Efficient and Effective Augmentation Strategy for Adversarial Training

Sravanti Addepalli[†] [*] **Samyak Jain**[†] [◇] [‡] [*] **R.Venkatesh Babu**[†]
[†] Video Analytics Lab, Indian Institute of Science, Bangalore
[◇] Indian Institute of Technology (BHU) Varanasi

Abstract

Adversarial training of Deep Neural Networks is known to be significantly more data-hungry when compared to standard training. Furthermore, complex data augmentations such as AutoAugment, which have led to large gains in standard training of image classifiers, have not been successful with Adversarial Training. We first explain this contrasting behavior by viewing data augmentations in training as a problem of domain generalization and further propose Diverse Augmentation-based Joint Adversarial Training (DAJAT) to effectively use data augmentations in adversarial training. We aim to handle the conflicting goals of enhancing the diversity of the training dataset, and training with data that is close to the test distribution, by using a combination of simple and complex augmentations with separate batch normalization layers during training. We further utilize the popular Jensen-Shannon divergence loss to encourage the *joint* learning of the *diverse augmentations*, thereby allowing simple augmentations to guide the learning of complex ones. Lastly, to improve the computational efficiency of the proposed method, we propose and utilize a two-step defense, Ascending Constraint Adversarial Training (ACAT) that uses an increasing epsilon schedule and weight-space smoothing to prevent gradient masking. The proposed method DAJAT achieves substantially better robustness-accuracy trade-off when compared to existing methods on the RobustBench Leaderboard on ResNet-18 and WideResNet-34-10. The code for implementing DAJAT is available here: https://github.com/val-iisc/DAJAT.

1 Introduction

Deep Neural Network (DNN) based image classifiers are vulnerable to crafted imperceptible perturbations known as Adversarial Attacks [45] that can flip the predictions of the model to unrelated classes leading to disastrous implications. Adversarial Training [17, 31, 57] has been the most successful defense strategy, where a model is explicitly trained to be robust in the presence of such attacks. While early defenses focused on designing suitable loss functions for training, subsequent works [33, 38] showed that with careful hyperparameter tuning, even the two most popular methods PGD-AT [31] and TRADES [57] yield comparable performance, highlighting the saturation in performance with respect to changes in the training loss. Schmidt et al. [39] observed that adversarial training has a large sample complexity and further gains require the use of additional training data. Subsequent works [7, 19] indeed used additional data whose distribution is close to that of the original dataset, in order to obtain performance gains. The availability of large amounts of relevant data is impractical to assume, leading to an exploration towards augmentations based on Generative Adversarial Networks [16] and Diffusion based models [23, 19]. However, the use of such generative models incurs an

[*] Equal Contribution.
Correspondence to Sravanti Addepalli <sravantia@iisc.ac.in>, Samyak Jain <samyakjain.cse18@itbhu.ac.in>
[‡] Work done during internship at Video Analytics Lab, Indian Institute of Science.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

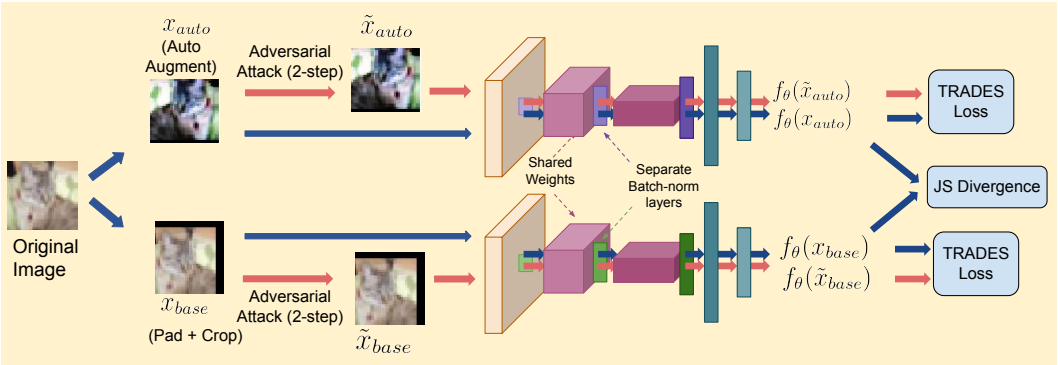


Figure 1: A Schematic representation of the proposed approach DAJAT

additional training cost, and suffers from limited diversity in low-data regimes, and in datasets with high resolution images.

A simple and efficient solution to improve the diversity of training data in standard Empirical Risk Minimization (ERM) based training has been the use of random transformations such as rotation, color jitter, and variations in contrast, sharpness and brightness [28, 10, 11], which can change images significantly in input space while belonging to the same class as the original image. However, prior works have surprisingly found that such augmentations that cause large changes in the input distribution do not help adversarial training [38, 18, 44]. This limits the augmentations in adversarial training to simple ones - zero padding followed by random crop, and horizontal flip [38, 33, 18] - which may not be able to fill in the large data requirement of Adversarial Training.

In this work, we firstly analyze the reasons for this contrasting trend between Standard and Adversarial Training, and further show that **it is indeed possible to utilize complex augmentations effectively in Adversarial training as well**, by jointly training on simple and complex data augmentations using separate batch-normalization layers for each kind as shown in Fig.1. While complex augmentations increase the data diversity resulting in better generalization, simple augmentations ensure that the model specializes on the training data distribution as well. We further minimize the Jenson-Shannon divergence between the softmax outputs of various augmentations to enable the simple augmentations to guide the learning of complex ones. In order to improve the computational efficiency of the proposed method, we use two attack steps (instead of 10) during training. By progressively increasing the magnitude of perturbations and performing smoothing in weight space, we show that it is indeed possible to improve the stability of training. Our contributions are listed below:

- We analyze the reasons for the failure of strong data augmentations in adversarial training by viewing data augmentations in training as a domain generalization problem, and further propose *Diverse Augmentation based Joint Adversarial Training* (DAJAT) to utilize data augmentations effectively in Adversarial training. The proposed approach can be integrated with many augmentations and adversarial training methods to obtain performance gains.
- We propose and integrate DAJAT with an efficient 2-step defense strategy, *Ascending Constraint Adversarial Training* (ACAT) that uses linearly increasing ε schedule, cosine learning rate and weight-space smoothing to prevent gradient masking and improve convergence.
- We obtain improved robustness and large gains in standard accuracy on multiple datasets (CIFAR-10, CIFAR-100, ImageNette) and model architectures (RN-18, WRN-34-10). We obtain remarkable gains in a low data scenario (CIFAR-100, Imagenette) where data augmentations are most effective. **On CIFAR-100, we outperform all existing methods on the RobustBench leaderboard [9], including those that utilize additional training data.**

2 Related Works

We discuss various existing strategies for improving the Adversarial robustness of Deep Networks.

Adversarial Training (AT): Goodfellow et al. [17] proposed FGSM-AT where single-step adversarial samples were used for training. However, these models were susceptible to gradient masking [34], where the local loss landscape becomes convoluted leading to the generation of weaker single-step

attacks during training. This leads to a false sense of security against single-step attacks, while the models are still susceptible to stronger multi-step attacks such as PGD [31]. PGD-AT [31, 38] used multi-step attacks in a similar adversarial training formulation to obtain robust models that stood the test of time against several attacks [3, 8, 42]. TRADES [57] explicitly optimizes the trade-off between the accuracy on natural and adversarial examples by minimizing the cross-entropy loss on natural images along with the Kullback-Leibler (KL) divergence between the predictions of adversarial and clean images. In the proposed defense, we use the base loss from TRADES-AT [57].

Several works have explored the use of auxiliary techniques in AT such as weight-space smoothing [26, 50], architectural changes [18, 53] and increasing the diversity of training data by using additional natural and synthetic data [39, 7, 18, 37]. Increasing the diversity of training data achieves significant performance gains since the sample complexity of adversarial training is known to be high [39].

Augmentations in Adversarial Training: While data augmentations such as contrast, sharpness and brightness adjustments are known to improve performance in the standard training regime, they have not led to substantial gains in adversarial training. AutoAugment [10] uses Proximal Policy Optimization to find the set of policies that can yield optimal performance on a given dataset. Contrary to prior works [38, 18], we show that the policies that are optimized for standard training indeed yield a boost in performance of adversarial training as well, when used in the proposed training framework.

In a recent work, Rebuffi et al. [37] show that it is possible to obtain substantial gains in robust accuracy by using spatial composition based augmentations such as CutMix [54] and CutOut [13] that preserve low-level features of the image. Cutmix replaces part of an image with another and also combines the output softmax vectors in the same ratio, while CutOut blanks out a random area of an image. The authors hypothesize that the augmentations used in Adversarial training need to preserve low-level features, which severely limits the possibilities for mitigating the large sample complexity in adversarial training. We show that by using the proposed approach DAJAT, it is indeed possible to use augmentations such as color jitter, contrast, sharpness and brightness adjustments that significantly change the low level statistics of images.

3 Preliminaries: Notation and Threat Model

We consider the Adversarial Robustness of DNN based image classifiers. An input image is denoted as $x \in \mathcal{X}$ and the corresponding ground truth label as $y \in [0, 1]$. We denote a simple transformation of x obtained using Pad, Crop and Horizontal flip (referred to as Base augmentations) by x_{base} and other transformations of x by the respective subscript. For example, x_{auto} refers to the image x being transformed by AutoAugment (AA) [10] followed by the base augmentations. The function mapping of the classifier C from input space $\mathcal{X}$ to the softmax vectors is denoted using $f_\theta(.)$, where θ denotes the network parameters. Adversarial examples corresponding to the images x, x_{base} and x_{auto} are denoted using $\tilde{x}$, $\tilde{x}_{\text{base}}$ and $\tilde{x}_{\text{auto}}$ respectively. We consider the ℓ_∞ norm based threat model, where $\tilde{x}$ is a valid perturbation within ε if it belongs to the set $\mathcal{A}_\varepsilon(x) = \{\tilde{x} : ||\tilde{x} - x||_\infty \leq \varepsilon\}$.

4 Motivation: Role of Augmentations in Neural Network Training

In this section, we first explore the contrasting factors that influence the training of neural networks when data is augmented (Sec.4.1), and further delve into the specifics of adversarial training which make it challenging to obtain gains using complex data augmentations (Sec.4.2).

4.1 Impact of Augmentations in Neural Network Training

Conjecture-1: We hypothesize that the role of data augmentations in the training of Neural Networks is influenced by the following contrasting factors:

(i) Reduced overfitting due to an increase in diversity of the augmented dataset, leading to better generalization of the network to the test set.

(ii) Larger domain shift between the augmented data distribution and the test data distribution, leading to a drop in performance on the test set.

(iii) Capacity of the Neural Network in being able to generalize well to the augmented data distribution and the unaugmented data distribution for the given task.

Justification: The training of Neural Networks using augmented data can be considered as a problem of domain generalization, where the network is trained on a source domain (augmented data) and is

Table 1: **Impact of augmentations:** Performance (%) of ACAT models on Base augmentations and AutoAugment (Auto). Clean and robust accuracy against GAMA attack [42] are reported. The use of AutoAugment results in $\sim 1.5 - 2\%$ drop in robust accuracy.

Model	Test: Train ↓	No Aug Clean	No Aug Robust	AutoAugment Clean	AutoAugment Robust
ResNet-18	Base	82.41	**50.00**	63.79	37.07
	Auto	**82.54**	48.11	**76.40**	**43.22**
WideResNet-34-10	Base	86.71	**55.58**	68.24	40.83
	Auto	**86.80**	53.99	**82.64**	**48.98**

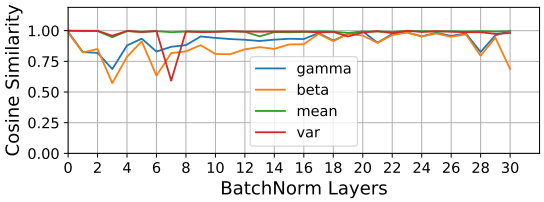


Figure 2: **Comparison of BN layer statistics** for a WRN-34-10 model trained on CIFAR-10 using DAJAT. BN layers of the Base augmentations (Pad+Crop,H-Flip) are compared with those of AutoAugment. Initial layer (L3) parameters are diverse, while those of deeper layers (L25) are similar.

expected to generalize to a target domain (test data). We use the theoretical formulation by Ben-David et al. [4] shown below to justify the respective claims in Conjecture-1:

$$\epsilon_t(f) \leq \epsilon_s(f) + \frac{1}{2} d_{\mathcal{F}\Delta\mathcal{F}}(s,t) + \lambda \tag{1}$$

(i) The use of a more diverse or larger source dataset reduces overfitting, improving the performance of the network on the source distribution. From Eq.1, expected error on the target distribution ϵ_t (test set in this case) is upper bounded by the expected error on the source distribution ϵ_s (augmented dataset) along with other terms. Therefore, improved performance on the augmented distribution can improve the performance on the test set as well.

