TWINS: A Fine-Tuning Framework for Improved Transferability of Adversarial Robustness and Generalization

Ziquan Liu[1], Yi Xu[2]*, Xiangyang Ji[3] and Antoni B. Chan[1]*
[1]Department of Computer Science, City University of Hong Kong
[2]School of Artificial Intelligence, Dalian University of Technology
[3]Department of Automation, Tsinghua University
ziquanliu.cs@gmail.com, yxu@dlut.edu.cn, xyji@tsinghua.edu.cn, abchan@cityu.edu.hk

Abstract

*Recent years have seen the ever-increasing importance of pre-trained models and their downstream training in deep learning research and applications. At the same time, the defense for adversarial examples has been mainly investigated in the context of training from random initialization on simple classification tasks. To better exploit the potential of pre-trained models in adversarial robustness, this paper focuses on the fine-tuning of an adversarially pre-trained model in various classification tasks. Existing research has shown that since the robust pre-trained model has already learned a robust feature extractor, the crucial question is how to maintain the robustness in the pre-trained model when learning the downstream task. We study the model-based and data-based approaches for this goal and find that the two common approaches cannot achieve the objective of improving both generalization and adversarial robustness. Thus, we propose a novel statistics-based approach, **T**wo-**WI**ng **N**ormli**S**ation (TWINS) fine-tuning framework, which consists of two neural networks where one of them keeps the population means and variances of pre-training data in the batch normalization layers. Besides the robust information transfer, TWINS increases the effective learning rate without hurting the training stability since the relationship between a weight norm and its gradient norm in standard batch normalization layer is broken, resulting in a faster escape from the sub-optimal initialization and alleviating the robust overfitting. Finally, TWINS is shown to be effective on a wide range of image classification datasets in terms of both generalization and robustness.*

1. Introduction

The adversarial vulnerability of deep neural networks (DNNs) [60] is one of the major obstacles for their wide applications in safety-critical scenarios such as self-driving cars [19] and medical diagnosis [20]. Thus, addressing this issue has been one focus of deep learning research in the past eight years. Existing works have proposed to improve adversarial robustness from different perspectives, including data augmentation [22, 49, 54, 57], regularization [38, 44, 45, 52] and neural architecture [24, 29]. However, most of existing works investigate the problem under the assumption that the training data is sufficient enough, and training from scratch gives a satisfactory performance, which is not realistic in the real world. There are a large number of computer vision tasks where training from scratch is inferior to training from pre-trained weights, such as fine-grained image classification (e.g., Caltech-

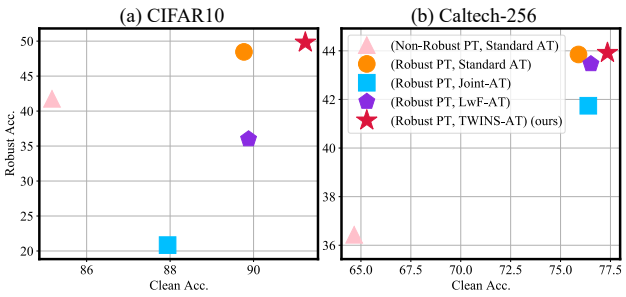

Figure 1. The performance of fine-tuning robust and non-robust large-scale pre-trained (PT) ResNet50 [27, 56] on CIFAR10 [35] and Caltech-256 [23]. We compare standard adversarial training (AT), Learning without Forgetting (LwF) (*model approach*) [40], joint fine-tuning with UOT data selection (*data approach*) [47] and our TWINS fine-tuning. The robust accuracy is evaluated using l_∞ norm bounded AutoAttack [12] with $\epsilon = 8/255$. On CIFAR10, the data-based and model-based approach fail to improve clean and robust accuracy. On Caltech, both approaches improve the clean accuracy but hurt the robust accuracy. Our TWINS fine-tuning improves the clean and robust performance on both datasets. The pink triangle denotes the performance of standard AT with the non-robust pre-trained ResNet50, which drops considerably compared with fine-tuning starting from the robust pre-trained model.

*Corresponding author

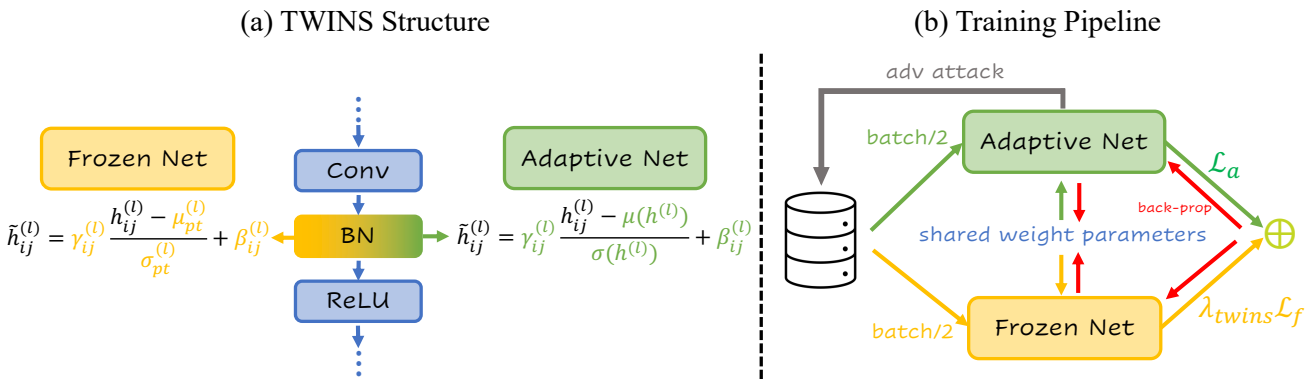

Figure 2. The TWINS structure and training pipeline. **(a)** The Frozen Net and Adaptive Net have the same structure and share the weight parameters, except for batch normalization (BN) layers. The Frozen Net uses pre-trained means and standard deviations (STD) in the normalization layer, while Adaptive Net uses the mean and STD computed from the current batch as in standard BN. **(b)** In each step of mini-batch stochastic gradient descent (SGD), we split the batch of adversarial examples, generated from attacking the Adaptive Net, into two sub-batches and feed them to the Adaptive Net and Frozen Net respectively. The loss of two networks are combined and back-propagated to their shared parameters to train the network. In the *inference stage*, only the Adaptive Net is used.

UCSD Birds-200-2011 or CUB200 [61]), object detection [43] and semantic segmentation [50].

On the other hand, pre-trained models have been considered as the foundation models in deep learning [5] as a result of their strong performance and wide employment in computer vision [18, 25, 26, 46], as well as natural language processing [6, 14, 53]. Thus, how to better use the pre-trained model in downstream has emerged as a major research topic in many vision and language tasks, such as image classification under distribution shifts [48, 64], object detection [37] and semantic segmentation [30, 36]. There are a few papers that investigate the pre-trained model's robustness in target tasks [7, 8, 16, 32, 33, 58, 63]. [7, 58] mainly considers the transfer between small-scale datasets (e.g., CIFAR100 to CIFAR10), while [8, 33] use adversarial robust pre-training and fine-tuning on the same dataset, without considering a large-scale and general pre-trained model. Finally, [16, 63] investigate different kinds of robustness to corruption or out-of-distribution samples, and are not devoted to adversarial robustness.

In this paper, we consider how to transfer the adversarial robustness of a large-scale robust pre-trained model (e.g., a ResNet50 pre-trained on ImageNet [13] with adversarial training) on various downstream classification tasks when fine-tuning with adversarial training. This problem setting is becoming more important as the standard pre-trained models do not learn robust representations from the pre-training data and are substantially weaker than the robust pre-trained counterparts in some challenging downstream tasks, e.g., fine-grained classification as shown in our experiment. Meanwhile, more large-scale robust pre-trained models are released (e.g., ResNet [56] and ViT [4]), which makes the robust pre-trained models more accessible. However, naively applying adversarial training to fine-tune from the robustly pre-trained model will lead to suboptimal robustness, since the robust representations learned by the robust pre-trained model are not fully utilized. For example, [58] suggests that the robustness from a pre-trained model needs to be explicitly maintained for its better transfer to the downstream.

