attacks. [27] assessed the absolute difference between the adversarial loss and its first-order Taylor expansion, indicating that more resilient models had lower local linearity values. For adversarial resilience, they replaced the FGSM-based regularization with a Local Linearity Regularization.

[45] divided the robust error $R_{rob}$ as the sum of natural error $R_{nat}$ and boundary error $R_{db}$, in contrast to earlier techniques. When the gap between data and the decision border is sufficiently tiny (less than $\varepsilon$), boundary error arises, which is also why adversarial instances exist. As a result, they presented TRADES as a method to reduce $R_{db}$ by solving the following problem:

$$\min_{f} E\{\mathcal{L}(f(x), y) + \max_{x' \in B(x,\epsilon)} \mathcal{L}(f(x), f(x'))/\lambda)\}, \tag{1}$$

where $\lambda$ is a coefficient that determines the degree of regularization. Such decomposition has been shown to be successful, with TRADES outperforming PGD-AT on CIFAR-10 and error rates lowered by 10%. The regularization term in TRADES is meant to push natural instances and their adversarial equivalents together, regardless of whether natural data is categorised properly or not. [40] looked into the impact of misclassified instances and presented Misclassification Aware adveRsarial Training (MART), which focuses on misclassified cases with weights of $1 - P_y(x, \theta)$, where $P_y(x, \theta)$ is the probability of the ground truth label $y$.

Unnoticeable noises might cause significant changes in feature space due to the amplification of deep models [10]. Some studies look at adversarial training via the lens of representation. Adversarial Logit Pairing (ALP) was introduced by [17], which encourages logits for pairs of cases to be comparable. However, ALP is initially ineffective due to incorrect adversarial training objectives [8]. [24] used the popular triplet loss for regularization, which employs adversarial instances as anchors, to improve the alignment of natural data representations and their adversarial counterparts.

Adversarial regularization is one of the most important types of adversarial training[30]. Adversarial regularization is more flexible than the original formulation of adversarial training, and it necessitates a thorough grasp of adversarial robustness. Furthermore, the breakdown of robust error allows unlabeled data to improve adversarial robustness.

## 3 Preliminaries

A standard classifier (e.g., a DNN) $f : \mathbb{R}^d \rightarrow \mathbb{R}^c$ with parameter $\theta$ transfers an input picture to classification probabilities learnt by empirical risk minimization (ERM), given the data distribution $D$ over images $x \in \mathbb{R}^d$ and their labels $y \in \mathbb{R}^c$:

$$\min_{\theta} \mathbb{E}_{(x,y)\sim D}\ \mathcal{L}(f(x; \theta), y),$$

where $\mathcal{L}(.,.)$ is the cross-entropy loss by default: $\mathcal{L}(f(x; \theta), y) = y^T log(f(x; \theta))$.

### 3.1 Adversarial training

Many approaches for improving DNN adversarial resilience have been developed, with adversarial training (AT) based methods [22] being among the most effective. The majority of current AT algorithms [45] [22] [32] aim to maximise a hybrid loss that combines a conventional classification loss and a robustness loss term:

$$\min_{\theta} \mathbb{E}_{(x,y)\sim D}\ [(1-\lambda)\mathcal{L}_c + \lambda\mathcal{L}_a], \tag{2}$$

where $\mathcal{L}_c$ represents the loss in classification over standard (or clean) pictures, and $\mathcal{L}_c$ denotes the loss in robustness against adversarial cases. $\lambda$ is a hyper-parameter with a set weight. For example, in PGD-AT [22] and its variations [37], a common type of AT:

$$\mathcal{L}_c = \mathcal{L}(f(x;\theta), y),\ \ \mathcal{L}_a = \max_{\delta\in\mathcal{B}(\epsilon)} \mathcal{L}(f(x+\delta;\theta), y), \tag{3}$$

where $\mathcal{B}(\epsilon) = \{\delta \mid \|\delta\|_\infty \leq \epsilon\}$ denotes the permitted perturbation set. TRADES [45] employ the same $\mathcal{L}_c$ as PGD-AT, but substitute $\mathcal{L}_a$ with a soft logits-pairing term instead of cross-entropy. $\mathcal{L}_a$ is to maximise the margins between successfully categorized photos in MMA training [6].