(ii) The expected error on the target distribution ϵ_t is upper bounded by the distribution shift between the source and target distributions $\frac{1}{2} d_{\mathcal{F}\Delta\mathcal{F}}(s,t)$ along with other terms. Therefore, a larger domain shift between the augmented and test data distributions can indeed limit the performance gains on the test set.

(iii) The constant λ in Eq.1 measures the risk of the optimal joint classifier: $\lambda = \min_{f \in \mathcal{F}} \epsilon_s(f) + \epsilon_t(f)$.

Neural Networks with a higher capacity can minimize the expected risk on the source set ϵ_s and the risk of the optimal joint classifier effectively. Therefore, capacity of the Neural Network and complexity of the task influence the gains that can be obtained using augmentations.

4.2 Analysing the role of Augmentations in Adversarial Training

We analyse the trade-off between the factors described in Conjecture-1 for adversarial training when compared to standard ERM training. In addition to the goal of improving accuracy on clean samples, adversarial training aims to achieve local smoothness of the loss landscape as well. Hence, the complexity of Adversarial Training is higher than that of standard ERM training, making it important to use larger model capacities to obtain gains using data augmentations (based on Conjecture-1 (iii)). This justifies the gains obtained by Rebuffi et al. [37] on the WRN-70-16 architecture by using CutMix based augmentations (2.9% higher robust accuracy and 1.23% higher clean accuracy). The same method does not obtain significant gains on smaller architectures such as ResNet-18 where a 1.76% boost in robust accuracy is accompanied by a 2.55% drop in clean accuracy.

Secondly, while the distribution shift between augmented data and test data ($\frac{1}{2} d_{\mathcal{F}\Delta\mathcal{F}}(s,t)$) may be sufficiently low for natural images leading to improved generalization to test set (Conjecture-1(i,ii)), the same may not be true for adversarial images. There is a large difference between the augmented data and test data in pixel space, although they may be similar in feature space. Since adversarial attacks perturb images in pixel space, the distribution shift between the corresponding perturbations widens further as shown in Fig.7, 8 and 9. Based on Conjecture-1(ii), unless this difference is accounted for, complex augmentations cannot improve the performance of adversarial training. (Ref. Appendix-B). This trend has also been observed empirically by Rebuffi et al. [37], based on which they conclude that the augmentations designed for robustness need to preserve low-level features.

We present the performance of Adversarial Training by using either Base augmentations (Pad+Crop, Flip) or AutoAugment [10] during training and inference on the CIFAR-10 dataset using ResNet-18 and WideResNet-34-10 architectures in Table-1. Firstly, we note that by using AutoAugment during training alone, robust accuracy on the test set drops by $\sim 1.5 - 2\%$ which is as observed in prior work [18]. Secondly, the clean and robust accuracy drop by $\sim 6.5\%$ when augmented images are

used for both training and testing, highlighting the complexity of the learning task. We present additional results by training without any augmentations, and by training using a combination of both augmentations in every minibatch in Table-15. We note that the use of Base Augmentations alone (Pad+Crop+HFlip) still gives the best overall performance on the unaugmented test set.

5 Proposed Method

5.1 Background

We briefly discuss the TRADES-AWP defense [57, 50], which is the base algorithm used in DAJAT.

$$\mathcal{L}_{\text{AWP}} = \max_{\hat{\theta} \in \mathcal{M}(\theta)} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_{\text{CE}}(f_{\theta+\hat{\theta}}(x_i), y_i) + \beta \cdot \max_{\tilde{x}_i \in \mathcal{A}_\varepsilon(x_i)} \text{KL}(f_{\theta+\hat{\theta}}(x_i) || f_{\theta+\hat{\theta}}(\tilde{x}_i)) \tag{2}$$

Firstly, an adversarial attack is generated by maximizing the KL divergence between the softmax predictions of the clean and adversarial examples iteratively for 10 attack steps. An Adversarial Weight Perturbation (AWP) step additionally perturbs the weights of the model to maximize the overall loss in weight-space. The weight perturbations are constrained in the feasible region $\mathcal{M}(\theta)$ such that for a given layer l, $||\hat{\theta}_l|| \leq \gamma \cdot ||\theta_l||$. The overall training loss is a combination of Cross-entropy loss on clean samples and the KL divergence term. The latter is weighted by a factor β that controls the robustness-accuracy trade-off. Training the model using Adversarial Weight Perturbations leads to smoothing of loss surface in the weight space, resulting in better generalization [50, 44].

5.2 Diverse Augmentation based Joint Adversarial Training (DAJAT)

As discussed in Section-4, the use of augmentations in training can be viewed as a problem of domain generalization, where performance on the source distribution or augmented dataset is crucial towards improving the performance on the target distribution or test set. Since adversarial training is inherently challenging, for limited model capacity, it is difficult to obtain good performance on the training data that is transformed using complex augmentations. Moreover, the large distribution shift between augmented data and test data, specifically with respect to low-level statistics, results in poor generalization of robust accuracy to the test set.

To mitigate these challenges, we propose the combined use of simple and complex augmentations during training, so that the model can benefit from the diversity introduced by complex augmentations, while also specializing on the original data distribution that is similar to the simple augmentations. We propose to use separate batch normalization layers for simple and complex augmentations, so as to offset the shift in distribution between the two kinds of augmentations. In our main approach, we propose to use Pad and Crop followed by Horizontal Flip (Pad+Crop+HFlip) as the Simple augmentations, and Autoaugment followed by Pad+Crop+HFlip as the complex augmentations. We justify the choice of this augmentation pipeline in Appendix-B.3.

Motivated by AugMix [22], we additionally minimize the Jenson-Shannon (JS) divergence between the softmax outputs of different augmentations, so as to allow the simple augmentations to guide the learning of complex ones. We present the training loss $\mathcal{L}_{\text{DAJAT}}$ of the proposed method, Diverse Augmentation based Joint Adversarial Training (DAJAT) below in Eq.5:

$$\mathcal{L}_{\text{TR}}(\theta, x, y) = \mathcal{L}_{\text{CE}}(f_\theta(x), y) + \beta \cdot \max_{\tilde{x} \in \mathcal{A}_\varepsilon(x)} \text{KL}(f_\theta(x) || f_\theta(\tilde{x})) \tag{3}$$

$$\tilde{\theta} = \underset{\hat{\theta} \in \mathcal{M}(\theta)}{\text{argmax}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_{\text{TR}}(\theta + \hat{\theta}, x_{i,\text{base}}, y_i) \tag{4}$$

$$\mathcal{L}_{\text{DAJAT}} = \frac{1}{T+1} \cdot \frac{1}{N} \sum_{i=1}^{N} \{\mathcal{L}_{\text{TR}}(\tilde{\theta}, x_{i,\text{base}}, y_i) + \sum_{t=1}^{T} \mathcal{L}_{\text{TR}}(\tilde{\theta}, x_{i,\text{auto(t)}}, y_i)\} + \frac{1}{N} \sum_{i=1}^{N} \text{JSD}(f_{\tilde{\theta}}(x_{i,\text{base}}), f_{\tilde{\theta}}(x_{i,\text{auto(1)}}), \ldots, f_{\tilde{\theta}}(x_{i,\text{auto(T)}}) \tag{5}$$

Adversarial attacks are generated individually for each augmentation by maximizing the respective KL divergence term of the TRADES loss shown in Eq.3. To improve training efficiency, we compute $\tilde{x}$ using two attack steps with a step-size of ε. We use a combination of a linearly increasing schedule

of ε, cosine learning rate schedule and model weight-averaging [26] to improve the stability and performance of adversarial training (Details in Sec.5.4). The DAJAT loss (Eq.5) is a combination of the TRADES 2-step loss on each of the augmentations x_{base} and $x_{\text{auto(t)}}$, along with an adversarial weight perturbation step on the loss corresponding to the base augmentations alone to reduce computational cost. For every batch normalization layer, two sets of running statistics and affine parameters are maintained and used for simple and complex augmentations respectively (Ref: Algorithm-1).

While we use AutoAugment in our main approach, we show in Table- 6 that the proposed approach works well with other augmentations as well. The role of the base (primary) augmentations is primarily to learn the batch normalization layers that would be used during inference time, and to provide better supervision for the training of complex augmentations using the JS divergence term. The role of the complex (secondary) augmentations is to enhance the diversity of the training dataset. Therefore we use a single primary augmentation and multiple instances of the secondary augmentation. We find that the gains in performance saturate with the addition of more instances of secondary augmentation, and therefore recommend the use of a single base augmentation and two instances of secondary augmentation for the best performance-accuracy trade-off. We note from Table-2 that in this setting, the computational complexity of the proposed method is on par with the TRADES-AWP [57, 50] defense, while achieving considerable performance gains.

5.3 Split Batch Normalization Layers for Different Augmentations

A Batch Normalization (BN) layer [25] is implemented as follows on a given feature map $g(x_i)$ of an input image x_i: $\hat{g}(x_i) = \frac{g(x_i)-\mu}{\sigma} \cdot \gamma + \beta$. Here, μ and σ denote the mean and standard deviation of the current mini-batch during training. During inference, these are set to the running mean and variance computed during training. γ and β constitute parameters of the network that are trained.

Prior works use separate batch normalization layers to improve the performance of standard [52, 49, 32] and adversarial training [51, 48], in both supervised [52, 49, 32, 51, 48] and self-supervised settings [27, 15]. The proposed defense DAJAT uses separate batch normalization layers for simple and complex augmentations as discussed in the previous section. We do not use separate batch normalization for clean and adversarial images here.

In DAJAT, we maintain two sets of BN statistics μ and σ, and two sets of affine parameters, β and γ for every BN layer. We plot the cosine similarity between the BN layers corresponding to Base augmentations and AutoAugment across different layers on a WideResNet-34-10 model trained using DAJAT on CIFAR-10 in Fig.2. While the mean and variance are relatively more similar across most of the layers (0-30 in x-axis), we note significant differences in the γ and β values, specifically in the initial layers. We present a detailed study on the importance of having separate running statistics and affine parameters in Appendix-C, and conclude that our method works effectively by separating either the running statistics or affine parameters alone as well. However, it works best by using a combination of both (Table-11). We further note that very small differences in running statistics (as seen in Fig.2 and 11) can also result in noteworthy performance gains. Finally, we find that the use of a single batch-norm layer for both augmentations degrades results significantly. Therefore, the use of different parameters for each augmentation type ensures that the function mapping differs for each augmentation, thereby effectively offsetting the large differences in low level statistics.

5.4 Ascending Constraint Adversarial Training (ACAT)

In this section, we discuss the methods incorporated to improve the training efficiency of DAJAT in greater detail. We apply these methods to the TRADES-AWP defense to independently analyse their impact, and term the proposed defense as Ascending Constraint Adversarial Training (ACAT). We aim to improve the training efficiency by reducing the number of attack steps of the base defense from 10 to 2. We use two attack steps for training since it is known to be more stable when compared to single-step adversarial training, while still being computationally efficient [43].

As shown in Fig.3, naively reducing the number of attack steps to 2 in TRADES-AWP AT (Fixed constraint AT) causes a drop in clean and robust accuracy. While the drop is larger at higher training ε, a drop in clean accuracy is seen at $\varepsilon = 8/255$ as well. Further, the large robustness gap between last and best epochs indicates that the training stability deteriorates towards the end of training. The instability of TRADES-AWP AT at $\varepsilon = 8/255$ is higher for larger model capacities (WideResNet-34-10) as discussed in Appendix-E.1.

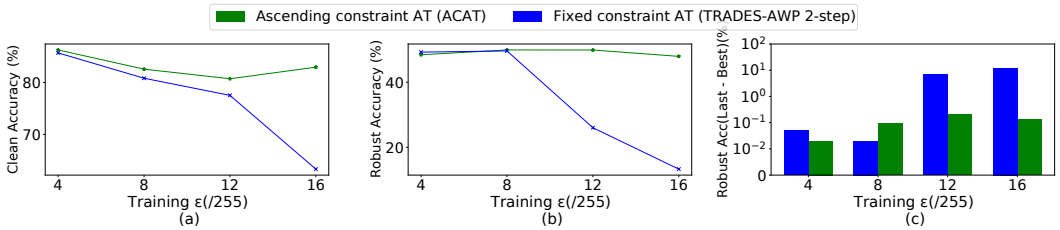


Figure 3: **Comparison of ACAT (2-step) against TRADES-AWP 2-step baseline** on CIFAR-10 with ResNet-18 architecture. (a) Clean accuracy, (b) Robust accuracy, and (c) Difference in Robust accuracy between the last and best epochs are reported. ACAT has better performance and stability even at large training ε values. Robust Accuracy is reported against GAMA attack [42] on $\varepsilon = 8/255$.