Following the idea that the key to improving the transferability of robustness is to maintain the robustness of the pre-training stage during fine-tuning [58], we first evaluate the data-based and model-based approach on two representative datasets, CIFAR10 and Caltech-256. The data-based approach uses pre-trained data in the fine-tuning and keeps their performance under adversarial attack, while the model-based approach regularizes the distance of features of the fine-tuned and pre-trained model. Our experiment shows that both methods fail to improve the robustness and generalization (Fig. 1), since the two methods are too aggressive in retaining the robustness and hurt the learning in downstream. Thus, we propose a subtle approach that keeps the batch-norm (BN) statistics of pre-training for preserving the robustness, which we call **T**wo-**WI**ng **N**ormali**S**ation (TWINS) fine-tuning. TWINS has two neural networks with fixed and adaptive BN layers respectively, where the fixed BN layers use the population means and STDs of pre-training for normalization, while the adaptive BN layers use the standard BN normalization. Our experiment first demonstrates the importance of pre-trained BN statistics in the robust fine-tuning and then finds the benefit of TWINS in adversarial training dynamics. As the relationship between weight norm and its gradient norm no longer holds in TWINS, it is able to increase the gradient magnitude without increasing the gradient variance. At the initial training stage, TWINS has a faster escaping speed from the sub-optimal initialization than vanilla adversarial training [42]. At the final training stage, the gradient of TWINS is more stable than adversarial training, which alleviates the robust

overfitting effect [57]. In summary, the contributions of our paper are as follows:

1. We focus on the fine-tuning of *large-scale* robust pre-trained models as a result of their potential importance in various downstream tasks. We evaluate current approaches to retain the pre-training robustness in fine-tuning, and show that they cannot substantially improve the robustness.
2. We propose TWINS, a statistics-based approach for better transferability of robustness and generalization from the pre-training domain to the target domain. TWINS has two benefits: a) it keeps the robust statistics for downstream tasks, thus helps the transfer the robustness to downstream tasks and b) it enlarges the gradient magnitude without increasing gradient variance, thus helps the model escape from the initialization faster and mitigates robust overfitting. The mechanisms of these two benefits are validated by our empirical study.
3. The effectiveness of TWINS is corroborated on five downstream datasets by comparing with two popular adversarial training baselines, adversarial training (AT) [49] and TRADES [65]. On average, TWINS improves the clean and robust accuracy by 2.18% and 1.21% compared with AT, and by 1.46% and 0.69% compared with TRADES. The experiment shows the strong potential of robust pre-trained models in boosting downstream's robustness and generalization when using more effective fine-tuning methods.

2. Related Work

Adversarial defense. There are several major approaches to improving the adversarial robustness of DNNs. The training of DNNs can be regularized to induce biases that are beneficial to adversarial robustness, such as locally linear regularization [52], margin maximization [15, 44] and Jacobian regularization [31]. The most commonly used adversarial defense is adversarial training (AT) [49], which directly trains the DNN on adversarial examples generated from PGD attack. Later, TRADES [65] is proposed to add a KL regularization to AT and achieves stronger adversarial robustness. Our paper proposes TWINS to improve adversarial training in the fine-tuning stage when the initial model is adversarially pre-trained. We compare TWINS-AT and TWINS-TRADES with vanilla AT and TRADES in our experiment and show the strong effectiveness of TWINS in the robust fine-tuning setting.

Fine-tuning for downstream robustness. Several aspects of robustness in pre-training and fine-tuning have been studied in existing works. Adversarial contrastive learning [8, 33] is proposed to pre-train on a dataset with contrastive learning and then fine-tune on the *same* dataset, without considering the transferability of robustness from a large-scale pre-trained model to a *different* downstream task. In contrast, our paper investigates a more general problem, where task-specific pre-training is not needed for a new task as we use one robust large-scale pre-trained model trained on ImageNet. [56] considers the robust pre-training on the large-scale ImageNet and its transfer to downstream tasks, but focuses on the performance on clean instead of adversarial images. The Learning-without-Forgetting (LwF) [40] approach for retaining robustness is shown to be effective in the *small-scale* transfer experiment [58], but is not effective in our experiment setting of transfer of large-scale models. [32] proposes a learning rate schedule to improve the adversarial robustness of fine-tuned models, and [17] proposes robust informative fine-tuning for pre-trained language models to robustly keep pre-training information in downstream. The difference between [17, 32] and our work is that they assume a standard pre-trained model instead of the adversarial pre-trained model. [16, 63] investigate the performance of pre-trained models in downstream tasks, but the focus is the robustness to out-of-distribution samples instead of adversarial perturbations.

Batch normalization. There are existing papers proposing the two-branch BN structure for different purposes with different technical details. [59] proposes dual normalization for a better trade-off between accuracy and robustness, where the normalization is a weighted sum of normalized clean and adversarial input. [62] proposes a similar two-branch BN structure, where one branch is for adversarial examples and the other is for clean examples. The major difference between our work and [59, 62] is that both BN branches in TWINS are for adversarial examples and one branch (Frozen Net) has fixed BN statistics from pre-training so as to better maintain the pre-trained robustness, whereas [59, 62] uses clean examples in BN and aims to improve the accuracy for clean images.

3. The Model-based and Data-based Approach to Retaining Adversarial Robustness

This section introduces the two common approaches for keeping adversarial robustness of pre-training in downstream, model-based and data-based approaches. Denote the feature vector output of a neural network as $g_{\boldsymbol{\theta}}(\boldsymbol{x})$, the training sample in the downstream task as $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N \sim \mathbb{P}$, and the loss function as $\mathcal{L}(\boldsymbol{w}^T g_{\boldsymbol{\theta}}(\boldsymbol{x})+b, \boldsymbol{y})$, where $(\boldsymbol{w}, b)$ are the parameters of last classification layer. We assume that the pre-trained model is trained on adversarial examples generated from the PGD attack [49], where the l_∞ norm of the adversarial attack is bounded by ϵ, and during fine-tuning we use adversarial training with the same PGD attack to obtain adversarial robustness in downstream tasks. In short, we consider the robust pre-training and robust fine-tuning setting in this paper, if not specified otherwise.

Model-based approaches. We first introduce the model-based approach, which keeps the pre-trained model $\boldsymbol{\theta}_{pt}$ during fine-tuning so as to maintain its robustness. The objective function is

$$\sum_{(\boldsymbol{x}_n,\boldsymbol{y}_n)\sim\mathbb{P}} \mathcal{L}(\boldsymbol{w}^T g_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}_n)+b,\boldsymbol{y}_n)+\lambda_{LwF}\|g_{\boldsymbol{\theta}_{pt}}(\tilde{\boldsymbol{x}}_n)-g_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}_n)\|_2, \tag{1}$$

where the adversarial example generated from $\boldsymbol{x}_n$ is denoted as $\tilde{\boldsymbol{x}}_n$. The regularization term of the loss aims to minimize the distance between the features from the pre-trained and the fine-tuned models, which is expected to maintain the robustness of the pre-trained model. This approach is originally proposed in [40] to prevent the catastrophic forgetting in continual learning, and is used in [58] to preserve adversarial robustness in transfer learning. Note that [58] uses the LwF method in *standard* fine-tuning instead of robust fine-tuning as in our paper.

Data-based approaches. The objective function of the data-based approach is

$$\sum_{(\boldsymbol{x}_n,\boldsymbol{y}_n)\sim\mathbb{P}} \mathcal{L}(\boldsymbol{w}^T g_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}_n)+b,\boldsymbol{y}_n) + \lambda_{UOT} \sum_{(\boldsymbol{x}_m,\boldsymbol{y}_m)\sim\mathbb{Q}} \mathcal{L}(\boldsymbol{w}_q^T g_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}_m)+b_q,\boldsymbol{y}_m), \tag{2}$$

where $\mathbb{P}$ and $\mathbb{Q}$ are data distribution of the target and the pre-training tasks, $\boldsymbol{w}_q, b_q$ are the classification layer for pre-trained data from $\mathbb{Q}$. This method regularizes the current fine-tuned model feature extractor so that its prediction is still robust on the pre-training data. The joint training method is proposed in [47] to improve the performance of fine-tuning in downstream tasks where the training data is not sufficient.

Next, we test the performance of these two approaches on two standard image classification datasets, CIFAR10 [35] and Caltech-256 [23], with results shown in Figure 1. We use a grid search for learning rate and λ_{LwF} (λ_{UOT}) and report the result of the model with the best robust accuracy. See Section 5 and the supplemental material for the experiment setting. On CIFAR10, both approaches fail to improve either clean or robust accuracy; on Caltech-256, the two approaches improve the clean accuracy by a small margin but deteriorate the robustness. One reason why the model- and data-based approaches fail is that the regularization term might be too strong, thus hurting the learning in the downstream.

4. TWINS Fine-Tuning

The previous section demonstrates that both data- and model-based approaches cannot substantially improve the adversarial robustness in the downstream task. Thus we propose the TWINS for the better fine-tuning of robust pre-trained models for downstream adversarial robustness.