### 3.2 Conditional learning and inference

For additional controllablity, several domains have investigated the notion of conditioning a trained model's inference on each testing input. The dynamic inference literature, whose basic notion is to dynamically alter the computational route per input, is a good place to start. [16][36], for example, add numerous side branches to DNN so that early predictions may be routed via them. [9][39] permitted an input to choose whether or not to travel through each layer. The slimmable network [44], which is related to our work, seeks to train a DNN with configurable channel width during runtime. Batch normalisation (BN) was superseded by switchable BN (S-BN), which uses separate BNs for each different-width subnetwork. The average loss throughout all different-width subnetworks is then optimized to train the new network. Furthermore, many generative models [25] may generate outputs based on input class labels.

## 4 Approach

### 4.1 Main Idea

The purpose of OAT is to adopt a distribution of DNNs $f(.,\lambda;\theta) \sim F$, conditioned on $\lambda \sim P_\lambda$, so that various DNNs chosen from this learnt distribution can have varying accuracy-robustness trade-offs depending on the input$\lambda$.

While traditional DNN training samples data, OAT recommends sampling one f$f(.,\lambda;\theta) \sim F$ per data. Each time, we condition $f$ on $\lambda$: specifically, it will accept a hyperparameter $\lambda \sim P_\lambda$ as an input and adjust the current AT loss function using this same hyperparameter:

$$\mathcal{L}(x, y, \lambda) = E_{(x,y)\sim D, \lambda\sim P_\lambda}[(1-\lambda)\mathcal{L}_c + \lambda\mathcal{L}_a]. \tag{4}$$

The loss characterized by the current $\lambda$ generates the gradient with respect to each $(x, y)$. As a result, OAT optimizes a dynamic loss function with variable parameters every sample and forces weight sharing between iterations.

**Algorithm 1** OAT Algorithm Outline

```
Input: Training set $\mathcal{D}$, model $f$, $P_\lambda$, maximal steps $T$
Ouput: Model Parameter $\theta$
  for $t = 1$ to $T$ do
    Sample a minibatch of $(x, y)$ from $\mathcal{D}$;
    Sample a minibatch of $\lambda$ from $P_\lambda$;
    Generate $x_{adv}$(Using PGD);
    Update network parameter $\theta$ by Eq. (4)

  end for
```

Model-conditional training is what we call it. We employ cross-entropy loss for $\mathcal{L}_c$ and $\mathcal{L}_a$, as PGD-AT did in Eq (3), without losing generality: $\mathcal{L}_c = \mathcal{L}(f(x;\theta), y)$, $\mathcal{L}_a = \max_{\delta \in \mathcal{B}(\epsilon)} \mathcal{L}(f(x+\delta;\theta), y)$. Other kinds of $\mathcal{L}_c$ and $\mathcal{L}_a$, such as those used in TRADES and MMA training, are also compatible with OAT. The OAT algorithm is described in detail in Algo. 1.

Although the two serve fundamentally distinct functions, our model sampling concept was inspired by and may be inextricably tied to dropout [33]. Dropout is applied to a DNN by sampling a sub-network from the main network at each iteration, which comprises of all units that have survived dropout. As a result, training a DNN with dropout may be thought of as training a large number of sub-networks with considerable weight sharing, where each thinning network is infrequently (or seldom) trained. The trained DNN then acts as an ensemble of those subnetworks, which improves its generalization. OAT samples a model per-data as well, however each sampled model is characterized by a distinct loss function rather than a new architecture. Furthermore, we may view the OAT trained model as an ensemble of those sampled models; nevertheless, unlike the static "model averaging" interpretation in dropout, the model output at run time is conditioned on the given input, which could be interpreted as "model selection."

We embed $\lambda$ from a scalar to a high-dimensional vector in $\mathbb{R}^d$ to encode and condition on it. While one-hot encoding is a basic approach, we encode various s using (nearly)-orthogonal random vectors for improved scalability and experimentally higher performance.