Curriculum learning approaches [41, 6] have been used to improve robustness-accuracy trade-off in adversarial training. Prior works [40, 1] also show that training convergence at large ε bounds can be improved by linearly increasing ε as training progresses. Inspired by this, we propose to linearly increase ε alongside a cosine learning rate schedule and weight-space smoothing for improving the stability and convergence of training (Ref: Algorithm-2). The use of a lower learning rate at larger ε improves training stability. As shown in Fig.3 and Table-13, the performance and stability of the ACAT are significantly better when compared to the TRADES-AWP 2-step baseline, specifically at larger perturbation bounds and higher model capacities, at the same computational cost. The proposed defense maintains a good clean accuracy at all the training ε values considered, and has $< 0.1\%$ robustness difference between last and best epochs.

We thus show that even without modifying the training loss function it is possible to obtain significant gains in the stability and performance of 2-step adversarial training by using the proposed Ascending Constraint Adversarial Training. The proposed method can indeed improve the performance of other defenses such as GAT [42] and NuAT [43] as well, as shown in Table-14.

Analysis of the proposed ACAT defense:

Conjecture-2: We hypothesize that given a Neural Network f_{θ_ε} that minimizes the TRADES loss [57] within a maximum perturbation radius of ε, and has sufficient smoothness in weight space within an ℓ_2 radius ψ around θ_ε, there exist $\varepsilon' > \varepsilon$ and a model $f_{\theta_{\varepsilon'}}$ where $||\theta_{\varepsilon'} - \theta_\varepsilon|| \leq \psi$, such that $f_{\theta_{\varepsilon'}}$ has a lower TRADES loss within ε' when compared to f_{θ_ε}.

Justification presented in Section-E.3.

Based on Conjecture-2, by selecting a small enough learning rate η and large enough ψ, it is possible to move towards a better solution w.r.t. the TRADES loss at ε'. The smoothness of loss surface further ensures that the gradients at ε' are bounded with a low magnitude, preventing the onset of catastrophic overfitting, which is known to be associated with large magnitude gradients [30]. This motivates the proposed approach ACAT that effectively combines an increasing ε schedule, cosine learning rate schedule that allows low learning rates at larger ε values, and adversarial weight perturbation to ensure smoothness in the loss surface.

Stability improvements obtained using ACAT: We consider an extreme case of using single-step attacks for training, to demonstrate the stability improvements obtained using ACAT. We consider a checkpoint obtained at $\varepsilon = 7.25/255$ while training using the proposed ACAT defense on CIFAR-10 dataset for a WideResNet-34-10 architecture. We use this model as an initialization and perform single-step adversarial training using the TRADES-AWP training objective at ε=$7.5/255$. Our goal here is to merely verify the stability of single-step training, although the use of single-step attack makes the robustness gains marginal. We present the clean accuracy and robust accuracy against PGD-20 attack using different values of Adversarial Weight Perturbation step size γ in Fig.4(a). It is evident from the plots that for a fixed value of learning rate, increasing γ can delay the onset of gradient masking, thereby improving the training stability. We also note marginal improvements in performance at higher γ, due to the enhanced generalization. Therefore, weight-space smoothing indeed helps in improving the performance and stability of the proposed defense. We also note from Fig.4(b) that for a fixed value of $\gamma = 0.01$, lower learning rate leads to better stability, therefore cosine schedule also helps in improving stability. We note that the value of gamma in relation to the learning rate is crucial. Therefore, in the proposed approach, we use a fixed value of γ and reduce the learning rate using a cosine learning rate schedule to improve training stability as ε increases.

Table 2: **CIFAR-10:** Performance (%) of the proposed defenses ACAT and DAJAT when compared to the state-of-the-art. Robust evaluations are performed on GAMA [42] and AutoAttack [8].

		CIFAR-10, ResNet-18				CIFAR-10, WideResNet-34		
Method	**Steps**	**Clean**	**GAMA**	**AutoAttack**	**Time/epoch (sec)**	**Clean**	**GAMA**	**AutoAttack**
NuAT2-WA [43]	2	82.21	**50.97**	**50.75**	109	86.32	55.08	54.76
ACAT, Ours (Base, 2step)	2	**82.41**	50.00	49.80	95	**86.71**	**55.58**	**55.36**
PGD-AT [31]	10	81.12	49.08	48.75	182	86.07	52.70	52.19
TRADES-AWP [57, 50]	10	80.47	50.06	49.87	228	85.19	55.87	55.69
TRADES-AWP-WA	10	80.41	49.89	49.67	228	85.10	56.07	55.87
TRADES-AWP-WA (200 epochs)	10	81.99	51.65	51.45	228	85.36	56.35	56.17
DAJAT, Ours (Base, AA)	2 + 2	85.60	51.27	51.06	160	87.87	56.97	56.68
DAJAT, Ours (Base, 2*AA)	2 + 4	85.99	51.71	51.48	219	**88.90**	57.22	56.96
DAJAT, Ours (Base, 3*AA)	2 + 6	**86.67**	**51.81**	**51.56**	280	88.64	**57.34**	**57.05**

Table 3: **CIFAR-100, ImageNette:** Performance (%) of the proposed defense DAJAT when compared to the state-of-the-art. Robust evaluations are performed on GAMA [42] and AutoAttack [8].

		CIFAR-100, ResNet-18			CIFAR-100, WideResNet-34			IN-10, ResNet-18		
Method	**No. Steps**	**Clean**	**GAMA**	**AutoAttack**	**Clean**	**GAMA**	**AutoAttack**	**Clean**	**GAMA**	**AutoAttack**
TRADES-AWP [57, 50]	10	58.81	25.51	25.30	62.41	29.70	29.54	82.73	57.52	57.40
TRADES-AWP-WA	10	59.88	25.81	25.52	62.73	29.92	29.59	82.03	57.04	56.89
ACAT, Ours (Base, 2step)	2	62.05	26.35	26.10	65.75	30.61	30.23	82.34	57.12	56.96
DAJAT, Ours (Base, AA)	2 + 2	65.75	27.58	27.21	67.82	**31.65**	31.26	85.27	61.50	61.19
DAJAT, Ours (Base, 2*AA)	2 + 4	66.84	27.61	27.32	68.74	31.58	**31.30**	86.01	**62.52**	**62.31**
DAJAT, Ours (Base, 3*AA)	2 + 6	**66.96**	**27.90**	**27.62**	**70.35**	31.15	30.89	**86.92**	62.14	61.89

6 Experiments and Results

6.1 State-of-the-art comparison

We compare the performance of the proposed defenses ACAT and DAJAT against the 2-step and 10-step state-of-the-art defenses NuAT2-WA [42], TRADES-AWP [57, 50] and PGD-AT [31, 38] in Tables- 2, 3 and 4 on the CIFAR-10, CIFAR-100 [29] and ImageNette [24, 12] datasets. Although there have been several defenses after this, we show in Table-22 that TRADES-AWP outperforms them in most settings, and the proposed defense DAJAT consistently outperforms all methods, showing substantial gains in performance. Further, since we integrate the proposed approach with TRADES-AWP, we primarily compare our approach with this across all datasets. We show in Table-23 that DAJAT can be integrated with other base defenses [1, 36] as well to obtain performance gains. We integrate model weight averaging with the TRADES-AWP defense (termed as TRADES-AWP-WA) as well for a fair comparison. We train all models for 110 epochs unless specified otherwise.

ACAT: We compare the performance of the proposed 2-step defense ACAT with the existing state-of-the-art 2-step defense NuAT2-WA [43] in the first partition of Table-2. While we achieve similar performance on ResNet-18, we obtain improvements in both clean and robust accuracy on WideResNet-34-10. We show in Section-E.4 that our proposed defense ACAT can be integrated with the Nuclear Norm training objective as well to obtain improved results. The performance of the proposed ACAT defense is superior to PGD-AT (10-step) defense [38] as well. When compared to the TRADES-AWP (10-step) defense, ACAT achieves substantial gains in clean accuracy and improved robust accuracy on CIFAR-100 (Table-3), and improved clean accuracy at a slight drop in robust accuracy on CIFAR-10 (Table-2), at half the computational cost.

DAJAT: We present three variants of the proposed defense DAJAT, by using one, two and three AutoAugment based augmentations for every image, denoted as (Base, AA), (Base, 2*AA) and (Base, 3*AA) respectively. From Tables-2 and 3, we note that even by using only a single augmentation (Base, AA), we obtain improved clean and robust accuracy when compared to most of the baselines considered across all datasets and models. By increasing the number of augmentations to 2 (Base, 2*AA), we observe consistent gains in robust and clean accuracy in all cases. In this setting, the computational complexity of the proposed approach matches with that of TRADES-AWP [50] as shown in Table-2. With the setting (Base, 3*AA), we further obtain marginal improvements in performance over (Base, 2*AA). Overall, using the (Base, 2*AA) approach, which has comparable time complexity as the TRADES-AWP 10-step defense, we obtain large gains ranging from 3.8% to 7% on clean accuracy and $\sim 1.8\%$ in robust accuracy against AutoAttack [8] across most settings. On the Imagenette dataset we obtain 4.2% higher clean accuracy and 4.49% higher robust accuracy, showing that augmentation strategies work best when the amount of training data is less when

Table 4: Performance (%) of DAJAT using **higher attack steps** and 200 training epochs on **CIFAR-10 and CIFAR-100**. Evaluations are performed against AutoAttack [8] and Common Corruptions (CC) [21].

Dataset	Model	Method	Clean	AutoAttack	CC
CIFAR-10	ResNet-18	TRADES-AWP-WA	81.99	51.45	72.64
		Ours (Base, 2*AA)	**85.71**	**52.50**	**76.13**
	WRN-34-10	TRADES-AWP-WA	85.36	56.17	75.83
		Ours (Base, 2*AA)	**88.71**	**57.81**	**80.12**
CIFAR-100	ResNet-18	TRADES-AWP-WA	59.11	25.97	47.95
		Ours (Base, 2*AA)	**65.45**	**27.69**	**54.85**
	WRN-34-10	TRADES-AWP-WA	60.30	28.68	48.95
		Ours (Base, 2*AA)	**68.75**	**31.85**	**56.95**

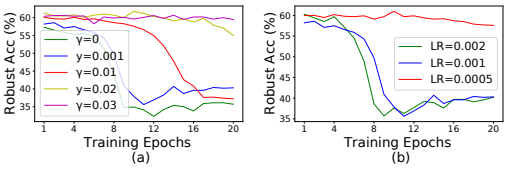


Figure 4: **Impact of (a) AWP step size γ and (b) learning rate (LR) on stability of single-step TRADES-AWP training** using $\varepsilon = 7.5/255$ on a model pretrained upto $\varepsilon = 7.25/255$ using ACAT. (a) For a fixed LR, increasing γ improves training stability. (b) For a fixed value of $\gamma = 0.01$, lower LR leads to better stability.

compared to the complexity of the task. We observe a similar trend using smaller subsets of training data on CIFAR-10 in Fig. 12.

Scaling to higher attack steps, longer training and larger models: While the use of 2 attack steps helps in improving the training efficiency of DAJAT, we show that by using more attack steps and longer training epochs, we can indeed obtain further gains in performance. The varying ε schedule in DAJAT allows the use of an increasing schedule in the number of steps as well, thereby limiting the overall cost associated with higher attack steps. We present results by increasing the number of attack steps from 2 to 5 uniformly every 50 epochs in Table-4. We obtain $\sim 3.5\%$ and $6-9\%$ higher clean accuracy, along with $1-1.6\%$ and $1.7-3.2\%$ gains in robust accuracy against AutoAttack on CIFAR-10 and CIFAR-100 respectively when compared to the TRADES-AWP-WA 200 epoch baseline. This achieves better robustness-accuracy trade-off when compared to all existing defenses that use the same model architecture on the RobustBench leaderboard [9]. On the CIFAR-100 RobustBench leaderboard, we outperform all existing methods using the same model architecture, w.r.t. both clean and robust accuracy, including the ones that use additional data, showing that data augmentations can be effectively used to overcome the large sample complexity of adversarial training. The use of diverse augmentations in DAJAT also improves the generalization of the model to Common Corruptions by 4% and 8% on CIFAR-10-C and CIFAR-100-C datasets [21] respectively as shown in Table-4. Further, we find that the performance gains obtained using DAJAT are higher on larger capacity models (Ref:Table-16).

Computational Efficiency: As shown in Appendix-F.6, the use of ACAT strategy in the proposed DAJAT defense enables us to achieve similar computational complexity as TRADES-AWP defense, while obtaining gains in performance. Using DAJAT, we achieve 10% reduction in FLOPs (training) and training time, while obtaining 3.8-5.5% higher clean accuracy and 1-1.6% higher robust accuracy.