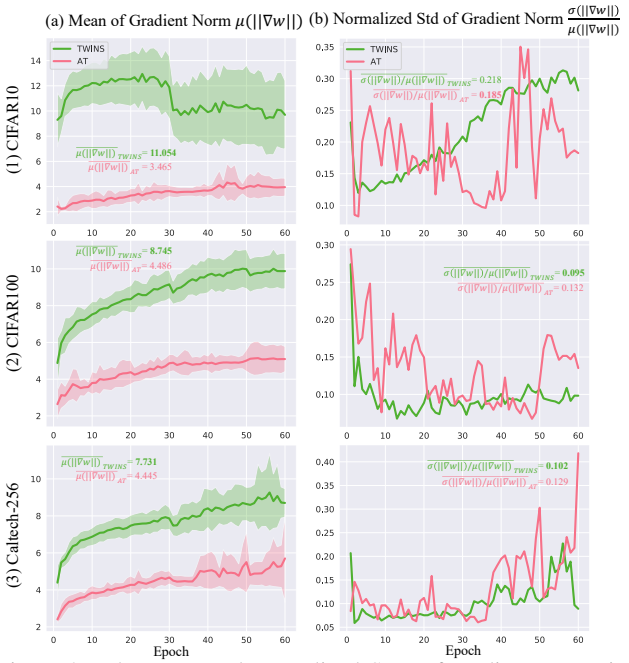

Figure 3. The mean and normalized STD of gradient norms in AT and TWINS-AT on CIFAR10, CIFAR100 and Caltech-256. The averaged $\mu(\|\nabla \boldsymbol{w}\|)$ and $\sigma(\|\nabla \boldsymbol{w}\|)/\mu(\|\nabla \boldsymbol{w}\|)$ over epochs are shown in each plot. The gradient magnitudes of TWINS-AT are substantially larger than those of AT, while the Normalized STDs of gradient norm in TWINS-AT are not obviously increased (CIFAR10) or even decreased (CIFAR100 and Caltech-256) compared with AT. This property leads to the faster escaping speed of TWINS-AT from the initial sub-optimum compared with AT (Fig. 4), and reduced robust overfitting (Tab. 1).

4.1. Proposed Method

Though BN layers contain only a few parameters compared to convolution and fully connected layers, they play an important role in the good performance of DNNs. [21] shows that even if we only train the BN layers, the performance of a DNN is already quite impressive. [51] finds that only training the parameters in BN layers in an image generator is effective for small datasets. [39] proposes adaptive BN for domain adaptation, which updates the BN statistics with data from a target domain. These works motivate us to propose a statistics-based approach for retaining pre-training information in the target task.

Typical BN layers track the mean and STD of the training set and save them for the inference stage. As this distribution information for each layer might be helpful for downstream robustness, we propose the TWINS robust fine-tuning, which maintains two networks, *Frozen Net* that uses the BN statistics from the robust pre-trained model, and *Adaptive Net* that learns its BN statistics from the downstream task. Instead of using two independent networks for the Frozen and Adaptive Net, we let the two networks share weight parameters, excluding the BN layers, to save the model size and inference time. At initialization, both

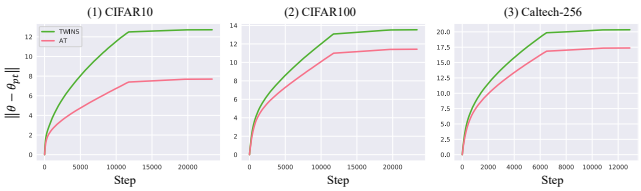

Figure 4. The distance between the current step's weight $\boldsymbol{\theta}$ and the initialization $\boldsymbol{\theta}_{pt}$. On the three datasets, TWINS-AT has a faster escaping speed from the sub-optimal initial model than AT, which is due to TWIN-AT's larger gradient norm as shown in Figure 3.

networks and their BN statistics are initialized by the robust pre-trained models. During training, the Frozen Net uses the population means and STDs of pre-training data computed in the pre-training stage in the normalization operation, while the Adaptive Net uses the current batch's mean and STD in the normalization and updates its running mean and STD with the target training data. Fig. 2 shows the general pipeline of TWINS training and the network structure.

The training objective of TWINS with adversarial training (TWINS–AT) in one mini-batch is:

$$\sum_{j=1}^{B/2} \mathcal{L}(\boldsymbol{w}^T g_{\boldsymbol{\theta}_a}(\tilde{\boldsymbol{x}}_j^{(a)}) + b, \boldsymbol{y}_j) + \tag{3}$$

$$\lambda_{twins} \sum_{i=B/2+1}^{B} \mathcal{L}(\boldsymbol{w}^T g_{\boldsymbol{\theta}_f}(\tilde{\boldsymbol{x}}_i^{(a)}) + b, \boldsymbol{y}_i), \tag{4}$$

where $\boldsymbol{\theta}_f$ and $\boldsymbol{\theta}_a$ denote the Frozen and Adaptive Nets respectively, and $\tilde{\boldsymbol{x}}^{(a)}$ is the adversarial image for $\boldsymbol{x}$ when attacking the Adaptive Net $\boldsymbol{\theta}_a$. We split the batch into two different sub-batches to avoid doubled batch sizes in TWINS training. Since the Frozen and Adaptive Nets share weight parameters, the number of parameters in TWINS is only increased by a very small amount (i.e., BN parameters). Thus, TWINS-AT only has a negligible cost in terms of memory and training time compared with vanilla AT. The TWINS structure can also be used in the TRADES in a similar way. Similar to [39], we can use the target training set to update the BN statistics so that they are more relevant to the downstream task. We call this procedure warmup in TWINS fine-tuning. The pseudo codes of TWINS-AT and TWINS-TRADES are given in Alg. 1 of the supplemental.

4.2. The mechanism of TWINS

The first benefit of TWINS is mentioned in the motivation of TWINS, i.e., the BN statistics have robustness information in the pre-training domain that can be leveraged by robust fine-tuning to improve the downstream robustness. This argument is validated by our ablation study in Section 5, where we initialize the means and STDs with 1.0 and 0.0 for Frozen Net instead of the pre-trained means and STDs and check the accuracy and robustness. Figure 5 shows that the TWINS with (1,0) initialization cannot match the performance with TWINS with pre-trained

Method	Rob. Acc.	C10	C100	Caltech	CUB	Dogs
AT	Best ↑	51.84	31.38	49.09	27.08	21.19
	Final ↑	49.41	28.52	48.37	26.60	19.80
	Gap ↓	2.43	2.86	0.73	0.48	1.39
TWINS-AT	Best ↑	53.23	31.60	48.80	29.24	20.89
	Final ↑	52.40	31.08	48.40	29.24	20.58
	Gap ↓	**0.83**	**0.52**	**0.40**	**0.00**	**0.29**

Table 1. The robust accuracy drop of AT and TWINS-AT, where the adversarial attack is PGD10. TWINS-AT has smaller accuracy drop compared with AT, indicating that the TWINS-AT is less prone to robust overfitting as a result of reduced variance of gradient norms and stable training as shown in Fig. 3.

statistics, indicating that the robustness information in pre-training is essential to the effectiveness of TWINS.

It is intriguing that even with the (1,0) initialization, TWINS still outperforms the AT baseline in terms of robustness or accuracy on some datasets, e.g., Stanford Dogs and CIFAR100. This suggests that besides retaining the pre-training information, TWINS provides some other benefits during robust fine-tuning. By analyzing the gradient of TWINS, we find that TWINS implicitly increases the effective learning rate without increasing the oscillation and empirically validate this finding. We detail this analysis next.

Effective learning rate We first write the gradient of a weight vector for TWINS training. Consider the l-th layer's weight $\boldsymbol{w}_j^{(l)}$ and its output after BN layer

$$\tilde{h}_{ij}^{(l)} = \frac{\hat{h}_{ij}^{(l)} - \frac{2}{B}\sum_{k=1}^{B/2} \hat{h}_{kj}^{(l)}}{\sqrt{\frac{2}{B}\sum_{k=1}^{B/2}(\hat{h}_{kj}^{(l)} - \hat{\mu}_j^{(l)})^2}} \tag{5}$$

$$= \frac{\boldsymbol{w}_j^{(l)^T}(\boldsymbol{h}_i^{(l-1)} - \boldsymbol{\mu}^{(l-1)})}{\sqrt{\frac{2}{B}\sum_{k=1}^{B/2}(\boldsymbol{w}_j^{(l)^T}(\boldsymbol{h}_i^{(l-1)} - \boldsymbol{\mu}^{(l-1)}))^2}}, \tag{6}$$

where we denote $h_{ij}^{(l)}$, $\hat{h}_{ij}^{(l)}$ and $\tilde{h}_{ij}^{(l)}$ as the output of ReLU, convolution or fully connected layer and BN layer respectively, for i-th sample, j-th output variable at l-th layer. The gradient with respect to $\boldsymbol{w}_j^{(l)}$ is

$$\nabla_a \boldsymbol{w}_j^{(l)} = \frac{\partial \mathcal{L}_a(\tilde{\boldsymbol{x}}_i)}{\partial \boldsymbol{w}_j^{(l)}} = \frac{\nabla \tilde{h}_{ij}^{(l)}(\boldsymbol{h}_i^{(l-1)} - \boldsymbol{\mu}^{(l-1)})}{\sqrt{\frac{2}{B}\sum_{k=1}^{B/2}(\boldsymbol{w}_j^{(l)^T}(\boldsymbol{h}_i^{(l-1)} - \boldsymbol{\mu}^{(l-1)}))^2}}$$
$$- \frac{\nabla \tilde{h}_{ij}^{(l)} \boldsymbol{w}_j^{(l)^T}(\boldsymbol{h}_i^{(l-1)} - \boldsymbol{\mu}^{(l-1)})}{(\frac{2}{B}\sum_{k=1}^{B/2}(\boldsymbol{w}_j^{(l)^T}(\boldsymbol{h}_i^{(l-1)} - \boldsymbol{\mu}^{(l-1)}))^2)^{3/2}} \boldsymbol{\Sigma}^{(l-1)} \boldsymbol{w}_j^{(l)},$$

where $\boldsymbol{\mu}^{(l-1)}$ and $\boldsymbol{\Sigma}^{(l-1)}$ are the mean and covariance matrix from $(l-1)$-th layer, $\nabla \tilde{h}_{ij}^{(l)}$ represents the gradient of loss with respect to $\tilde{h}_{ij}^{(l)}$. If we write the weight as its norm $\|\boldsymbol{w}_j^{(l)}\|$ multiplied by the unit vector $\boldsymbol{u}_j^{(l)}$, there is a relationship between $\nabla \boldsymbol{w}_j^{(l)}$ and $\|\boldsymbol{w}_j^{(l)}\|$,

$$\|\nabla_a \boldsymbol{w}_j^{(l)}\| = \frac{1}{\|\boldsymbol{w}_j^{(l)}\|} \|\nabla_a \boldsymbol{u}_j^{(l)}\| \tag{7}$$