To accomplish our conditioning of $f$ on $\lambda$, we use the Feature-wise Linear Modulation (FiLM) module [26]. If the output feature map of a layer is $h \in \mathbb{R}^{C\times H\times W}$, FiLM applies a channel-wise affine transformation on it, with the parameters $\gamma \in \mathcal{R}^c$ and $\beta \in \mathcal{R}^c$ being dependent on the input $\lambda$:

$$FiLM(h_c; \gamma_c, \beta_c) = \gamma_c h_c + \beta_c, \gamma = g_1(\lambda), \beta = g_2(\lambda),$$

$g_1$ and $g_2$ are two multi-layer perceptrons (MLPs) with Leaky ReLU, and the subscripts correspond to the $c^{th}$ feature map of $h$ and the $c^{th}$ element of, respectively. FiLM's output is subsequently sent on to the network's subsequent tiers. In practise, the affine transformation is applied after each batch normalisation (BN) layer, and each MLP has two C-dimensional layers.

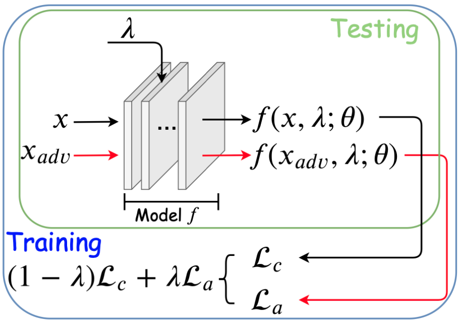

**Fig. 2.** Illustration of the OAT structure in its whole. The input and loss function hyperparameters are both set to the hyperparameter $\lambda$. It can be defined during testing and is varied in training. source: Wang, Haotao, et al. "Once-for-all adversarial training: In-situ tradeoff between robustness and accuracy for free." Advances in Neural Information Processing Systems 33 (2020): 7449-7461.

### 4.2 Overcoming a unique bottleneck: standard and adversarial feature statistics

After putting the prototype OAT framework in place in the preceding section, an interesting finding was made: while altering $\lambda$ can result in varied standard and robust accuracies, the numbers obtained are drastically deteriorated when compared to dedicatedly trained models with fixed $\lambda$. Further analysis suggests that the "conflict" resulting from combining standard and adversarial characteristics in one model appears to be the source of the bottleneck, and that a BN split might effectively resolve it.

To demonstrate this contradiction, we train ResNet34 on the CIFAR-10 dataset using PGD-AT, with the $\lambda$ ranging from 0 (normal training) to 1 (common setting for PGD-AT). The statistics of the final BN layer, namely the running mean and running variance, are then visualised as shown in Fig. 2. While gradually shifting causes continuous feature statistics transitions/shifts, we see that the feature statistics of $\lambda = 0$ (black dots) are unusually isolated from those of other nonzero $\lambda$s; the distance is substantial even between $\lambda = 0$ and 0.1. Meanwhile, all nonzeros result in feature statistics that are mostly overlapping. This is also consistent with our findings, which reveal that the preliminary OAT tends to fit either only clean pictures (resulting in lower robust accuracy) or only adversarial images (resulting in lower robust accuracy) (resulting in degraded standard accuracy). As a result, we hypothesise that the gap between standard ($\lambda = 0$) and adversarial ($\lambda \neq 0$) feature statistics accounts for the difficulty in combining their learning.

Despite the fact that the problem setting and purpose were considerably different from ours [42] [41], the authors observed similar differences between conventional and adversarial feature statistics. They devised a technique called dual batch normalization (dual BN), which separates the standard and adversarial feature statistics into two separate BNs. This has been proven to boost

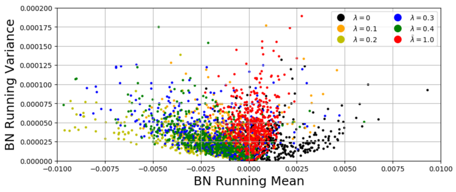

**Fig. 3.** On CIFAR10, the running mean (x-axis) and variance (y-axis) of the last BN layer from ResNet34 trained with PGD-AT and fixed lambda values. source: Wang, Haotao, et al. "Once-for-all adversarial training: In-situ tradeoff between robustness and accuracy for free." Advances in Neural Information Processing Systems 33 (2020): 7449-7461.

picture identification accuracy, whereas adversarial examples operate as a particular data augmentation in their application environment.