6.2 Ablation experiments

We present ablation experiments to highlight the significance of different components of the proposed approach in Table-5 on the CIFAR-10 dataset using ResNet-18 architecture. All experiments are run for 110 training epochs, except A7 which is run for 220 epochs. We show the importance of the JS divergence term in the proposed loss in the ablations A1-A6 of Table-5. Using the JS divergence term we obtain $\sim 1\%$ higher clean accuracy across (Base, AA), (Base, 2*AA) and (Base, 3*AA) settings of the proposed defense. For (Base, 2*AA) and (Base, 3*AA) we obtain marginal improvements in robust accuracy as well. From A7, A9 and A10, we find that the proposed JS divergence term helps even in the case where both augmentations of an image are generated using the same pipeline. Using two AutoAugment based transformations, we obtain 1.6% higher clean accuracy when compared to the 220 epoch 2-step defense at a comparable computational cost. Comparing A9, A10 and A11, we note that the use of simple and complex augmentations indeed shows improvements over the case of using 2 complex or simple augmentations alone. The importance of split-batch norm in the proposed approach can be evidently seen by comparing A12 and A14. By using single batch norm (A12), robust accuracy drops by 8.24%. Further, in this case if the JS term is also dropped, the robustness of the network is almost completely lost. This shows that using a single batch norm layer for diverse augmentations makes it harder for the network to converge. We also note that the JS divergence term indeed helps in improving the convergence of training in addition to improving performance. From Table-17, we note that the use of varying epsilon schedule in DAJAT improves the stability of training resulting in large performance gains w.r.t. the use of fixed ε, specifically at larger model capacities.

Table 5: **Ablation experiments** performed on the **CIFAR-10** dataset using ResNet-18 architecture. Robust Accuracy is reported against GAMA attack [42]

Method	# Steps	Clean	Robust	Method	# Steps	Clean	Robust
[A1] Ours (Base, AA), no JS div	2 + 2	84.55	51.45	[A8] Ours (Base, 2step)	2	82.41	50.00
[A2] Ours (Base, AA)	2 + 2	85.60	51.27	[A9] Ours (AA, AA)	2 + 2	84.68	49.71
[A3] Ours (Base, 2*AA), no JS div	2 + 4	85.07	51.53	[A10] Ours (Base, Base)	2 + 2	83.93	49.88
[A4] Ours (Base, 2*AA)	2 + 4	85.99	51.71	[A11] Ours (Base, AA)	2 + 2	85.60	51.27
[A5] Ours (Base, 3*AA), no JS div	2 + 6	85.31	51.67	[A12] Ours (Base, 3*AA) Single BN	2 + 6	86.68	43.57
[A6] Ours (Base, 3*AA)	2 + 6	86.67	51.81	[A13] Ours (Base, 3*AA) Single BN, no JS	2 + 6	75.64	4.20
[A7] Ours (Base, 2step), 220 epochs	2	83.05	50.31	[A14] Ours (Base, 3*AA)	2 + 6	86.67	51.81

Table 6: Impact of using **other augmentations** in DAJAT. Performance on CIFAR-10 dataset with ResNet-18 architecture is reported. Robust evaluations are done against GAMA attack [42]. †PreAct-ResNet18 with Swish activation is used [37, 35].

	Augmentation		Base + Aug		Base + 2 * (Aug)			Augmentation		Base + Aug		Base + 2 * (Aug)	
Augmentation	Clean	Robust	Clean	Robust	Clean	Robust	Augmentation	Clean	Robust	Clean	Robust	Clean	Robust
No Augmentation	76.32	43.20	78.08	41.71	77.42	41.07	Cutout [13]	82.38	**50.14**	84.91	**51.40**	85.11	51.60
Pad+Crop+H-Flip	82.41	50.00	83.69	51.30	83.62	51.09	Colour Jitter	**82.98**	48.82	84.50	51.19	84.85	51.62
AutoAugment [10]	82.54	48.11	84.94	51.23	**85.99**	**51.71**	Mixup [56]	79.08	45.07	85.18	50.18	84.24	50.01
Cutmix [54]	79.03	41.57	82.33	50.90	81.64	49.50	RandAugment [11]	82.48	44.66	84.61	51.01	85.47	51.33
Cutmix† [37, 35]	82.01	47.65	84.58	50.97	85.49	51.58	Augmix [22]	82.38	48.84	**84.96**	50.4	85.18	50.51

6.3 Combining the proposed defense with other augmentations

We explore combining the proposed defense DAJAT with other augmentations in Table-6, and the impact of individual augmentations in AutoAugment in Table-10. We do not use the JS divergence term for Cutmix and Mixup since they involve changes in the label space. Without using any augmentation in the training dataset, we obtain poor clean and robust accuracy, highlighting the importance of the simple augmentations. The proposed approach is able to obtain good performance gains using AutoAugment [10], Color Jitter and CutOut [13] augmentations, highlighting that it can work well with pixel-level and spatial augmentations. In case of CutMix [54] and Mixup [56], since the base accuracy obtained by using the augmentation alone is itself very poor, combining them with the proposed approach does not yield competent results. However they are considerably better than naively using the augmentation. This shows that the proposed approach enables the use of a variety of augmentations without the need for careful selection. The use of Cutmix and Mixup in adversarial training are challenging since they involve changes in label space. We note that Rebuffi et al. [37] obtain considerable gains using Cutmix along with many other improvements. However, naively augmenting using CutMix leads to a considerably degraded performance. By incorporating some of the tricks reproduced by Rade et al. [35] we obtain improvements in the CutMix baseline and obtain further gains in the proposed method as well (Ref: Appendix-F.7, Table-21).

7 Conclusion

Contrary to prior knowledge, we show that it is indeed possible to use common augmentation strategies that modify the low-level statistics of images to improve the performance of adversarial training. We propose a novel defense Diverse Augmentation based Joint Adversarial Training (DAJAT), that uses a combination of simple and complex augmentations with separate batch normalization layers to allow the network training to benefit from the diverse training data distribution obtained using complex augmentations, while also specializing on a distribution that is close to the test set. The use of JS divergence term between network predictions of different augmentations enables the joint learning across various augmentations. We improve the efficiency of the proposed defense by utilizing the proposed approach Ascending Constraint Adversarial Training (ACAT), that improves the stability and performance of TRADES 2-step adversarial training significantly by using a linearly increasing ε schedule along with a cosine learning rate schedule and weight-space smoothing.

Limitations: While we focus our work on *how* to use augmentations effectively in adversarial training, we do not focus on *which* augmentations are best suited for the same. We believe this work can open up further possibilities towards finding better data augmentations for adversarial training.

8 Acknowledgments and Disclosure of Funding

This work was supported by a research grant (CRG/2021/005925) from SERB, DST, Govt. of India. Sravanti Addepalli is supported by a Google PhD Fellowship in Machine Learning.

References

[1] S. Addepalli, S. Jain, G. Sriramanan, and V. B. Radhakrishnan. Scaling adversarial training to large perturbation bounds. In *The European Conference on Computer Vision (ECCV)*, 2022. 7, 8, 24, 28, 29

[2] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein. Square attack: a query-efficient black-box adversarial attack via random search. In *The European Conference on Computer Vision (ECCV)*, 2020. 30

[3] A. Athalye, N. Carlini, and D. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning (ICML)*, 2018. 3, 22, 29

[4] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan. A theory of learning from different domains. *Machine learning*, 79(1):151–175, 2010. 4

[5] A. T. Bui, T. Le, Q. H. Tran, H. Zhao, and D. Phung. A unified wasserstein distributional robustness framework for adversarial training. In *International Conference on Learning Representations (ICLR)*, 2022. 28

[6] Q.-Z. Cai, C. Liu, and D. Song. Curriculum adversarial training. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, 2018. 7

[7] Y. Carmon, A. Raghunathan, L. Schmidt, J. C. Duchi, and P. S. Liang. Unlabeled data improves adversarial robustness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 1, 3

[8] F. Croce and M. Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning (ICML)*, 2020. 3, 8, 9, 28, 29, 30

[9] F. Croce, M. Andriushchenko, V. Sehwag, E. Debenedetti, N. Flammarion, M. Chiang, P. Mittal, and M. Hein. Robustbench: a standardized adversarial robustness benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. 2, 9, 25

[10] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2, 3, 4, 10, 15, 17, 18, 19

[11] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2020. 2, 10

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 8, 24

[13] T. DeVries and G. W. Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 3, 10

[14] Y. Dong, K. Xu, X. Yang, T. Pang, Z. Deng, H. Su, and J. Zhu. Exploring memorization in adversarial training. In *International Conference on Learning Representations (ICLR)*, 2022. 28

[15] L. Fan, S. Liu, P.-Y. Chen, G. Zhang, and C. Gan. When does contrastive learning preserve adversarial robustness from pretraining to finetuning? *Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021. 6

[16] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems (NeurIPS)*, 27, 2014. 1

[17] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015. 1, 2, 30

[18] S. Gowal, C. Qin, J. Uesato, T. Mann, and P. Kohli. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *arXiv preprint arXiv:2010.03593*, 2020. 2, 3, 4

[19] S. Gowal, S.-A. Rebuffi, O. Wiles, F. Stimberg, D. A. Calian, and T. A. Mann. Improving robustness using generated data. *Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021. 1

[20] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015. 24

[21] D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations (ICLR)*, 2019. 9

[22] D. Hendrycks*, N. Mu*, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan. Augmix: A simple method to improve robustness and uncertainty under data shift. In *International Conference on Learning Representations (ICLR)*, 2020. 5, 10

[23] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 2020. 1

[24] J. Howard and S. Gugger. Fastai: a layered api for deep learning. *Information*, 11(2):108, 2020. 8, 24

[25] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, Proceedings of Machine Learning Research. PMLR, 2015. 6

[26] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. G. Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018. 3, 6

[27] Z. Jiang, T. Chen, T. Chen, and Z. Wang. Robust pre-training by adversarial contrastive learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 6

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems (NeurIPS)*, 2012. 2

[29] A. Krizhevsky et al. Learning multiple layers of features from tiny images. 2009. 8, 24

[30] T. Li, Y. Wu, S. Chen, K. Fang, and X. Huang. Subspace adversarial training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 7

[31] A. Madry, A. Makelov, L. Schmidt, T. Dimitris, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018. 1, 3, 8, 26, 27, 30

[32] A. Merchant, B. Zoph, and E. D. Cubuk. Does data augmentation benefit from split batchnorms. *ArXiv*, abs/2010.07810, 2020. 6

[33] T. Pang, X. Yang, Y. Dong, H. Su, and J. Zhu. Bag of tricks for adversarial training. *International Conference on Learning Representations (ICLR)*, 2021. 1, 2, 24, 27

[34] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. Practical black-box attacks against machine learning. In *Proceedings of the ACM Asia Conference on Computer and Communications Security (ACM ASIACCS)*, 2017. 2

[35] R. Rade. PyTorch implementation of uncovering the limits of adversarial training against norm-bounded adversarial examples, 2021. URL https://github.com/imrahulr/adversarial_robustness_pytorch. 10, 24, 27, 28

[36] R. Rade and S.-M. Moosavi-Dezfooli. Reducing excessive margin to achieve a better accuracy vs. robustness trade-off. In *International Conference on Learning Representations (ICLR)*, 2022. 8, 24, 28, 29

[37] S.-A. Rebuffi, S. Gowal, D. A. Calian, F. Stimberg, O. Wiles, and T. A. Mann. Data augmentation can improve robustness. *Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021. 3, 4, 10, 15, 27

[38] L. Rice, E. Wong, and J. Z. Kolter. Overfitting in adversarially robust deep learning. In *International Conference on Machine Learning (ICML)*, 2020. 1, 2, 3, 8

[39] L. Schmidt, S. Santurkar, D. Tsipras, K. Talwar, and A. Madry. Adversarially robust generalization requires more data. *Advances in neural information processing systems (NeurIPS)*, 31, 2018. 1, 3

[40] A. Shaeiri, R. Nobahari, and M. H. Rohban. Towards deep learning models resistant to large perturbations. *arXiv preprint arXiv:2003.13370*, 2020. 7

[41] C. Sitawarin, S. Chakraborty, and D. A. Wagner. Improving adversarial robustness through progressive hardening. *ArXiv*, abs/2003.09347, 2020. 7

[42] G. Sriramanan, S. Addepalli, A. Baburaj, and R. Venkatesh Babu. Guided Adversarial Attack for Evaluating and Enhancing Adversarial Defenses. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3, 4, 7, 8, 10, 19, 20, 23, 24, 26, 30

[43] G. Sriramanan, S. Addepalli, A. Baburaj, and R. Venkatesh Babu. Towards Efficient and Effective Adversarial Training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 6, 7, 8, 23, 26, 27

[44] D. Stutz, M. Hein, and B. Schiele. Relating adversarially robust generalization to flat minima. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 2, 5

[45] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2013. 1

[46] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, 2016. 19

[47] H. Wang and Y. Wang. Self-ensemble adversarial training for improved robustness. In *International Conference on Learning Representations (ICLR)*, 2022. 28

[48] H. Wang, T. Chen, S. Gui, T.-K. Hu, J. Liu, and Z. Wang. Once-for-all adversarial training: In-situ tradeoff between robustness and accuracy for free. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 6

[49] H. Wang, C. Xiao, J. Kossaifi, Z. Yu, A. Anandkumar, and Z. Wang. Augmax: Adversarial composition of random augmentations for robust training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 6

[50] D. Wu, S.-T. Xia, and Y. Wang. Adversarial weight perturbation helps robust generalization. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3, 5, 6, 8, 15, 20, 21, 24, 27, 28, 29

[51] C. Xie and A. Yuille. Intriguing properties of adversarial training at scale. In *International Conference on Learning Representations (ICLR)*, 2020. 6

[52] C. Xie, M. Tan, B. Gong, J. Wang, A. L. Yuille, and Q. V. Le. Adversarial examples improve image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 6

[53] C. Xie, M. Tan, B. Gong, A. Yuille, and Q. V. Le. Smooth adversarial training. *arXiv preprint arXiv:2006.14536*, 2020. 3

[54] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 3, 10

[55] S. Zagoruyko and N. Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 24

[56] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations (ICLR)*, 2018. 10

[57] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning (ICML)*, 2019. 1, 3, 5, 6, 7, 8, 15, 21, 22, 23, 27, 28, 29

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section-7
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] There are no known negative societal impacts of our work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...