This relationship has been found in [2, 28] and extended to any scale-invariant layers such as layer normalization [3] by [41]. It means that there are two ways to increase the gradient magnitude or the effective learning rate: 1) find a steeper descent direction where $\|\nabla_a \boldsymbol{u}_j^{(l)}\|$ is increased and 2) decrease the weight norm so that $1/\|\boldsymbol{w}_j^{(l)}\|$ is increased. For a DNN with standard BN layers, the training will exploit this property to increase the gradient magnitude by reducing the weight norms with the help of weight decay regularization, which leads to a spurious increase in gradient magnitude and a larger variance of gradient estimation. In contrast, for a DNN with fixed BN layers such as the Frozen Net, the gradient norm is not correlated with weight norm,

$$\nabla_f \boldsymbol{w}_j^{(l)} = \frac{\partial \mathcal{L}_f(\tilde{\boldsymbol{x}}_i)}{\partial \boldsymbol{w}_j^{(l)}} = \nabla \tilde{h}_{ij}^{(l)} \frac{\boldsymbol{h}_i^{(l-1)}}{\sigma_{j,pt}^{(l)}}. \tag{8}$$

In this gradient, the only way to increase the gradient magnitude is to find the actual steeper direction. The overall gradient for the weight is

$$\Delta \boldsymbol{w}_j^{(l)} = \nabla_a \boldsymbol{w}_j^{(l)} + \lambda_{twins} \nabla_f \boldsymbol{w}_j^{(l)}. \tag{9}$$

Empirical Study To see the difference between the gradients in AT and TWINS-AT, we record the gradient of all weight parameters in each step of the two training methods and compute the mean and STD of the gradient norm in each epoch. The model parameters and their gradients are treated as long vectors and we compute the l_2 norm of the weight and gradient vector. Figure 3 shows the mean and normalized STD of gradient norms in 60 training epochs for three datasets. Here we show the normalized STD, i.e., dividing the STD by the mean, to see the relative effect of variance. The major finding is that the gradient magnitude of TWINS-AT is substantially larger than that of AT, while the variance of TWINS-AT is lower than AT in most epochs. Note that in theory, the ratio between STD and mean should remain the same after down-scaling the weight norm in standard BN, but in practice we do observe the high normalized variance of DNNs with standard BNs since we use one epoch's gradients to approximate the variance and mean.

One benefit of the larger gradient magnitude is that the model can escape from the initial sub-optimal point faster and find a better local optimum than the small gradient optimization [42]. We validate this hypothesis by recording the distance between the current model and the initial model during training. Figure 4 shows these weight distances on three datasets, where TWINS-AT moves away from the initial model much faster than the AT baseline. The robust overfitting effect of adversarial training [55] is partially a result of large gradient variance, especially at the final stage of training [9]. We compare the robust accuracy drop of AT and TWINS-AT in Table 1 and find that the small relative variance of TWINS-AT has the effect of reducing robust overfitting. See the experiment details in Sec. 5 and Supp.

5. Experiment

This section presents our experiment with TWINS. We first introduce our experiment setting and then show our main result and ablation study.

5.1. Experiment Settings

Dataset. We use five datasets in our experiment. CIFAR10 and CIFAR100 [35] are low-resolution image datasets, where the training and validation sets have 50,000 and 10,000 images, and CIFAR10 has 10 classes, while CIFAR100 has 100 classes. Caltech-256 [23] is a high-resolution dataset with 30,607 images and 257 classes, which is split into training and validation set with a ratio of 9:1. Caltech-UCSD Birds-200-2011 (CUB200) [61] is a high-resolution bird image dataset for fine-grained image classification, which contains 200 classes of birds, 5,994 training images and 5,794 validation images. Stanford Dogs [34] has high-resolution dog images from 120 dog categories, where the training and validation set has 12,000 and 8,580 images. For both low-resolution image datasets (CIFAR10 and CIFAR100) and high-resolution datasets (Caltech-256,CUB200 and Stanford Dogs), we resize the image to 224×224 so that the input sizes are the same for pre-training and fine-tuning. As with pre-training, the input image is normalized by the mean and STD of the pre-training set. Note that the resizing and normalization function is integrated into the model so we can attack the input image with the [0,1] bounds for pixel values as in standard adversarial attacks. We use the standard ImageNet data augmentation for high-resolution datasets [27]. For CIFAR datasets, we use random cropping with padding=4 and random horizontal flipping.

Adversarial Pre-Training. Large-scale adversarial pre-training on ImageNet is time-consuming, and thus [56] has released adversarially pre-trained ResNet50 and WideResNet50-2, trained with l_2 and l_∞ norm bounded attacks. In this paper, we adopt the pre-trained ResNet50 models, trained with l_∞ attack with bound $\epsilon_{pt} = 4/255$. We test other robust pre-trained models in our ablation study.

Training Setting. For baselines and our method, we train all parameters of the pre-trained model, i.e., *full fine-tuning* instead of linear probing [58], with PGD attacks of l_∞ norm. The PGD step is 10, $\epsilon_{ft} = 8/255$ and stepsize $\alpha = 2/255$. We set the batch size as 128 and train the model for 60 epochs and divide the learning rate by 0.1 at 30th and 50th epoch. In TWINS with warmup, we initialize the means and STDs with their pre-trained values, and update the means and STDs using the target training set.

Metric	Method	CIFAR10	CIFAR100	Caltech256	CUB200	Stanford Dogs
Clean Acc.	AT	89.77	69.48	75.90	65.74	60.09
	TWINS-AT	91.24(+1.47)	70.72(+1.24)	76.86(+0.96)	68.09(+2.35)	64.98(+4.89)
	TWINS-AT+warmup	91.95(+2.18)	72.12(+2.64)	77.35(+1.45)	67.64(+1.90)	66.12(+6.03)
	TRADES	87.06	62.76	69.70	58.92	59.99
	TWINS-TRADES	86.61(-0.45)	66.72(+3.96)	71.12(+1.42)	60.72(+1.80)	60.58(+0.59)
	TWINS-TRADES+warmup	86.60(-0.46)	65.91(+3.15)	73.39(+3.69)	61.05(+2.13)	63.96(+3.97)
PGD10	AT	52.24	28.52	48.37	26.60	19.80
	TWINS-AT	52.73(+0.49)	31.08(+2.56)	48.40(+0.03)	29.24(+2.64)	20.58(+0.78)
	TWINS-AT+warmup	52.46(+0.22)	29.12(+0.60)	49.13(+0.76)	27.67(+1.07)	19.48(-0.32)
	TRADES	54.04	32.20	47.28	27.87	21.36
	TWINS-TRADES	56.23(+2.19)	33.51(+1.31)	47.31(+0.03)	27.05(-0.82)	19.93(-1.43)
	TWINS-TRADES+warmup	55.81(+1.77)	33.48(+1.28)	48.53(+1.25)	26.68(-1.19)	19.46(-1.90)
AA	AT	48.46	23.47	43.85	22.82	12.30
	TWINS-AT	**49.81**(+1.35)	**26.73**(+3.26)	43.69(-0.16)	22.33(-0.49)	**14.37**(+2.07)
	TWINS-AT+warmup	49.02(+0.56)	25.72(+2.25)	**43.92**(+0.07)	**23.58**(+0.75)	13.80(+1.50)
	TRADES	50.31	26.40	43.39	22.21	12.05
	TWINS-TRADES	**51.71**(+1.40)	28.29(+1.89)	41.77(-1.62)	**22.68**(+0.47)	**13.36**(+1.31)
	TWINS-TRADES+warmup	51.10(+0.79)	**28.30**(+1.90)	**43.55**(+0.16)	21.92(-0.29)	10.94(-1.11)

Table 2. The performance of our TWINS-AT and TWINS-TRADES on five image classification tasks compared with AT and TRADES. The clean accuracy means the accuracy when testing images are input without adversarial perturbations. PGD10 and AA denote the robust test accuracy under PGD10 and AutoAttack. The increase and decrease in performance are denoted with green and red numbers. The **bold** numbers denote the best robust accuracy under AA. The proposed TWINS achieves better robustness and clean accuracy compared with the baseline. Averaged over the datasets, the clean and robust accuracy of TWINS are increased by 2.18% and 1.21% compared with AT, and 1.46% and 0.69% compared with TRADES. The means and STDs of the performance are in the supplemental.