We're going to make dual $BN$ for our situation. In the network, we replace all $BN$ layers with dual $BN$ layers. A dual BN is made up of two separate $BNs$, $BN_c$ and $BN_a$, which account for standard and adversarial properties, respectively. After that, a switch layer is used to choose which of the two $BNs$ should be active for the current sample. In [42] [41], the authors exclusively deal with clean instances (i.e., $\lambda = 0$) and adversarial examples (i.e., $\lambda = 1$) generated using "common" PGD-AT. They also solely seek to improve standard accuracy throughout testing. As a result, their teaching and testing of the switching policy is simple.

For both training and testing, OAT must use a range of adversarial instances constructed with all $\lambda$s between [0,1]. We route $\lambda = 0$ cases through $BN_c$ at both training and testing time, based on our data from Fig. 2, whereas all $\lambda = 0$ cases go to $BN_a$. The user specifies the value of $\lambda$ during testing, based on his or her selection between standard and robust accuracy: a greater $\lambda$ focuses robustness more, whereas a lower $\lambda$ favours accuracy. As we've seen in our trials, a modified dual $BN$ is a critical component of OAT's success.

## 4.3 Extending from OAT to OATS

The growing popularity of deploying DNNs in resource-constrained devices such mobile phones, IoT cameras, and outdoor robots has increased demand for DNN performance as well as efficiency. Recent work [43] [11] has shown that compressing models or reducing inference latency while maintaining accuracy and robustness is possible, leading to the enticing aim of "co-designing" a model to be accurate, trustworthy, and resource-efficient. The addition of the additional efficiency component, on the other hand, substantially complicates the design space, making it even more difficult to manually analyze the various model capabilities and training procedures.

Once-for-all Adversarial Training and Slimming(OATS) is a novel framework based on the OAT approach. The key concept is to use another model compact hyper-parameter to condition the model-conditional training. Models trained by

**Algorithm 2** OATS Algorithm Outline

**Input**: Training set $\mathcal{D}$, model $f$, $P_\Lambda$, maximal steps $T$, a list of pre-defined width factors
**Output**: Model Parameter $\theta$

```
for $t = 1$ to $T$ do
    Sample a minibatch of $(x, y)$ from $\mathcal{D}$;
    Sample a minibatch of $\lambda$s from $P_\Lambda$;
    Clear gradients: $optimizer.zero_grad()$;
    for width factor in width factor do
        switch the S-BN to current width factor on network $f$ and extract corresponding sub-network;
        Generate $x_{adv}$(Using PGD);
        Compute loss in Eq. (4): $loss = \mathcal{L}(x, y, \lambda)$
        Accumulate gradients: loss.backward();
    end for
    Update network parameter $\theta$ by Eq. (4): optimizer.step();
end for
```

OATS can concurrently explore the spectrum along the accuracy, robustness, and model efficiency dimensions by altering the two hyper-parameter inputs (loss weight $\lambda$, and model compactness), providing the three's trade-off.

We implement the model compactness level parameter as the channel width factor as defined in [44], which is inspired by the state-of-the-art channel pruning method, the slimmable network [44]. Each width component represents a subnetwork within the larger network. The sub-network with a width factor of 0.5, for example, only owns the (first) half of the channels in each layer. To fulfil resource limitations ranging from loose to strict, we might pre-define a set of high-to-low acceptable widths. The overall procedure for OATS is described in Algo. 2, with the exception that in OAT, each BN (either $BN_c$ or $BN_a$) is replaced with a switchable batch normalisation (S-BN) [44], resulting in separate BNs for different-width sub-networks. As a result, each subnetwork's BNc and BNa are unique. Assuming three distinct widths are employed, each BN in the original core will become two S-BNs in OATS, resulting in a total of six BNs. We minimize the average loss in Eq. (4) of all varied width sub-networks to train the network.