(a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] https://github.com/val-iisc/DAJAT.

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Appendix-F.2 and details further specified in ReadMe of https://github.com/val-iisc/DAJAT

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Table:18

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Details on compute: Table-19, Details on resources: Appendix- F.2

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes] See Appendix- F.1

(b) Did you mention the license of the assets? [Yes] See Appendix- F.1

(c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We share the code for the proposed approach

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] We use only publicly released and open-sourced datasets and codes

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] See Appendix-F.1

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Details on the proposed defense DAJAT

The algorithm of the proposed approach is presented in Algorithm-1. In every training iteration, multiple augmentations are considered for every image x_i (L7). We consider one base augmentation and T complex augmentations. The base augmentation consists of Pad and Crop followed by Horizontal Flip, while the complex augmentations are a combination of AutoAugment [10] and the base augmentations. The attack generation for each augmentation (L8-L13) is similar to the ACAT algorithm discussed in Section-E.2. The DAJAT loss (L16) is a combination of the TRADES loss [57] (L17) on each augmentation, and a Jensen-Shannon (JS) divergence term between all augmentations. The JS divergence is a combination of KL divergence terms with respect to the average probability vector as shown below.

$$\mathrm{JSD}(f_\theta(x_{i,\mathrm{base}}), f_\theta(x_{i,\mathrm{auto(1)}}), \ldots, f_\theta(x_{i,\mathrm{auto(T)}}) = \frac{1}{T+1}\big\{\mathrm{KL}(f_\theta(x_{i,\mathrm{base}}, M) + \mathrm{KL}(f_\theta(x_{i,\mathrm{auto(1)}}, M) + \cdots + \mathrm{KL}(f_\theta(x_{i,\mathrm{auto(T)}}, M)\big\} \quad (6)$$

where M is defined as below,

$$M = \frac{1}{T+1}\big\{f_\theta(x_{i,\mathrm{base}}) + f_\theta(x_{i,\mathrm{auto(1)}}) + \cdots + f_\theta(x_{i,\mathrm{auto(T)}})\big\} \quad (7)$$

As shown in Table-5, the JS-divergence term improves accuracy on clean samples and training convergence by enabling the joint learning of representations across different augmentations. The model weights are perturbed to $\tilde{\theta}$ by maximizing the TRADES loss on the base augmentations alone within the constraint set $\mathcal{M}(\theta)$ (L18). This constraint set is chosen such that $||\tilde{\theta}_l|| \leq \gamma \cdot ||\theta_l||$ for any layer l. The network $\tilde{\theta}$ is updated using an SGD step to minimize the overall loss $\mathcal{L}_{\mathrm{DAJAT}}(\tilde{\theta})$ (L20). The model weights are further offset by δ_{DAJAT} which is the adversarial weight perturbation at θ.

B Augmentations

While existing works hypothesize that augmentations changing the low-level statistics of images cannot effectively improve adversarial training [37], we show in this work that with the use of split Batch-normalization layers and JS divergence term between different augmentations, it is indeed possible to obtain significant gains using augmentations that modify the low-level statistics of images as well. In this work, we use an existing augmentation strategy, AutoAugment to obtain an improvement in performance using the proposed training algorithm DAJAT. AutoAugment uses Proximal Policy Optimization to find the set of policies that can yield optimal performance on a given dataset. It consists of 25 unique sub-policies for a given dataset, where each sub-policy is a combination of two augmentations chosen from a set of pre-defined augmentations in series. The pre-defined augmentations include the spatial transformations - shear, rotation and translation, and augmentations that cause changes in low-level statistics of images - color, posterize, solarize, brightness, contrast, sharpness, autocontrast, equalize and invert.

We visualize some of the augmentations generated using AutoAugment and Base augmentations (Pad+Crop, Horizontal Flip) in Fig.5 and Fig.6 respectively. It can be noted that multiple augmentations generated using AutoAugment are significantly more diverse than the augmentations generated using the base augmentations. The use of multiple diverse augmentations leads to improved generalization on test set as discussed in Sec.4. Further we visualize the perturbations generated using the 2-step attack with KL-divergence loss on CIFAR-10 images without augmentations, with Base augmentations and with AutoAugment on Trades-AWP [50] model trained at $\varepsilon = 8/255$ in Fig.7, Fig.8 and Fig.9 respectively. For plotting, one image is selected at random from each of the ten classes. Because of the increased diversity amongst the images on using AutoAugment, we observe more diversity in the perturbations as well (Fig.9) when compared to the Base augmentations (Pad+Crop, Horizontal Flip) (Fig.8) and No augmentations (Fig.7) case.

B.1 Distinction between Simple and Complex Augmentations

We term the augmentations that preserve low-level features of images as simple augmentations, and those that modify the same as complex augmentations. To distinguish between simple and complex augmentations, we do not use the difference between two images in pixel-space, since this would incorrectly show that simple changes like horizontal-flip and crop are far apart. Instead, we use metrics that better capture low-level features at pixel and patch levels. This can be measured at a pixel-level using MSE between color histograms, and at a patch-level using patch-wise MSE. To compute patch-wise MSE between two images x_1 and x_2 , for every 8×8 patch in x_1 we find the nearest patch in x_2 and a horizontal flip of x_2 , and compute an average across all patches in x_1. We report the mean and standard deviation of this value across all images in the test set. We show the pair-wise distances between three sets of images (Unaugmented, Pad+Crop+HFlip, AutoAugment) in Table-7. Lower value indicates that the images are more similar. We note that Pad+Crop+HFlip augmentations have the

Algorithm 1 Diverse Augmentation based Joint Adversarial Training (DAJAT)

Input: Network f_θ, Training Dataset $\mathcal{D} = \{(x_i, y_i)\}$, Adversarial Threat model: ℓ_∞ bound of radius ε, number of epochs E, Maximum Learning Rate LR_{max}, M training mini-batches of size n, number of attack steps S, Cross-entropy loss ℓ_{CE}, Weight perturbation constraint $\mathcal{M}(\theta)$, Number of augmented images using autoaugment T, coefficient of KL divergence term β

for $epoch = 1$ **to** E **do**

  $\varepsilon_{\text{asc}} = epoch \cdot \varepsilon / E$

  $\text{LR} = 0.5 \cdot \text{LR}_{max} \cdot (1 + cosine((epoch - 1)/E \cdot \pi))$

  for $iter = 1$ **to** M **do**

    for $i = 1$ **to** n (in parallel) **do**

      for $a \in \{\text{base}, \text{auto}(1), \ldots, \text{auto(T)}\}$ **do**

        for $steps = 1$ **to** S **do**

          $\delta = 0.001 \cdot \mathcal{N}(0, 1)$

          $\delta = \delta + \varepsilon_{\text{asc}} \cdot \text{sign}\left(\nabla_\delta \text{KL}(f_\theta(x_{i,a}) || f_\theta(x_{i,a} + \delta))\right)$

          $\delta = Clamp\left(\delta, -\varepsilon_{\text{asc}}, \varepsilon_{\text{asc}}\right)$

          $\widetilde{x}_{i,a} = Clamp\left(x_{i,a} + \delta, 0, 1\right)$

        end for

      end for

    end for

$$\mathcal{L}_{\text{DAJAT}}(\theta) = \frac{1}{T+1} \cdot \frac{1}{n} \sum_{i=1}^{n} \left\{ \mathcal{L}_{\text{TR}}(\theta, (x_i, \tilde{x}_i)_{\text{base}}, y_i) + \sum_{t=1}^{T} \mathcal{L}_{\text{TR}}(\theta, (x_i, \tilde{x}_i)_{\text{auto}}, y_i) \right\} + \frac{1}{n} \sum_{i=1}^{n} \left\{ \text{JSD}(f_\theta(x_{i,\text{base}}), f_\theta(x_{i,\text{auto}(1)}), \ldots, f_\theta(x_{i,\text{auto(T)}})) \right\}$$

    where, $\mathcal{L}_{\text{TR}}(\theta, (x, \tilde{x}), y) = \mathcal{L}_{\text{CE}}(f_\theta(x), y) + \beta \cdot \text{KL}(f_\theta(x) || f_\theta(\tilde{x}))$

    $\tilde{\theta} = \underset{\theta \in \mathcal{M}(\theta)}{\text{argmax}} \frac{1}{n} \sum_{i=1}^{n} \left\{ \mathcal{L}_{\text{TR}}(\theta, (x_i, \tilde{x}_i)_{\text{base}}, y_i) \right\}$

    $\delta_{\text{DAJAT}} = \tilde{\theta} - \theta$

    $\theta = \tilde{\theta} - \text{LR} \cdot \nabla_{\tilde{\theta}}(\mathcal{L}_{\text{DAJAT}}(\tilde{\theta})) - \delta_{\text{DAJAT}}$

  end for

end for

advantage of being more similar to the distribution of unaugmented images that are expected during inference, while AutoAugment transformed images are farther away from the unaugmented images. AutoAugment consists of several augmentations of varying complexity levels, and may contain augmentations of similar complexity as the Base augmentations (Pad+Crop+HFlip) as well. This is reflected in the higher variance in pair-wise distances corresponding to AutoAugment.

Table 7: Distinction between Simple and Complex Augmentations in pixel space, in terms of MSE between Histograms and MSE between Patches.

Image pairs	Complexity	MSE between Histograms	MSE between Patches
Base (Pad+Crop+HFlip), Unaugmented	Simple	133.60 $\pm$ 94.05	43.68 $\pm$ 23.37
AutoAugment, Unaugmented	Complex	289.25 $\pm$ 405.11	51.39 $\pm$ 24.23

DAJAT allows separate function mappings for augmentations that resemble the inference-time distribution (Pad+Crop+HFlip), and those that lead to better diversity (AutoAugment). Base augmentations have low variation, and are similar to the distribution of unaugmented images, which is important to obtain performance gains using the batch-norm layer corresponding to these base augmentations during inference. On the other hand, the high variance of AutoAugment based transformations helps in improving the robust generalization of the overall model.