Metric	PT Model	Caltech256	CUB200	Dogs
Clean Acc	Random	48.99	12.38	7.27
	Non-Robust	64.66	53.30	41.99
	Robust	**75.90**	**65.74**	**60.09**
PGD10	Random	31.78	3.728	3.59
	Non-Robust	39.86	19.68	13.32
	Robust	**48.37**	**26.60**	**19.80**

Table 3. Comparison of random initialization, non-robust and robust pre-trained model on three difficult classification tasks, when fine-tuned with AT. The robust pre-trained model is indispensable to downstream robustness.

The momentum of updating statistics is 0.1, the batch size is 128, and the warmup only lasts one epoch. Note that in the warmup stage, the input samples are added with adversarial perturbations generated by the PGD attack, which has the same setting as the attack in training, and the classifier layer is the pre-trained classifier for the adversarial attack. Our pilot experiment shows that using adversarial examples as input is more effective than using clean exmaples in the warmup. The optimizer is SGD with momentum in all of our experiments. The learning rate, weight decay and regularization hyperparameter are determined by grid search, which is described in detail in the supplemental.

Adversarial robustness evaluation. Two standard adversarial attacks are used in our experiment, i.e., PGD and AutoAttack [12]. The adversarial perturbation is l_∞ norm bounded in our evaluation. The setting of PGD attack for validation set is the same as the PGD attack in training. The AutoAttack (AA) is a more reliable adversarial attack and more often used for evaluation than PGD in recent years.

Method	Metric	λ_{WD}			
		1e-5	1e-4	1e-3	1e-2
AT	Clean Acc.	89.92	89.94	89.77	**90.28**
	Robust Acc.	46.34	44.67	48.46	47.57
TWINS-AT	Clean Acc.	**91.90**	**91.42**	**91.24**	87.33
	Robust Acc.	**47.19**	**49.07**	**49.81**	**51.65**

Table 4. The performance of TWINS-AT and AT on CIFAR10 when the hyperparameter for weight decay λ_{WD} is changed. The robust accuracy is evaluated using AutoAttack. Our TWINS-AT achieves better adversarial robustness than AT for different λ_{WD}.

We use the standard attacks of AA, i.e., untargeted APGD-CE, targeted APGD-DLR, targeted FAB [11] and Square Attack [1], with $\epsilon = 8/255$. The robust accuracy in our experiment result denotes the accuracy under AA.

5.2. Experimental Result

On the three high-resolution datasets, we compare the performance of fine-tuning different initialization models, i.e., random initialization, standard pre-trained ResNet50 and robust pre-trained ResNet50. Table 3 shows that the pre-trained models are essential to the accuracy and robustness in challenging downstream tasks, since the random initialization is much worse than the two pre-trained models. The robust pre-trained model has a clear benefit over the standard pre-trained one, indicating that robust pre-training is indispensable to downstream robustness.

Table 2 shows the result of TWINS-AT and TWINS-TRADES compared with the baselines. Since AA is a more reliable attack, we highlight the best robust accuracy un-

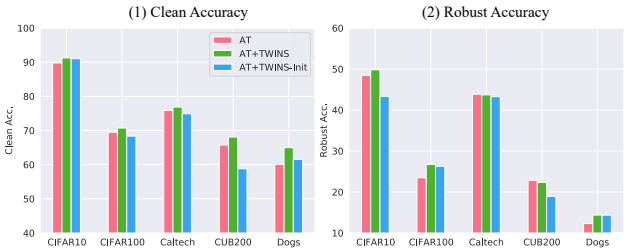

Figure 5. Ablation study where the BN statistics in TWINS are initialized with (0,1) for means and STDs, denoted as TWINS-Init. The population mean and STD of pre-training are crucial to TWINS.

der AA on each dataset. The TWINS fine-tuning learns more robust DNNs, as well as achieves a better clean accuracy on all five datasets, demonstrating the strong effectiveness of TWINS in the robust pre-training and robust fine-tuning setting. On CIFAR10 and CIFAR100, both TWINS-AT and TWINS-TRADES achieve better robustness and clean accuracy than their baselines, while the warmup only improves the clean accuracy and hurts the robustness sometimes. On Caltech-256, TWINS-AT improves upon the vanilla AT in both robustness and accuracy, but TWINS-TRADES does not perform better than vanilla TRADES. However, the warmup helps boost the performance of TWINS-TRADES as well as TWINS-AT and makes the TWINS with warmup perform better than the baselines.

On the two fine-grained image classification datasets, TWINS-AT and TWINS-TRADES generally perform better than baselines in terms of accuracy and robustness, if we look at the robust accuracy under AA, where only TWINS-AT on CUB200 has a slightly worse robust accuracy than its baseline. We find that the warmup improves the clean accuracy but hurts the robustness in most cases, except for CIFAR100 and Caltech-256. This can be a result of noisy adversarial perturbation, since we use the pre-trained classifier layer in the adversarial attack, or the insufficient update steps. Nevertheless, we note that the warmup can be considered as an operation for achieving a trade-off between robustness and accuracy.

5.3. Ablation Study

TWINS initialization. We use the pre-trained BN statistics in the Frozen Net to keep the robust information learned during pre-training. To show the importance of the pre-trained BN statistics, we use the standard initialization (mean=0 and STD=1) for BN statistics in the Frozen Net, denoted as TWINS-Init, and show the result on the five datasets in Figure 5. Both clean and robust accuracy drop when the (0,1) initialization is used in TWINS-Init, demonstrating the crucial role of pre-trained statistics in TWINS. The fact that TWINS-Init sometimes improves upon the AT baseline motivates us to investigate the effect of TWINS on gradient norms in Section 4.

Method	Metric	ϵ_{pt}			
		1/255	2/255	4/255	8/255
AT	Clean Acc.	65.47	67.08	69.48	69.93
	Robust Acc.	24.76	25.79	23.47	27.71
TWINS-AT	Clean Acc.	**68.55**	**70.45**	**70.72**	**72.59**
	Robust Acc.	**25.97**	**26.62**	**26.73**	**28.62**

Table 5. The performance of TWINS-AT and AT on CIFAR100 when robust pre-trained ResNet50 with different ϵ_{pt} are used.

Effect of weight decay. Weight decay is the reason for the decreasing weight norm in DNNs with BN layers, so increasing the hyperparameter of weight decay λ_{WD} is also a way to increase the gradient magnitude. Table 4 show the result of TWINS-AT and AT when different λ_{WD} are used when fine-tuning the robust pre-trained model on CIFAR10. The robust accuracy of TWINS-AT is consistently better than that of AT across different λ_{WD}'s. We draw the same conclusion on CIFAR100 (see supplemental). Note that the clean accuracy of TWINS-AT drops when a large λ_{WD} is used, suggesting that we should not use a too large λ_{WD} for TWINS-AT.

Different robust pre-trained models. The main experiment uses the robust pre-trained ResNet50 with $\epsilon_{pt} = 4/255$ as the initial model. We try different robust pre-trained models with different ϵ_{pt} in Table 5, which shows that a larger ϵ_{pt} is beneficial to both clean and robust accuracy in the downstream, and the proposed TWINS-AT is better than AT in both metrics with different pre-trained models.

6. Conclusion

This paper investigates the utility of robust pre-trained models in various downstream classification tasks. We first find that the commonly used data- and model-based approaches to maintain pre-training information do not work in the adversarial robust fine-tuning. We then propose a subtle statistics-based method, TWINS, for retaining the pre-training robustness in the downstream. In addition to the robustness preserving effect, we find that TWINS increases the gradient magnitude without sacrificing the training stability and improves the training dynamics of AT. Finally, the performance of TWINS is shown to be stronger than that of AT and TRADES on five datasets. One limitation of our work is that we only evaluate the robust supervised pre-trained ResNet50. Recently, robust pre-trained ViT's on ImageNet [4] have been released. Our statistics-based approach can be extended to the layer normalization, on which the increasing gradient magnitude argument also holds, and thus future work will extend TWINS to ViT.

Acknowledgement This work was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CityU 11215820) and the Fundamental Research Funds for the Central University of China (DUT No. 82232031).