## 5 Evaluation

### 5.1 Datasets and models

We evaluated the proposed method on ResNet34[12] using CIFAR-10[19]. A ResNet-34 is trained for image classification. In adversarial training, we set perturbation $\epsilon = 0.031$, perturbation step size $\eta_1 = 0.007$, number of iterations $K = 10$, learning rate $\eta_2 = 0.1$, batch size $m = 128$, and run 100 epochs on the training dataset. To evaluate the robust error, we applied PGD (white-box) attack with 20 iterations and the step size is 0.003.

## 5.2 Evaluation Metrics

Standard Accuracy (SA) refers to the accuracy of classification on the original clean test set. The (default) accuracy is denoted by SA. Robust Accuracy (RA) is the accuracy with which adversarial pictures produced from the original test set are classified. The model's robustness is measured by RA. We develop the SA-RA frontier, an empirical Pareto frontier between a model's attainable accuracy and resilience, by measuring the SAs and RAs of the models dedicatedly trained by PGD-AT with varied (fixed) $\lambda$ values, to more directly evaluate the trade-off between SA and RA. We might also change the input $\lambda$ in the OAT-trained models, and the resulting SA-RA trade-off curve should ideally be as near to the SA-RA frontier as feasible.

## 5.3 Sampling $\lambda$

Unless otherwise indicated, during training, $\lambda$s are uniformly sampled in an element-wise manner from the set $S1 = \{0, 0.1, 0.2, 0.3, 0.4, 1\}$, i.e., all $\lambda$s in a training batch are i.i.d. sampled from $S1$.

## 5.4 Results

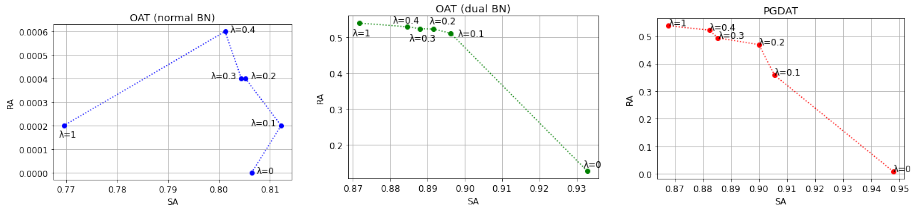

**Fig. 4.** OAT: SA-RA Trade-off of different methods on CIFAR10 dataset. $\lambda$ varies from the largest value to the smallest value in $S1$ for the points from top-left to bottom-right on each curve.

As demonstrated in Fig. 4, OAT (dual BN) model achieves a wide range of accuracy and robustness tradeoffs that are extremely near to or even above the SA-RA frontier (red curves in Fig. 4). For example, a single OAT (dual BN) model may be gradually modified from the most robust state (87.18% SA and 53.84% RA) to the most accurate state (93.28% SA and 12.7% RA) by simply changing the value of input at testing time.

One very important observation was that using Titan RTX GPUs with 4608 CUDA cores and 24 GB of RAM, and eight CPU cores, training with PGDAT for each lambda configuration took about four and a half hours. Consequently, the whole training process on the S1 set with PGDAT takes about 27 hours. However, the whole process of training OAT-BN took about six hours using the same hardware configuration, which shows a great decrease in finding different trades-off between standard and robust accuracies.

As demonstrated in Fig. 4, OAT (dual BN) produces a significantly superior SA-RA trade-off than OAT (regular BN), showing its value in model-conditional adversarial learning. One intriguing finding is that OAT (normal BN) models may readily collapse to only fit clean or adversarial pictures, which is consistent with our findings in Fig. 3 about the difference between standard and robust feature statistics.

Validation accuracy curves for different $\lambda$ values can be find in A.