B.2 Distribution shift of Natural and Adversarial images due to Augmentations

We compare the distribution shift between (natural-augmented, natural-unaugmented) pairs and (adversarial-augmented, adversarial-unaugmented) pairs. We consider two types of distances between image pairs: low-level (MSE between histograms/patches) and feature-level (FID). In terms of low-level distances, we expect

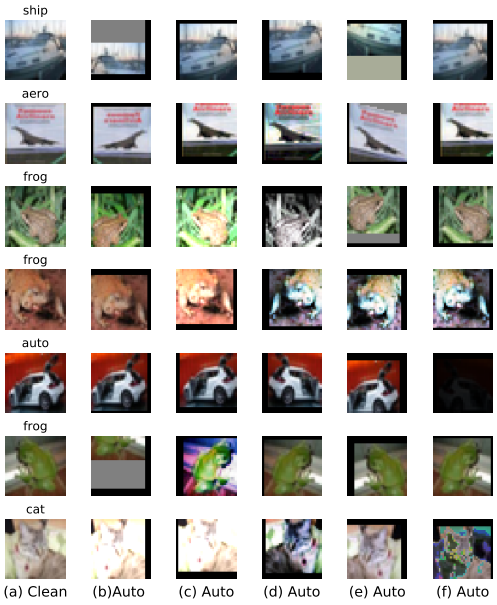


Figure 5: CIFAR-10 images in (a), along with respective random augmentations generated using AutoAugment [10] shown in columns (b-f)

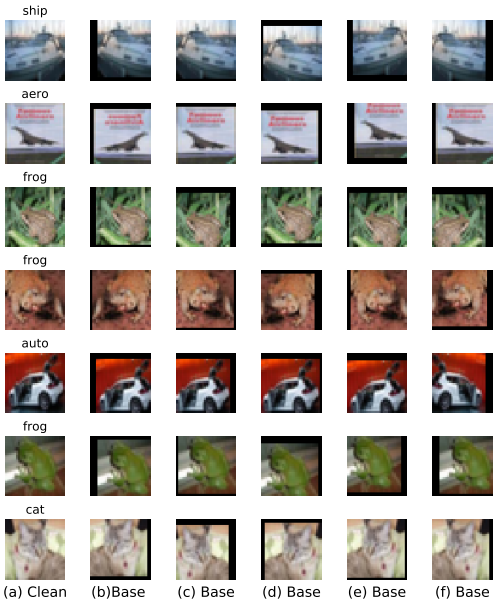


Figure 6: CIFAR-10 images in (a), along with respective random augmentations generated using Base augmentations (Pad+Crop and Horizontal Flip) shown in columns (b-f)

the distances between (natural-augmented, natural-unaugmented) pairs and the corresponding (adversarial-augmented, adversarial-unaugmented) pairs to be similar, since the perturbed images are only an ε away from natural images. However, as seen in Fig.7, 8 and 9, the perturbations of Pad+Crop+HFlip look similar to the perturbations of unaugmented images, while the perturbations of AutoAugment based images look different from those of unaugmented images. This is a result of larger pixel-level differences between the (natural-AutoAugment, natural-unaugmented) image pairs when compared to (natural-PadCrop, natural-unaugmented) image pairs, which serves as a more diverse initialization for the attack. The difference in the absolute perturbations results in a larger distance in feature space (Fréchet Inception Distance or FID) as shown in Table-8.

Table 8: FID between Augmented and Unaugmented images with Simple and Complex augmentations

Image pairs	**FID between Natural image pairs**	**FID between Adversarial image pairs**
Base (Pad+Crop+HFlip), Unaugmented	24.02	33.41
AutoAugment, Unaugmented	37.62	43.75

For better clarity, we summarize our findings in Table-9. The higher feature level distance between (adversarial-AutoAugment, adversarial-unaugmented) image pairs when compared to (natural-AutoAugment, natural-unaugmented) image pairs translates to higher $\frac{1}{2}d_{\mathcal{F}\Delta\mathcal{F}}(s,t)$ in Eq.1. Based on Conjecture-1(ii), unless this difference is accounted for, complex augmentations cannot improve the performance of adversarial training.

Table 9: Summary of pixel-level and feature-level distances between Augmented and Unaugmented image pairs for Natural and Adversarial Images. Base refers to the augmentations Pad+Crop+HFlip.

Natural/ Adversarial	**Distributions**	**Low-level distance**	**Feature-level distance**
Natural images	Base, Unaugmented	Low	Low
	Autoaugment, Unaugmented	High	Medium
Adversarial images	Base, Unaugmented	Low	Medium
	Autoaugment, Unaugmented	High	**High**

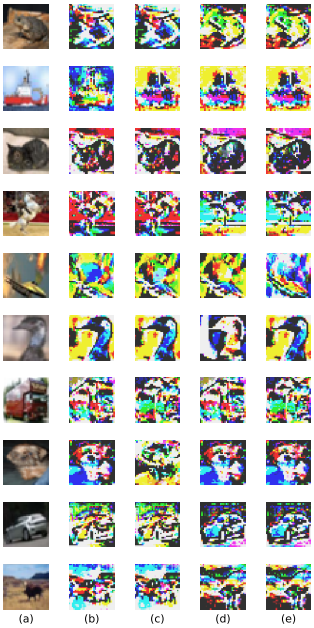


Figure 7: CIFAR-10 images in (a), along with perturbations of respective images generated using without any augmentation shown in columns (b-e). The attack is generated at $\varepsilon = 8/255$ and the corresponding perturbation's magnitude is scaled up for better visibility

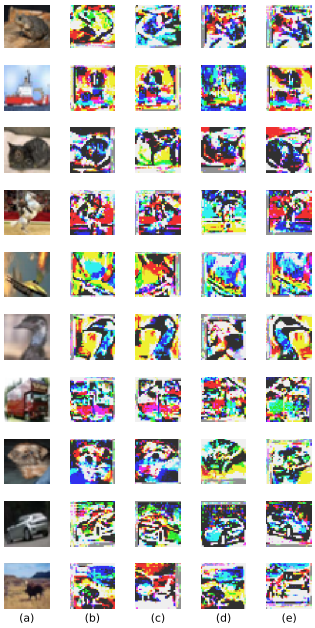


Figure 8: CIFAR-10 images in (a), along with perturbations of respective random augmentations generated using Base augmentations (Pad+Crop and Horizontal Flip) shown in columns (b-e). The attack is generated at $\varepsilon = 8/255$ and the corresponding perturbation's magnitude is scaled up for better visibility

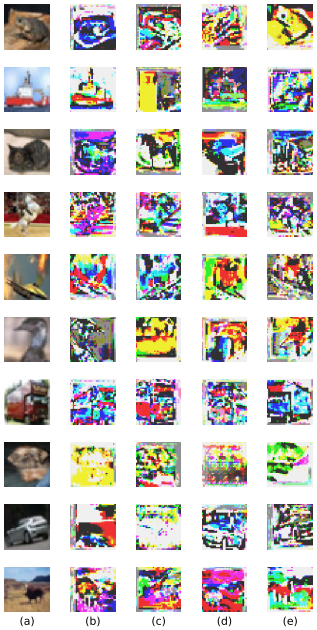


Figure 9: CIFAR-10 images in (a), along with perturbations of respective random augmentations generated using AutoAugment [10] shown in columns (b-e).The attack is generated at $\varepsilon = 8/255$ and the corresponding perturbation's magnitude is scaled up for better visibility

B.3 Justification on the choice of Simple and Complex Augmentation pipeline

In the proposed method, we use two sets of augmentations - Pad+Crop+HFlip and AutoAugment. The second set of augmentations (complex augmentations) consists of an autoaugment based transformation followed by the base augmentations (Pad+Crop+HFlip). This has two implications - firstly, this ensures that the complexity of these augmentations is always greater than or equal to the base augmentations. Secondly, since AutoAugment returns the unaugmented image as well, with a certain probability (0.22 for CIFAR-10 policy), the base augmentations form a subset of the complex augmentations. This trend is indeed reflected in the distribution of pair-wise feature-level similiarities (Cosine similarity between features obtained from an Inception-V3 network) between the following pairs: (Unaugmented, Pad+Crop+HFlip (PCHf)) and (Unaugmented, AutoAugment+PCHf) as shown in Fig.10(a).

When AutoAugment alone is applied, a large fraction of images have a very high cosine similarity, while others have a more spread out distribution as shown in Fig.10(b). When Pad+Crop+HFlip is applied in series after AutoAugment, the distribution of these images shifts to the left, leading to an overlap in the two distributions. Even with such a large overlap the method works well because, the role of the "complex" batch-norm layer is to allow the learning of a function that minimizes empirical risk across a wide distribution of data. While the test distribution may be different from these augmentations, learning from diverse data is known to prevent overfitting and improve generalization. However, since the task of adversarial training is inherently hard, and the objective of minimizing loss on a wider distribution of data makes the task harder, we observe a drop in overall accuracy. The use of a separate batch-norm layer for "simple" augmentations allows the network to specialize on a select subset that is close to the distribution of test set images, and has a low variance. While the diversity of simple augmentations is low, it is sufficient to learn the batch-norm statistics and affine parameters which constitute 0.05% of all parameters, while majority of the parameters are learned using both distributions, resulting in low overfitting.

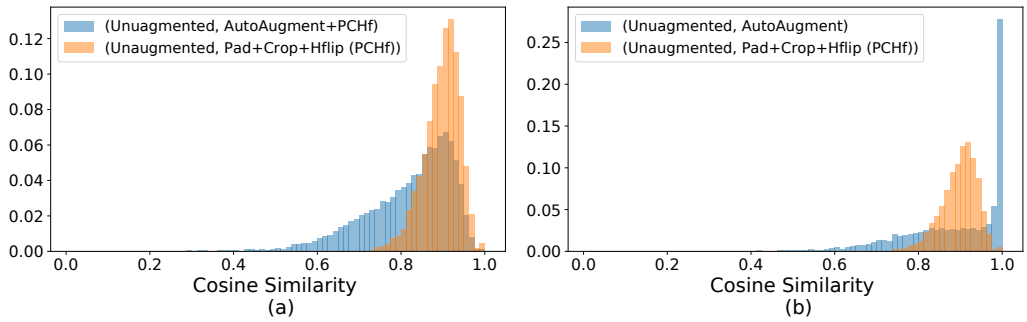


Figure 10: Normalized histogram of pair-wise cosine similarity between the features obtained using a pretrained Inception-V3 [46] model for different augmentations w.r.t. the respective input images. Histograms are plotted for the following pairs, (a) (Unaugmented, AutoAugment+Pad+Crop+Horizontal flip) and (Unaugmented, Pad+Crop+Horizontal flip), (b) (Unaugmented, AutoAugment) and (Unaugmented, Pad+Crop+Horizontal flip). The full test set of CIFAR-10 (10k images) is used for the plots.

B.4 Ablations

We show the impact of some of the important categories of augmentations individually in Table-10.

Table 10: Impact of individual augmentations within AutoAugment [10] on the proposed defense DA-JAT. Performance on CIFAR-10 dataset with ResNet-18 architecture is reported. Robust evaluations are done against GAMA attack [42]

	Augmentation		**Base + Augmentation + JS**	
Augmentation	**Clean Acc**	**Robust Acc**	**Clean Acc**	**Robust Acc**
AutoAugment	82.54	48.11	**84.94**	51.23
AutoAugment (without spatial augs)	**83.70**	48.80	84.94	51.40
Brightness	82.11	46.42	84.56	50.94
Sharpness	81.78	49.85	84.30	50.54
Color-Balance	82.31	**49.87**	84.39	**51.48**
Contrast	82.45	46.54	84.21	50.77

The robust accuracy improves when the spatial augmentations - shear, rotation and translation are not used for adversarial training. Amongst the augmentations that modify the low-level statistics of images, change in color balance gives maximum benefit. Further, it can be noted that although some augmentations such as change in brightness and contrast lead to a drop in robust accuracy when used directly, the use of the same augmentations in the proposed framework results in a significant boost in accuracy. Therefore there is no need to use carefully selected augmentations with the proposed framework.

C Split Batch-Normalization

We perform three ablation experiments - first by training with separate running statistics and common affine parameters, second by training with common running statistics and separate affine parameters, and third without using split batch-norm at all, as shown in Table-11. Robust Accuracy is reported against the GAMA attack. As shown in the Table-11, E2 and E3 (having either split running statistics or split affine parameters) perform similar to the proposed approach, where separate running statistics and affine parameters are used. This shows that our method learns a different function mapping for both augmentations, and this can be realized by having different running statistics or different affine parameters or using a combination of both. We further note that the use of a single batch-norm layer for both augmentations (E4) degrades the results significantly.

We compare the average cosine similarity of the running statistics across training iterations for both our method, and the case where we have separate running statistics and common affine parameters in Fig.11. We note that the scale and trend of the average cosine similarity is similar in both cases. In the case where common affine parameters are used, the gains in results w.r.t. single batch-norm case can be attributed to the small drop in cosine similarity over training. This indicates that small changes in these running statistics can indeed lead to a large impact in the overall results (as shown in Table-11). We further verify this by noting the large difference in

Table 11: **Split Batch-Normalization:** Impact of using common/ split running statistics and affine parameters. Using a combination of separate running statistics and affine parameters works the best.