References

[1] Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: a query-efficient black-box adversarial attack via random search. In *European Conference on Computer Vision*, pages 484–501. Springer, 2020. 7

[2] Sanjeev Arora, Zhiyuan Li, and Kaifeng Lyu. Theoretical analysis of auto rate-tuning by batch normalization. In *International Conference on Learning Representations*, 2019. 6

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 6

[4] Yutong Bai, Jieru Mei, Alan L Yuille, and Cihang Xie. Are transformers more robust than cnns? *Advances in Neural Information Processing Systems*, 34:26831–26843, 2021. 2, 8

[5] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021. 2

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 2

[7] Dian Chen, Hongxin Hu, Qian Wang, Li Yinli, Cong Wang, Chao Shen, and Qi Li. Cartl: Cooperative adversarially-robust transfer learning. In *International Conference on Machine Learning*, pages 1640–1650. PMLR, 2021. 2

[8] Tianlong Chen, Sijia Liu, Shiyu Chang, Yu Cheng, Lisa Amini, and Zhangyang Wang. Adversarial robustness: From self-supervised pre-training to fine-tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 699–708, 2020. 2, 3

[9] Tianlong Chen, Zhenyu Zhang, Sijia Liu, Shiyu Chang, and Zhangyang Wang. Robust overfitting may be mitigated by properly learned smoothening. In *International Conference on Learning Representations*, 2020. 6

[10] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014. 12

[11] Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *International Conference on Machine Learning*, pages 2196–2205. PMLR, 2020. 7

[12] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*, pages 2206–2216. PMLR, 2020. 1, 7

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 2

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 2

[15] Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. Mma training: Direct input space margin maximization through adversarial training. In *International Conference on Learning Representations*, 2019. 3

[16] Josip Djolonga, Jessica Yung, Michael Tschannen, Rob Romijnders, Lucas Beyer, Alexander Kolesnikov, Joan Puigcerver, Matthias Minderer, Alexander D'Amour, Dan Moldovan, et al. On robustness and transferability of convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16458–16468, 2021. 2, 3

[17] Xinshuai Dong, Anh Tuan Luu, Min Lin, Shuicheng Yan, and Hanwang Zhang. How should pre-trained language models be fine-tuned towards adversarial robustness? *Advances in Neural Information Processing Systems*, 34:4356–4369, 2021. 3

[18] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 2

[19] Shuo Feng, Xintao Yan, Haowei Sun, Yiheng Feng, and Henry X Liu. Intelligent driving intelligence test for autonomous vehicles with naturalistic and adversarial environment. *Nature communications*, 12(1):1–14, 2021. 1

[20] Samuel G Finlayson, John D Bowers, Joichi Ito, Jonathan L Zittrain, Andrew L Beam, and Isaac S Kohane. Adversarial attacks on medical machine learning. *Science*, 363(6433):1287–1289, 2019. 1

[21] Jonathan Frankle, David J. Schwab, and Ari S. Morcos. Training batchnorm and only batchnorm: On the expressive power of random features in {cnn}s. In *International Conference on Learning Representations*, 2021. 4

[22] Sven Gowal, Sylvestre-Alvise Rebuffi, Olivia Wiles, Florian Stimberg, Dan Andrei Calian, and Timothy A Mann. Improving robustness using generated data. *Advances in Neural Information Processing Systems*, 34:4218–4233, 2021. 1

[23] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007. 1, 4, 6

[24] Minghao Guo, Yuzhe Yang, Rui Xu, Ziwei Liu, and Dahua Lin. When nas meets robustness: In search of robust architectures against adversarial attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 631–640, 2020. 1

[25] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022. 2

[26] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual rep-

resentation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020. 2

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 6

[28] Elad Hoffer, Ron Banner, Itay Golan, and Daniel Soudry. Norm matters: efficient and accurate normalization schemes in deep networks. *Advances in Neural Information Processing Systems*, 31, 2018. 6

[29] Hanxun Huang, Yisen Wang, Sarah Erfani, Quanquan Gu, James Bailey, and Xingjun Ma. Exploring architectural ingredients of adversarially robust deep neural networks. *Advances in Neural Information Processing Systems*, 34:5545–5559, 2021. 1

[30] Jitesh Jain, Anukriti Singh, Nikita Orlov, Zilong Huang, Jiachen Li, Steven Walton, and Humphrey Shi. Semask: Semantically masked transformers for semantic segmentation. *arXiv preprint arXiv:2112.12782*, 2021. 2

[31] Daniel Jakubovitz and Raja Giryes. Improving dnn robustness to adversarial attacks using jacobian regularization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 514–529, 2018. 3

[32] Ahmadreza Jeddi, Mohammad Javad Shafiee, and Alexander Wong. A simple fine-tuning is all you need: Towards robust deep learning via adversarial fine-tuning. *arXiv preprint arXiv:2012.13628*, 2020. 2, 3

[33] Ziyu Jiang, Tianlong Chen, Ting Chen, and Zhangyang Wang. Robust pre-training by adversarial contrastive learning. *Advances in Neural Information Processing Systems*, 33:16199–16210, 2020. 2, 3

[34] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. CVPR workshop on fine-grained visual categorization (FGVC)*, volume 2. Citeseer, 2011. 6

[35] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 1, 4, 6

[36] Feng Li, Hao Zhang, Shilong Liu, Lei Zhang, Lionel M Ni, Heung-Yeung Shum, et al. Mask dino: Towards a unified transformer-based framework for object detection and segmentation. *arXiv preprint arXiv:2206.02777*, 2022. 2

[37] Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. Exploring plain vision transformer backbones for object detection. *arXiv preprint arXiv:2203.16527*, 2022. 2

[38] Yao Li, Martin Renqiang Min, Thomas Lee, Wenchao Yu, Erik Kruus, Wei Wang, and Cho-Jui Hsieh. Towards robustness of deep neural networks via regularization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7496–7505, 2021. 1

[39] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016. 4, 5

[40] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 1, 3, 4

[41] Zhiyuan Li, Kaifeng Lyu, and Sanjeev Arora. Reconciling modern deep learning with traditional optimization analyses: The intrinsic learning rate. *Advances in Neural Information Processing Systems*, 33:14544–14555, 2020. 6

[42] Chen Liu, Mathieu Salzmann, Tao Lin, Ryota Tomioka, and Sabine Süsstrunk. On the loss landscape of adversarial training: Identifying challenges and how to overcome them. *Advances in Neural Information Processing Systems*, 33:21476–21487, 2020. 2, 6

[43] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 128(2):261–318, 2020. 2

[44] Ziquan Liu and Antoni B Chan. Boosting adversarial robustness from the perspective of effective margin regularization. *British Machine Vision Conference (BMVC)*, 2022. 1, 3

[45] Ziquan Liu, Yufei Cui, and Antoni B Chan. Improve generalization and robustness of neural networks via weight scale shifting invariant regularizations. *ICML Workshop on Adversarial Machine Learning*, 2021. 1, 12

[46] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 2

[47] Ziquan Liu, Yi Xu, Yuanhong Xu, Qi Qian, Hao Li, Antoni Chan, and Rong Jin. Improved fine-tuning by better leveraging pre-training data. In *Thirty-Sixth Conference on Neural Information Processing Systems*, 2022. 1, 4, 12

[48] Ziquan Liu, Yi Xu, Yuanhong Xu, Qi Qian, Hao Li, Rong Jin, Xiangyang Ji, and Antoni B Chan. An empirical study on distribution shift robustness from the perspective of pre-training and data augmentation. *arXiv preprint arXiv:2205.12753*, 2022. 2

[49] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. 1, 3

[50] Shervin Minaee, Yuri Y Boykov, Fatih Porikli, Antonio J Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2021. 2

[51] Atsuhiro Noguchi and Tatsuya Harada. Image generation from small datasets via batch statistics adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2750–2758, 2019. 4

[52] Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Krishnamurthy Dvijotham, Alhussein Fawzi, Soham De, Robert Stanforth, and Pushmeet Kohli. Adversarial robustness through local linearization. *Advances in Neural Information Processing Systems*, 32, 2019. 1, 3

[53] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020. 2

[54] Sylvestre-Alvise Rebuffi, Sven Gowal, Dan Andrei Calian, Florian Stimberg, Olivia Wiles, and Timothy Mann. Data

augmentation can improve robustness. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. 1

[55] Leslie Rice, Eric Wong, and Zico Kolter. Overfitting in adversarially robust deep learning. In *International Conference on Machine Learning*, pages 8093–8104. PMLR, 2020. 6

[56] Hadi Salman, Andrew Ilyas, Logan Engstrom, Ashish Kapoor, and Aleksander Madry. Do adversarially robust imagenet models transfer better? *Advances in Neural Information Processing Systems*, 33:3533–3545, 2020. 1, 2, 3, 6

[57] Vikash Sehwag, Saeed Mahloujifar, Tinashe Handina, Sihui Dai, Chong Xiang, Mung Chiang, and Prateek Mittal. Robust learning meets generative models: Can proxy distributions improve adversarial robustness? In *International Conference on Learning Representations*, 2021. 1, 3

[58] Ali Shafahi, Parsa Saadatpanah, Chen Zhu, Amin Ghiasi, Christoph Studer, David Jacobs, and Tom Goldstein. Adversarially robust transfer learning. In *International Conference on Learning Representations*, 2020. 2, 3, 4, 6

[59] Chawin Sitawarin, Arvind Sridhar, and David Wagner. Improving the accuracy-robustness trade-off for dual-domain adversarial training. *ICML 2021 Workshop on Uncertainty and Robust- ness in Deep Learning*. 3

[60] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014*, 2014. 1

[61] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 2, 6

[62] Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan L Yuille, and Quoc V Le. Adversarial examples improve image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 819–828, 2020. 3