## 6 Discussion

Deep neural networks are known for being sensitive to adversarial attacks [35]. With the expanding use of deep neural networks (DNNs) in security-sensitive applications like self-driving cars [2] and biometrics [15], a pressing worry has arisen over the model's resilience against adversarial attacks, additional to its average accuracy on standard inputs. We propose to tackle the new difficult challenge of fast calibrating a trained model in-place in order to investigate the feasible trade-offs between its standard and robust accuracies without (re-)training it multiple times in this study. The issues we frequently encounter in our real-world self-driving applications drive our suggested method: how to update an autonomous agent in-place to fulfil the standard/robust accuracy criteria that vary across contexts and time. When an autonomous agent is put in a less confident or unfavourable environment, we could anticipate it to perceive and behave more carefully (i.e., to prioritize enhancing its robustness). In addition, our technique offers a unique way to effectively traverse the whole accuracy-robustness spectrum, allowing for a more thorough and fair comparison of model behaviours under various trade-off hyper-parameters without having to retrain. Many high-stakes real-world applications, such as self-driving [2], biometrics [15], medical image analysis [28], and computer-aided diagnostics [23], can benefit from our suggested methodologies.

## References

1. Athalye, A., Carlini, N.: On the robustness of the cvpr 2018 white-box adversarial example defenses. arXiv preprint arXiv:1804.03286 (2018)
2. Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L.D., Monfort, M., Muller, U., Zhang, J., et al.: End to end learning for self-driving cars. arXiv preprint arXiv:1604.07316 (2016)
3. Buckman, J., Roy, A., Raffel, C., Goodfellow, I.: Thermometer encoding: One hot way to resist adversarial examples. In: International Conference on Learning Representations (2018)

4. Chen, T., Liu, S., Chang, S., Cheng, Y., Amini, L., Wang, Z.: Adversarial robustness: From self-supervised pre-training to fine-tuning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 699–708 (2020)
5. CireAan, D., Meier, U., Masci, J., Schmidhuber, J.: Multi-column deep neural network for traffic sign classification. Neural networks **32**, 333–338 (2012)
6. Ding, G.W., Sharma, Y., Lui, K.Y.C., Huang, R.: Mma training: Direct input space margin maximization through adversarial training. arXiv preprint arXiv:1812.02637 (2018)
7. Ebrahimi, J., Rao, A., Lowd, D., Dou, D.: Hotflip: White-box adversarial examples for text classification. arXiv preprint arXiv:1712.06751 (2017)
8. Engstrom, L., Ilyas, A., Athalye, A.: Evaluating and understanding the robustness of adversarial logit pairing. arXiv preprint arXiv:1807.10272 (2018)
9. Figurnov, M., Collins, M.D., Zhu, Y., Zhang, L., Huang, J., Vetrov, D., Salakhutdinov, R.: Spatially adaptive computation time for residual networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1039–1048 (2017)
10. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014)
11. Gui, S., Wang, H., Yang, H., Yu, C., Wang, Z., Liu, J.: Model compression with adversarial robustness: A unified optimization framework. Advances in Neural Information Processing Systems **32** (2019)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
13. Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A.r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N., et al.: Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Signal processing magazine **29**(6), 82–97 (2012)
14. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
15. Hu, G., Yang, Y., Yi, D., Kittler, J., Christmas, W., Li, S.Z., Hospedales, T.: When face recognition meets with deep learning: an evaluation of convolutional neural networks for face recognition. In: Proceedings of the IEEE international conference on computer vision workshops. pp. 142–150 (2015)
16. Huang, G., Chen, D., Li, T., Wu, F., Van Der Maaten, L., Weinberger, K.Q.: Multi-scale dense networks for resource efficient image classification. arXiv preprint arXiv:1703.09844 (2017)
17. Kannan, H., Kurakin, A., Goodfellow, I.: Adversarial logit pairing. arXiv preprint arXiv:1803.06373 (2018)
18. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
19. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
20. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems **25** (2012)
21. Kurakin, A., Goodfellow, I., Bengio, S.: Adversarial machine learning at scale. arXiv preprint arXiv:1611.01236 (2016)
22. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083 (2017)