Method	Clean Accuracy	Robust Accuracy
[E1] split running statistics + split affine parameters (Ours)	**88.90**	**57.22**
[E2] split running statistics + common affine parameters	88.61	56.91
[E3] common running statistics + split affine parameters	88.86	57.01
[E4] common running statistics + common affine parameters (single Batch-Norm)	89.08	53.86

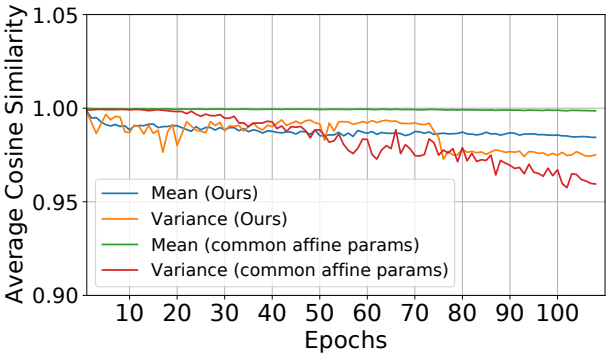


Figure 11: **Average Cosine Similarity (averaged over all the layers of WideResNet-34-10) of Running Mean and Variance** across training epochs. For the proposed approach DAJAT, and an ablation experiment that uses common batch-normalization affine parameters, the scale of cosine similarities are similar, indicating that small changes in running mean and variance can indeed have a large impact on model outputs.

performance of the models when different batch-norm parameters are used for training, in Table-12. Robust Accuracy is reported against the GAMA attack.

D Improvements of DAJAT in the low data regime

We compare the performance of the proposed DAJAT defense (using Base, 2*AA) with TRADES-AWP [50] 2-step baseline across different sizes of the CIFAR-10 training dataset on ResNet-18 architecture in Fig.12. We consider class balanced dataset for each case. We note that across different settings, the proposed approach achieves 5-9% gains in clean accuracy and an average gain of 1.3% in robust accuracy. The gain in clean accuracy in the low data regime is significantly high (8.93%) highlighting the need for augmentations in improving Adversarial Training performance, specifically in real-world settings where the training data is low.

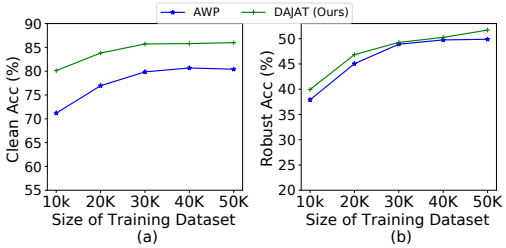


Figure 12: **Improvements of DAJAT in the low data regime:** Performance of proposed approach DAJAT (Base, 2*AA) when compared to the TRADES-AWP [50] 2-step baseline on ResNet-18 architecture and CIFAR-10 dataset. The proposed approach achieves improvements across different sizes of training dataset, with higher gains in the low data regime.

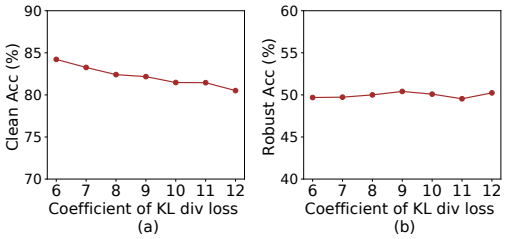


Figure 13: **Performance of ACAT across variation in β:** Performance of the proposed defense ACAT against variation in the coefficient of KL divergence loss between the clean and adversarial samples β. Higher β leads to improved robust accuracy (against GAMA [42] attack) at the cost of clean accuracy. ResNet-18 architecture and CIFAR-10 dataset are used.

Table 12: **Impact of using different batch-normalization layers** (Pad+Crop+HFlip vs AutoAugment) **at inference time.**

Method	Batch-norm (Inference)	Clean Acc	Robust Acc
Split running statistics + split affine parameters (Ours)	Pad+Crop+HFlip	**88.90**	**57.22**
Split running statistics + split affine parameters (Ours)	AutoAugment	76.17	44.45
Split running statistics + common affine parameters	Pad+Crop+HFlip	88.61	56.91
Split running statistics + common affine parameters	AutoAugment	78.69	45.41

Table 13: Fixed vs ascending ε Adversarial training on ResNet-18 and WRN-34-10 models with maximum $\varepsilon = 8/255$ on CIFAR-10 dataset with 2 step attack generation during training. Impact of gradient masking is higher for larger capacity models. This is effectively mitigated with the use of Ascending ε Adversarial Training

Architecture	Method	Clean(last)	Robust(last)	Clean(best)	Robust(best)	Clean(last - best)	Robust(last - best)
ResNet-18	Fixed ε AT	80.63	49.63	80.82	49.61	-0.19	0.02
ResNet-18	Ascending ε AT	**82.41**	**50.00**	**82.57**	**49.91**	-0.16	0.09
WRN-34-10	Fixed ε AT	86.69	44.87	**86.83**	54.76	-0.14	-9.89
WRN-34-10	Ascending ε AT	**86.71**	**55.58**	86.30	**55.46**	0.41	0.12

E Details on ACAT

E.1 Motivation: Instability of 2-step Adversarial Training

Ascending perturbation radius helps in mitigating gradient masking when lesser steps are used for attack generation. We present results of ACAT when compared to Fixed constraint AT (TRADES-AWP with 2 attack steps) on CIFAR-10 dataset using $\varepsilon = 8/255$ with ResNet-18 and WideResNet-34-10 model architectures in Table-13. Robust Accuracy is reported against the GAMA attack. Best accuracy is computed using PGD-20 attack, which is not very reliable. Hence, in some cases, best epoch may have a slightly lower accuracy when compared to the last epoch. On ResNet-18, we observe that the difference between last and best epochs for both methods is very low. However, on WideResNet-34-10, we observe the phenomenon of gradient masking in Fixed Constraint AT, with robust accuracy dropping by around 10% towards the end of training. We note that the difference between last and best epochs is very low for ACAT even on WideResNet-34-10. The main motivation of using an ascending perturbation radius is to stabilize training and prevent the onset of gradient masking.

E.2 ACAT training algorithm

The algorithm for the proposed ACAT defense is presented in Algorithm-2. We consider an ℓ_∞ threat model of perturbation radius ε. The perturbation bound for attack generation ε_{asc} is linearly increased from 0 to ε during training (L3). The learning rate follows a cosine schedule across the training epochs as shown in L4. The attack generation (L7-L12) is done for 2 iterations and follows the TRADES [57] settings. Initially Gaussian noise of magnitude 0.001 is added to every pixel. The KL divergence loss between the clean and perturbed images is maximized using the perturbation step size ε_{asc}. Further, the perturbation is clipped to remain within the threat model in every iteration.

As shown in L14, the TRADES-AWP [50] loss is used for adversarial training. The loss $\mathcal{L}_{\text{AWP}}(\theta)$ is maximized with respect to θ to find the perturbed model weights $\tilde{\theta}$ within the constraint set $\mathcal{M}(\theta)$ (L15). Further, the model training is done at $\tilde{\theta}$, after which the adversarial weight pertubation δ_{AWP} is subtracted to offset the perturbation in weights (L17). The defense ACAT does not use any additional training hyperparameters when compared to the TRADES-AWP defense. We vary the hyperparameter β to obtain optimal results. We show the trend of model performance across variation in β in Section-F.2.

E.3 Analysis of the proposed ACAT defense

Conjecture-2: We hypothesize that given a Neural Network f_{θ_ε} that minimizes the TRADES loss [57] within a maximum perturbation radius of ε, and has sufficient smoothness in weight space within an ℓ_2 radius ψ around θ_ε, there exist $\varepsilon' > \varepsilon$ and a model $f_{\theta_{\varepsilon'}}$ where $||\theta_{\varepsilon'} - \theta_\varepsilon|| \leq \psi$, such that $f_{\theta_{\varepsilon'}}$ has a lower TRADES loss within ε' when compared to f_{θ_ε}.

Justification: We first consider Adversarial Training within a perturbation bound ε using the TRADES-AWP loss shown below.

$$\mathcal{L}_{\text{AWP},\theta,\varepsilon} = \max_{\hat{\theta}\in\mathcal{M}(\theta)} \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_{\text{CE}}(f_{\hat{\theta}}(x_i), y_i) + \beta \cdot \max_{\tilde{x}_i \in \mathcal{A}_\varepsilon(x_i)} \text{KL}(f_{\hat{\theta}}(x_i)||f_{\hat{\theta}}(\tilde{x}_i)) \tag{8}$$

$$\theta_\varepsilon = \underset{\theta}{\text{argmin}}\, \mathcal{L}_{\text{AWP},\theta,\varepsilon} \tag{9}$$

The model f_{θ_ε} which is the minimizer of the above loss, achieves an optimal trade-off between the cross-entropy loss on clean samples (clean loss, $\mathcal{L}_{\text{clean}}$) and weighted KL divergence between clean and adversarial images [57] (adversarial loss, $\mathcal{L}_{\text{adv}}$).

$$\mathcal{L}_{\text{clean},\theta} = \sum_{i=1}^{N} \mathcal{L}_{\text{CE}}(f_\theta(x_i), y_i), \quad \mathcal{L}_{\text{adv},\theta,\varepsilon} = \beta \cdot \sum_{i=1}^{N} \text{KL}(f_\theta(x_i)||f_\theta(\tilde{x}_i)), \;\; \tilde{x} \in \mathcal{A}_\varepsilon(x) \tag{10}$$

Since the loss attains a minima at θ_ε, a direction of further reduction in both clean and adversarial losses does not exist (Eq.9). Gradient descent on the adversarial loss at θ_ε to parameter θ' results in a reduction in adversarial loss at the cost of a higher clean loss.

$$\theta' = \theta_\varepsilon - \eta \cdot \nabla \mathcal{L}_{\text{adv},\theta_\varepsilon,\varepsilon} \tag{11}$$

$$\mathcal{L}_{\text{adv},\theta',\varepsilon} < \mathcal{L}_{\text{adv},\theta_\varepsilon,\varepsilon}, \quad \mathcal{L}_{\text{clean},\theta'} > \mathcal{L}_{\text{clean},\theta_\varepsilon} \tag{12}$$

We assume that $\mathcal{L}_{\text{clean},\theta_\varepsilon}$ increases by a rate $\gamma_{\text{CE},\theta_\varepsilon}$ and $\mathcal{L}_{\text{adv},\theta_\varepsilon,\varepsilon}$ decreases by a rate $\gamma_{\text{KL}_\varepsilon,\theta_\varepsilon}$ in the direction of gradient descent of the adversarial loss ($-\nabla\mathcal{L}_{\text{adv},\theta_\varepsilon,\varepsilon}$). Since θ_ε is a minimizer of loss,

$$\mathcal{L}_{\text{AWP},\theta',\varepsilon} > \mathcal{L}_{\text{AWP},\theta_\varepsilon,\varepsilon} \implies \gamma_{\text{CE},\theta_\varepsilon} = \gamma_{\text{KL}_\varepsilon,\theta_\varepsilon} + \alpha, \;\; \alpha > 0 \tag{13}$$

Since the Cross-entropy loss is minimized at x, it attains a minima at x within the pixel neighborhood.

$$\mathcal{L}_{\text{CE}}(f_{\theta_\varepsilon}(x), y)) < \mathcal{L}_{\text{CE}}(f_{\theta_\varepsilon}(x+\epsilon), y) \tag{14}$$

Since the model f_θ is adversarially robust within ε, the loss surface is locally smooth within the open interval $(-\varepsilon, \varepsilon)$. The adversarial loss, which measures the KL divergence of the adversarial image w.r.t. the clean image is least at x and increases monotonically in the ε-neighborhood for adversarially robust models [3]. Therefore, adversarial loss at $\varepsilon' = \varepsilon + \delta$ for a small enough δ would be marginally higher than the loss at ε.

$$\mathcal{L}_{\text{adv},\theta_\varepsilon,\varepsilon'} > \mathcal{L}_{\text{adv},\theta_\varepsilon,\varepsilon} \tag{15}$$

Based on the local smoothness in weight space as well, the same property holds at any θ' in the gradient descent direction of $\mathcal{L}_{\text{adv},\theta_\varepsilon,\varepsilon}$ (satisfying Eq.11) such that $||\theta' - \theta_\varepsilon|| \leq \psi$, as shown below:

$$\mathcal{L}_{\text{adv},\theta_\varepsilon,\varepsilon'} = \mathcal{L}_{\text{adv},\theta_\varepsilon,\varepsilon} + \delta_{\theta_\varepsilon} \tag{16}$$

$$\mathcal{L}_{\text{adv},\theta',\varepsilon'} = \mathcal{L}_{\text{adv},\theta',\varepsilon} + \delta_{\theta'} \tag{17}$$

Since the adversarial loss at θ' is lower than the same at θ_ε, the loss surface at θ' has a lower local Lipschitz constant when compared to θ_ε leading to the following:

$$\delta_{\theta'} < \delta_{\theta_\varepsilon} \implies \mathcal{L}_{\text{adv},\theta',\varepsilon'} - \mathcal{L}_{\text{adv},\theta',\varepsilon} < \mathcal{L}_{\text{adv},\theta_\varepsilon,\varepsilon'} - \mathcal{L}_{\text{adv},\theta_\varepsilon,\varepsilon} \tag{18}$$