[63] Yutaro Yamada and Mayu Otani. Does robustness on imagenet transfer to downstream tasks? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9215–9224, 2022. 2, 3

[64] Yaodong Yu, Heinrich Jiang, Dara Bahri, Hossein Mobahi, Seungyeon Kim, Ankit Singh Rawat, Andreas Veit, and Yi Ma. An empirical study of pre-trained vision models on out-of-distribution generalization. In *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2021. 2

[65] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, pages 7472–7482. PMLR, 2019. 3, 12

A. Experimental Details

Fig. 1 shows the experiment result of UOT fine-tuning and Learning without Forgetting. In the UOT data selection, we follow the same experiment setting as in [47], i.e., the distance function in UOT is the cosine-based distance and $\epsilon_c = 0.01$. In all the experiments, we search the hyperparameters using grid search and determine the optimal hyperparameter based on the performance of the validation set. In UOT fine-tuning, λ_{UOT} is searched from $\{0.001, 0.01, 0.1, 1.0, 10.0\}$, learning rate is searched from $\{0.001, 0.003, 0.01\}$ and weight decay is searched from {1e-5,1e-4,1e-3}. In LwF, learning rate and weight decay are searched with the same range as in UOT fine-tuning and λ_{LwF} is searched from {1e-5,1e-4,1e-3,1e-2,1e-1}. In Fig. 1a, UOT's hyperparameter is $\lambda_{UOT} = 0.01, \eta = 0.001, \lambda_{WD}$ = 1e-4 and LwF's hyperparameter is λ_{LwF} = 1e-4, $\eta = 0.001, \lambda_{WD}$ = 1e-4. In Fig. 1b, UOT's hyperparameter is $\lambda_{UOT} = 0.01, \eta = 0.01, \lambda_{WD}$ = 1e-4 and LwF's hyperparameter is λ_{LwF} = 1e-2, $\eta = 0.003, \lambda_{WD}$ = 1e-4.

In Tab. 6, we list the hyperparameters of baselines and TWINS in Tab. 2. The β parameter in TRADES and TWINS-TRADES is fixed as 6.0, which is the default value in the original TRADES experiment [65], so we only search the learning rate and weight decay parameter in TRADES. The search range for learning rate η is {3e-4,1e-3,3e-3,1e-2,3e-2}, for weight decay λ_{WD} is {1e-5,1e-4,1e-3,1e-2}, for λ_{twins} is {0.1,0.2,0.3,0.4,0.5,1.0}.

Tab. 1 shows the best and final robust accuracy under PGD10 attack. The robust accuracy is not evaluated using AutoAttack for the efficiency since we evaluate the model for every epoch. To save space, we use C10 and C100 to denote CIFAR10 and CIFAR100. Caltech, CUB and Dogs are short for Caltech-256, CUB200 and Stanford Dogs. In Tab. 3, we use the same hyperaparameter search for random initialization and standard pre-trained model initialization. For random initialization, we increase the training time to 120 epochs and decay the learning rate at 60 and 100 epochs to improve the performance and make a more fair comparison. Tab. 4 and Tab. 10 show the performance of AT and TWINS-AT on CIFAR10 and CIFAR100 when λ_{WD} is changed. We keep other hyperparameters untouched and only change the λ_{WD}. Tab. 5 changes the pre-trained model and keeps other hyperparameters the same as in Tab. 6.

For AT and TWINS-AT, we run the experiment on Nvidia-3090-24G GPU. For TRADES and TWINS-TRADES, we run the experiment on Nvidia-V100-32G GPU since they require a double batch size in effect. We run each experiment twice for baselines and TWINS and report the one with the optimal robustness performance.

Algorithm 1 gives the pseudo-code of TWINS-AT and TWINS-TRADES for reference. We include our code for TWINS-AT and TWINS-TRADES in the supplemental and encourage the readers to run our code.

B. Derivation of Equation (7)

We include the derivation of Equation (7) to make our paper self-contained. We use the γ to represent the weight norm $\|\boldsymbol{w}_j^{(l)}\|$, then the gradient of $\boldsymbol{w}_j^{(l)}$ in Adaptive Net is written as $\nabla_a \gamma \boldsymbol{u}_j^{(l)}$ and we extract the norm term γ out of the gradient,

$$\begin{aligned}\nabla_a \gamma \boldsymbol{u}_j^{(l)} &= \frac{\nabla \tilde{h}_{ij}^{(l)} (\boldsymbol{h}_i^{(l-1)} - \boldsymbol{\mu}^{(l-1)})}{\gamma \sqrt{\frac{2}{B}\sum_{k=1}^{B/2} (\boldsymbol{u}_j^{(l)^T} (\boldsymbol{h}_i^{(l-1)} - \boldsymbol{\mu}^{(l-1)}))^2}} \\ &\quad - \frac{\nabla \tilde{h}_{ij}^{(l)} \boldsymbol{u}_j^{(l)^T} (\boldsymbol{h}_i^{(l-1)} - \boldsymbol{\mu}^{(l-1)})}{\gamma (\frac{2}{B}\sum_{k=1}^{B/2} (\boldsymbol{u}_j^{(l)^T} (\boldsymbol{h}_i^{(l-1)} - \boldsymbol{\mu}^{(l-1)}))^2)^{3/2}} \boldsymbol{\Sigma}^{(l-1)} \boldsymbol{u}_j^{(l)} \\ &= \frac{1}{\gamma} \nabla_a \boldsymbol{u}_j^{(l)}, \qquad (10)\end{aligned}$$

$$\Rightarrow \|\nabla_a \boldsymbol{w}_j^{(l)}\| = \frac{1}{\|\boldsymbol{w}_j^{(l)}\|} \|\nabla_a \boldsymbol{u}_j^{(l)}\| \qquad (11)$$

This gives us the relationship between weight norm and its gradient norm.

C. More Experiment Result

Tab. 10 shows the performance of TWINS-AT when different λ_{WD}'s are used. For λ_{WD}=1e-4, 1e-3 and 1e-2, the adversarial robustness of TWINS-AT is better than the AT baseline. Note that as in the CIFAR10 experiment, TWINS-AT does not achieve a better robust accuracy when λ_{WD} is too large, i.e., 1e-2. On other 3 WD settings, the clean accuracy of TWINS is clearly better than AT. Thus, the performance of TWINS-AT is not very sensitive to the weight decay hyperparameter if we use λ_{WD}=1e-3 or 1e-4.

The clean and robust accuracy on the validation set in each epoch of AT and TWINS-AT are shown in Fig. 6. On CIFAR10, CIFAR100, Caltech, and Dogs, the convergence of TWINS-AT is faster than the AT baseline, as a result of larger effective learning rate and faster escape from initialization. The figure also shows that the robust overfitting effect is obvious on CIFAR10 (Fig. 6.a2) and CIFAR100 (Fig. 6.b2), while TWINS-AT is less prone to the robust overfitting.

Table 9 shows the experiment result of running three trials for AT and TWINS-AT. The improvement of TWINS over AT is significant in most cases.

We run an experiment using the DTD dataset [10], which only contains texture images and looks very different from ImageNet. The clean (robust) accuracy of TWINS-AT and AT is 57.39% (21.54%) and 55.96% (21.17%). Similar to [45], we find that although the domain difference is large, keeping the pre-training information during fine-tuning is beneficial, especially for the clean accuracy.

Method	CIFAR10	CIFAR100	Caltech256	CUB200	Stanford Dogs
AT	(1e-3,1e-3)	(1e-3,1e-4)	(3e-3,1e-3)	(1e-2,1e-4)	(1e-3,1e-4)
TWINS-AT	(3e-3,1e-3,1.0)	(1e-3,1e-4,0.3)	(3e-3,1e-3,0.4)	(3e-3,1e-4,0.3)	(3e-3,1e-4,1.0)
TWINS-AT+warmup	(3e-3,1e-3,1.0)	(1e-3,1e-4,1.0)	(3e-3,1e-3,0.4)	(1e-2,1e-4,0.3)	(3e-3,1e-4,0.3)
TRADES	(1e-2,1e-4)	(1e-3,1e-4)	(1e-2,1e-4)	(1e-2,1e-4)	(1e-3,1e-4)
TWINS-TRADES	(1e-2,1e-4,1.0)	(1e-2,1e-4,1.0)	(3e-3,1e-4,1.0)	(1e-2,1e-4,3.0)	(3e-3,1e-4,1.0)
TWINS-TRADES+warmup	(1e-2,1e-4,1.0)	(1e-2,1e-4,1.0)	(3e-3,1e-4,0.4)	(1e-2,1e-4,0.3)	(1e-3,1e-4,1.0)

Table 6. The hyperparameters of AT, TRADES and TWINS in our experiment. The format means (η, λ_{WD}, λ_{twins}) for TWINS and (η, λ_{WD}) for baselines.

We ran the experiment of only fine-tuning the classification layer and updating the BN stats, while fixing the backbone weights. Table 8 shows that learning with the pre-trained robust features achieves high clean accuracy but low robust accuracy, consistent with our experiment result that the warmup in TWINS mainly benefits the clean accuracy.