23. Mansoor, A., Bagci, U., Foster, B., Xu, Z., Papadakis, G.Z., Folio, L.R., Udupa, J.K., Mollura, D.J.: Segmentation and image analysis of abnormal lungs at ct: current approaches, challenges, and future trends. Radiographics **35**(4), 1056–1076 (2015)
24. Mao, C., Zhong, Z., Yang, J., Vondrick, C., Ray, B.: Metric learning for adversarial robustness. Advances in Neural Information Processing Systems **32** (2019)
25. Mirza, M., Osindero, S.: Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014)
26. Perez, E., Strub, F., De Vries, H., Dumoulin, V., Courville, A.: Film: Visual reasoning with a general conditioning layer. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 32 (2018)
27. Qin, C., Martens, J., Gowal, S., Krishnan, D., Dvijotham, K., Fawzi, A., De, S., Stanforth, R., Kohli, P.: Adversarial robustness through local linearization. Advances in Neural Information Processing Systems **32** (2019)
28. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015)
29. Schmidt, L., Santurkar, S., Tsipras, D., Talwar, K., Madry, A.: Adversarially robust generalization requires more data. Advances in neural information processing systems **31** (2018)
30. Shaham, U., Yamada, Y., Negahban, S.: Understanding adversarial training: Increasing local stability of supervised models through robust optimization. Neurocomputing **307**, 195–204 (2018)
31. Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al.: Mastering the game of go with deep neural networks and tree search. nature **529**(7587), 484–489 (2016)
32. Sinha, A., Namkoong, H., Volpi, R., Duchi, J.: Certifying some distributional robustness with principled adversarial training. arXiv preprint arXiv:1710.10571 (2017)
33. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research **15**(1), 1929–1958 (2014)
34. Sun, K., Zhu, Z., Lin, Z.: Towards understanding adversarial examples systematically: Exploring data size, task and model factors. arXiv preprint arXiv:1902.11019 (2019)
35. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 (2013)
36. Teerapittayanon, S., McDanel, B., Kung, H.T.: Branchynet: Fast inference via early exiting from deep neural networks. In: 2016 23rd International Conference on Pattern Recognition (ICPR). pp. 2464–2469. IEEE (2016)
37. Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., Madry, A.: Robustness may be at odds with accuracy. arXiv preprint arXiv:1805.12152 (2018)
38. Wang, H., Chen, T., Gui, S., Hu, T., Liu, J., Wang, Z.: Once-for-all adversarial training: In-situ tradeoff between robustness and accuracy for free. Advances in Neural Information Processing Systems **33**, 7449–7461 (2020)
39. Wang, X., Yu, F., Dou, Z.Y., Darrell, T., Gonzalez, J.E.: Skipnet: Learning dynamic routing in convolutional networks. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 409–424 (2018)

40. Wang, Y., Zou, D., Yi, J., Bailey, J., Ma, X., Gu, Q.: Improving adversarial robustness requires revisiting misclassified examples. In: International Conference on Learning Representations (2019)
41. Xie, C., Tan, M., Gong, B., Wang, J., Yuille, A.L., Le, Q.V.: Adversarial examples improve image recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 819–828 (2020)
42. Xie, C., Yuille, A.: Intriguing properties of adversarial training at scale. arXiv preprint arXiv:1906.03787 (2019)
43. Ye, S., Xu, K., Liu, S., Cheng, H., Lambrechts, J.H., Zhang, H., Zhou, A., Ma, K., Wang, Y., Lin, X.: Adversarial robustness vs. model compression, or both? In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 111–120 (2019)
44. Yu, J., Yang, L., Xu, N., Yang, J., Huang, T.: Slimmable neural networks. arXiv preprint arXiv:1812.08928 (2018)
45. Zhang, H., Yu, Y., Jiao, J., Xing, E., El Ghaoui, L., Jordan, M.: Theoretically principled trade-off between robustness and accuracy. In: International conference on machine learning. pp. 7472–7482. PMLR (2019)
46. Zügner, D., Akbarnejad, A., Günnemann, S.: Adversarial attacks on neural networks for graph data. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 2847–2856 (2018)

## A Appendix

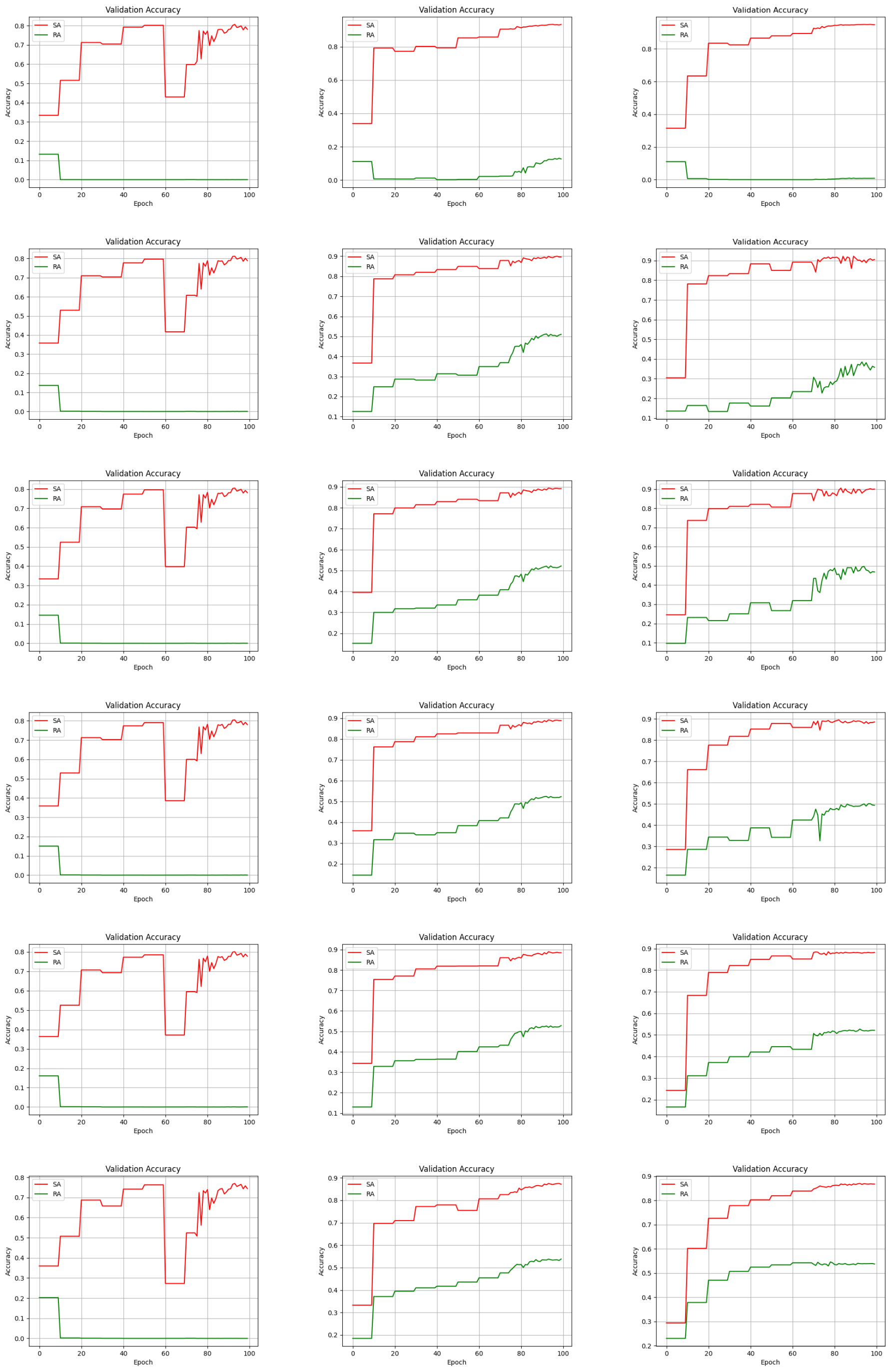

**Fig. 5.** Left: OAT(normal BN), Middle: OAT(dual BN), Right: PGDAT. row1: $\lambda = 0$, row2: $\lambda = 0.1$, row3: $\lambda = 0.2$, row4: $\lambda = 0.3$, row5: $\lambda = 0.4$, row6: $\lambda = 1$,