We try different λ_{twins} on CIFAR10 in Table 7. TWINS is not very sensitive to λ_{twins}, when it is not very large.

	1e-1	3e-1	1.0	3.0	10.0
Clean Acc.	91.18	91.56	91.24	90.46	10.0
AA	49.11	48.51	49.8	48.51	10.0

Table 7. Performance of TWINS with different λ_{twins} on CIFAR10.

	Fine-tune Adv. Train			Fine-tune Std. Train		
Dataset	C10	C100	Caltech	C10	C100	Caltech
Clean Acc.	79.48	63.28	78.97	92.11	77.92	81.41
PGD10	12.46	9.84	36.17	0.67	1.00	21.01

Table 8. Adversarial and standard fine-tuning with fixed backbone weights and adaptive BN stats.

	Metric	C10	C100	Caltech	CUB	Dogs
AT	Clean	89.81(0.16)	69.02(0.42)	75.31(0.55)	65.51(0.25)	59.50(0.54)
	Rob.	47.56(1.15)	24.93(1.29)	**43.39(0.52)**	**23.25(0.39)**	11.90(0.68)
TWINS-AT	Clean	**91.56(0.38)**	**70.95(0.55)**	**76.92(0.50)**	**67.89(0.17)**	**65.25(0.26)**
	Rob.	**48.99(0.89)**	**26.06(0.62)**	42.43(1.09)	21.89(0.38)	**13.95(0.54)**

Table 9. Mean and std of AT and TWINS.

Method	Metric	λ_{WD}			
		1e-5	1e-4	1e-3	1e-2
AT	Clean Acc.	68.51	69.48	67.92	**66.48**
	Robust Acc.	**25.45**	23.47	**26.69**	27.16
TWINS-AT	Clean Acc.	**71.64**	**70.72**	**70.74**	65.74
	Robust Acc.	24.65	**26.73**	**26.69**	**27.81**

Table 10. The performance of TWINS-AT and AT on CIFAR100 when the hyperparameter for weight decay λ_{WD} is changed.

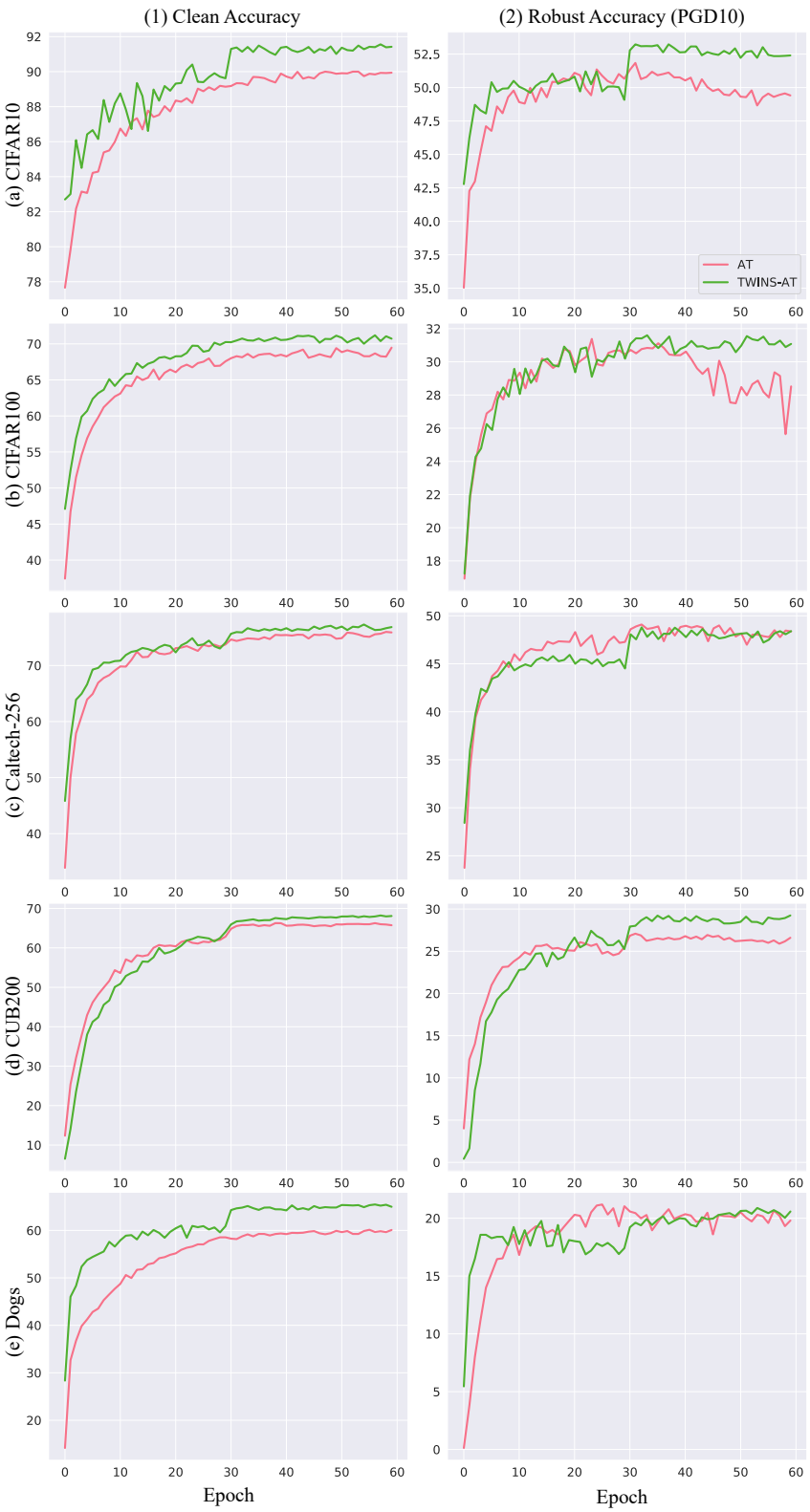

Figure 6. The curve of clean accuracy and robust accuracy (PGD10 attack) on validation set.

Algorithm 1: Training algorithm of TWINS-AT and TWINS-TRADES

Input: Training data $\mathcal{D}_{tr}$, TWINS parameter λ_{twins}, learning rate η, parameter of TRADES β, number of epochs N_e, batch size B

Output: Model parameters ($\boldsymbol{\theta}$,$\boldsymbol{w}$,b)

Initialize model parameters and BN statistics with pre-trained parameters and BN statistics;

for $i = 1, \ldots, N_e$ **do**

  Adjust η;

  Split $\mathcal{D}_{tr}$ into N_B =ceil(N_{tr}/B) batches;

  for $b = 1, \ldots, N_B$ **do**

    Generate adversarial examples $\{\tilde{\boldsymbol{x}}_i^{(a)}, \boldsymbol{y}_i\}_{i=1}^{B}$ by attacking Adaptive Net $\boldsymbol{\theta}_a$;

    if *AT* **then**

      $Loss = \sum_{j=1}^{B/2} \mathcal{L}(\boldsymbol{w}^T g_{\boldsymbol{\theta}_a}(\tilde{\boldsymbol{x}}_j^{(a)}) + b, \boldsymbol{y}_j) + \lambda_{twins} \sum_{i=B/2+1}^{B} \mathcal{L}(\boldsymbol{w}^T g_{\boldsymbol{\theta}_f}(\tilde{\boldsymbol{x}}_i^{(a)}) + b, \boldsymbol{y}_i)$;

    end

    if *TRADES* **then**

      $Loss = \frac{2}{B} \sum_{j=1}^{B/2} \mathcal{L}(\boldsymbol{w}^T g_{\boldsymbol{\theta}_a}(\tilde{\boldsymbol{x}}_j^{(a)}) + b, \boldsymbol{y}_j) + \beta \frac{2}{B} \sum_{j=1}^{B/2} D_{KL}(\boldsymbol{w}^T g_{\boldsymbol{\theta}_a}(\tilde{\boldsymbol{x}}_j^{(a)}) + b, \boldsymbol{w}^T g_{\boldsymbol{\theta}_a}(\boldsymbol{x}_j) + b) +$
      $\lambda_{twins} [\frac{2}{B} \sum_{i=B/2+1}^{B} \mathcal{L}(\boldsymbol{w}^T g_{\boldsymbol{\theta}_f}(\tilde{\boldsymbol{x}}_i^{(a)}) + b, \boldsymbol{y}_i) + \beta \frac{2}{B} \sum_{i=B/2+1}^{B} D_{KL}(\boldsymbol{w}^T g_{\boldsymbol{\theta}_f}(\tilde{\boldsymbol{x}}_i^{(a)}) + b, \boldsymbol{w}^T g_{\boldsymbol{\theta}_f}(\boldsymbol{x}_i) + b)]$;

    end

    $(\boldsymbol{\theta}, \boldsymbol{w}, b) := (\boldsymbol{\theta}, \boldsymbol{w}, b) - \eta \nabla_{(\boldsymbol{\theta}, \boldsymbol{w}, b)} Loss$ # $\boldsymbol{\theta}_a$ and $\boldsymbol{\theta}_f$ share the parameter $\boldsymbol{\theta}$

  end

end