Rearranging terms,

$$\mathcal{L}_{\text{adv},\theta_\varepsilon,\varepsilon'} - \mathcal{L}_{\text{adv},\theta',\varepsilon'} > \mathcal{L}_{\text{adv},\theta_\varepsilon,\varepsilon} - \mathcal{L}_{\text{adv},\theta',\varepsilon} \tag{19}$$

For small enough gradient descent step size η, the above can be related to the rate of reduction in the adversarial loss as shown below:

$$\gamma_{\text{KL}_{\varepsilon'},\theta_\varepsilon} = \gamma_{\text{KL}_\varepsilon,\theta_\varepsilon} + \alpha', \;\; \alpha' > 0 \tag{20}$$

Since the Cross-entropy loss does not depend on ε, from Eq.13 and Eq.20,

$$\gamma_{\text{CE},\theta_\varepsilon} - \gamma_{\text{KL}_{\varepsilon'},\theta_\varepsilon} = \alpha - \alpha' \tag{21}$$

Therefore, the rate of increase in $\mathcal{L}_{\text{AWP},\theta_\varepsilon,\varepsilon'}$, or $\alpha - \alpha'$, is less than the rate of increase in $\mathcal{L}_{\text{AWP},\theta_\varepsilon,\varepsilon}$, or α. For small enough α, $\alpha' > \alpha$, so that the overall loss $\mathcal{L}_{\text{AWP},\theta_\varepsilon,\varepsilon'}$ decreases. Based on this, for small enough

Algorithm 2 Ascending Constraint Adversarial Training (ACAT)

Input: Network f_θ, Training Dataset $\mathcal{D} = \{(x_i, y_i)\}$, Adversarial Threat model: ℓ_∞ bound of radius ε, number of epochs E, Maximum Learning Rate LR_{max}, M training mini-batches of size n, Cross-entropy loss ℓ_{CE}, Weight perturbation constraint $\mathcal{M}(\theta)$, coefficient of KL divergence term β
for $epoch = 1$ **to** E **do**
  $\varepsilon_{\text{asc}} = epoch \cdot \varepsilon / E$
  $\text{LR} = 0.5 \cdot \text{LR}_{max} \cdot (1 + cosine((epoch - 1)/E \cdot \pi))$
  for $iter = 1$ **to** M **do**
    for $i = 1$ **to** n (in parallel) **do**
      for $steps = 1$ **to** 2 **do**
        $\delta = 0.001 \cdot \mathcal{N}(0, 1)$
        $\delta = \delta + \varepsilon_{\text{asc}} \cdot \text{sign}\left(\nabla_\delta \text{KL}(f_\theta(x_i) || f_\theta(x_i + \delta))\right)$
        $\delta = Clamp\left(\delta, -\varepsilon_{\text{asc}}, \varepsilon_{\text{asc}}\right)$
        $\widetilde{x}_i = Clamp\left(x_i + \delta, 0, 1\right)$
      end for
    end for
    $\mathcal{L}_{\text{AWP}}(\theta) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_{\text{CE}}(f_\theta(x_i), y_i) + \beta \cdot \text{KL}(f_\theta(x_i) || f_\theta(\tilde{x}_i))$
    $\tilde{\theta} = \underset{\theta \in \mathcal{M}(\theta)}{\text{argmax}}\, \mathcal{L}_{\text{AWP}}(\theta)$
    $\delta_{\text{AWP}} = \tilde{\theta} - \theta$
    $\theta = \tilde{\theta} - \text{LR} \cdot \nabla_{\tilde{\theta}}(\mathcal{L}_{\text{AWP}}(\tilde{\theta})) - \delta_{\text{AWP}}$
  end for
end for

η, $\exists\, \theta_{\varepsilon'}\ s.t.\ ||\theta_{\varepsilon'} - \theta_\varepsilon|| \leq \psi$ and $\mathcal{L}_{\text{AWP},\theta_{\varepsilon'},\varepsilon'} < \mathcal{L}_{\text{AWP},\theta_\varepsilon,\varepsilon'}$. Hence, it is possible to move to $\theta_{\varepsilon'}$ which has a lower overall loss than the loss at θ_ε. ■

Table 14: **Integrating ACAT with different loss formulations** on the CIFAR-10 dataset with WideResNet-34-10 architecture. Robust accuracy is reported against the GAMA attack [42].

Method	# Attack Steps	Clean Acc	Robust Acc	Time per epoch (seconds)
TRADES-AWP	2	85.49	41.62	412
ACAT (with TRADES loss)	2	86.71	55.58	412
NuAT2-WA	2	86.32	55.08	334
ACAT (with NuAT loss)	2	86.19	**55.91**	530
GAT2-WA	2	87.36	50.24	267
ACAT (with GAT loss)	2	**87.79**	54.70	396

E.4 Integrating ACAT with other efficient training methods

The proposed ACAT defense uses the KL divergence loss between clean and adversarial images, similar to the TRADES adversarial training algorithm [57]. We present results by integrating the proposed ACAT defense with losses from existing efficient adversarial training algorithms [42, 43] in Table-14. We obtain a significant boost in performance over the respective baselines, when we use ACAT with GAT [42] and TRADES [57] losses, and a marginal boost when integrated with the NuAT defense [43]. The adversarial weight perturbation step in the proposed defense results in an increase in computational time when compared to the respective baselines. We choose the KL divergence based loss for both proposed defenses ACAT and DAJAT since it results in an optimal trade-off between performance and training time.

E.5 Analysis of the effect of augmentations on ACAT

We analyze how using different combinations of augmentations affect ACAT in Table-15. From these results we can say that using hard augmentations like AutoAugment and a mix of base and AutoAugment each on 50% on the batch leads to a degradation in the performance of the model as compared to using simple augmentations like Pad+Crop+Horizontal Flip.

Table 15: Impact of augmentations: Performance (%) of ACAT models on Base augmentations and AutoAugment (Auto). Clean and robust accuracy against GAMA attack [42] are reported. The use of AutoAugment results in ~ 1.5 - 2% drop in robust accuracy. The use of Base Augmentations alone (Pad+Crop+HFlip) gives the best overall performance on the unaugmented test set.

Architecture	Train set	No-Aug (Clean)	No-Aug (Robust)	AutoAug (Clean)	AutoAug (Robust)
ResNet-18	No-Augmentation	73.50	43.64	44.98	18.50
	Base (Pad+Crop+HFlip)	82.41	50.00	63.79	37.07
	AutoAugment	**82.54**	48.11	**76.40**	**43.22**
	Base (50% batch) + AutoAugment (50% batch)	81.15	**50.01**	70.89	40.93
WideResNet-34-10	No-Augmentation	80.34	47.98	54.66	26.44
	Base (Pad+Crop+HFlip)	86.71	**55.58**	68.24	40.83
	AutoAugment	**86.80**	53.99	**82.64**	**48.98**
	Base (50% batch) + AutoAugment (50% batch)	86.52	54.15	75.90	45.56

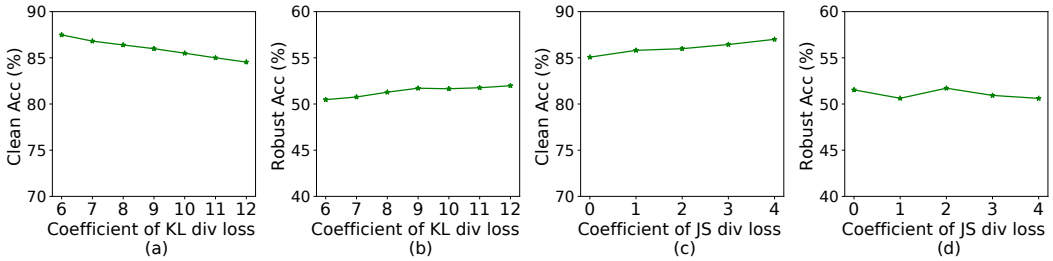


Figure 14: **Performance of DAJAT across variation in hyperparameters:** Performance of the proposed defense DAJAT (Base, 2*AA) on ResNet-18 architecture and CIFAR-10 dataset across variation in (a,b) coefficient of KL divergence loss between softmax outputs of clean and adversarial images. (c,d) coefficient of the JS divergence loss between the representations of different augmentations. Robust accuracy against GAMA [42] attack is shown.

F Details on Experiments and Results

F.1 Details on Datasets

We perform evaluations of the proposed defenses ACAT and DAJAT on the CIFAR-10, CIFAR-100 [29] and ImageNette-10 [24, 12] datasets, comprising of 10, 100 and 10 classes respectively. The resolution of images in the CIFAR-10 and CIFAR-100 datasets is 32x32, while it is 128x128 in ImageNette dataset. We use ResNet-18 [20] and WideResNet-34-10 [55] architectures for the CIFAR-10 and CIFAR-100 datasets and ResNet-18 architecture for ImageNette. While CIFAR-10 is the most popular dataset used for benchmarking adversarial defenses, CIFAR-100 has a larger number of classes with one-tenth the number of images in each class, making it a more challenging setting. ImageNette dataset is used to show the performance of the proposed method on higher resolution images. We consider an ℓ_∞ threat model with $\varepsilon = 8/255$ for the primary evaluations across all datasets. We use TRADES-AWP [50] [2] as the base code for most of our analysis. For analysis in Table-23 we use HAT [36] [3] and OAAT [1][4] as the base codes. For cutmix analysis in Table-21 we use [35] [5] as the base code. The license for each of these codes are available on their respective github repositories. Since the datasets used in this work are public, commonly used for research purposes and do not contain any objectionable content, we find no need to take any consent from the authors of these datasets.

F.2 Details on Training Settings

The training for the baseline methods and the proposed approach is done for 110 epochs as is common in literature [33]. We use SGD optimizer along with cosine learning rate schedule for the proposed method with a momentum of 0.9. We fix the maximum learning rate to 0.2 and weight decay to 5e-4 for the proposed approach across all datasets and model architectures. For the baselines, we use the training settings from the official codes released by the authors. We performed all the experiments on two NVidia V100 GPUs, two RTX-2080 GPUs and one RTX-3090 GPU. We use a validation split of 1000 images from the train dataset both for CIFAR-10 and

[2] https://github.com/csdongxian/AWP
[3] https://github.com/imrahulr/hat
[4] https://github.com/val-iisc/OAAT
[5] https://github.com/imrahulr/adversarial_robustness_pytorch

Table 16: Performance gains obtained using DAJAT on CIFAR-100 using **larger capacity models**

Training Algorithm	Architecture	Clean Accuracy	Robust Accuracy (GAMA)	Robust Accuracy (AutoAttack)
TRADES-AWP	WRN-34-10	62.73	29.92	29.59
DAJAT (**Ours**)		68.74	31.58	31.30
Gains using DAJAT		**6.01**	**1.66**	**1.71**
TRADES-AWP	WRN-34-20	63.12	30.15	29.83
DAJAT (**Ours**)		70.49	32.91	32.55
Gains using DAJAT		**7.37**	**2.76**	**2.72**

Table 17: Comparison of DAJAT with Fixed-ε and Ascending-ε schedules on ResNet-18 and WideResNet-34-10 architectures on CIFAR-10 and CIFAR-100 datasets.

Model architecture	Dataset	Epsilon schedule	Clean Acc	Robust Acc	Clean Acc (with Weight Averaging)	Robust Acc (with Weight Averaging)
ResNet-18	CIFAR-10	Fixed ε	86.57	51.17	86.25	51.44
	CIFAR-10	Varying ε (DAJAT)	86.13	51.37	85.99	51.71
	CIFAR-100	Fixed ε	66.03	26.24	65.50	26.55
	CIFAR-100	Varying ε (DAJAT)	66.50	27.12	66.84	27.61
WRN-34-10	CIFAR-10	Fixed ε	91.46	31.01	90.09	47.30
	CIFAR-10	Varying ε (DAJAT)	89.12	56.98	88.90	57.22
	CIFAR-100	Fixed ε	71.04	19.90	70.60	25.97
	CIFAR-100	Varying ε (DAJAT)	68.82	30.75	68.74	31.58

CIFAR-100 images, except for the results reported in Table-4, where we use the full dataset to compare against the state-of-the-art defenses on the RobustBench leaderboard [9].

For the proposed approaches ACAT and DAJAT (Base, 2*AA), we vary the value of β to achieve an optimal trade-off between the clean and adversarial accuracy. As shown in Fig.13 and Fig.14(a,b), as we increase β, robust accuracy improves and clean accuracy degrades initially, with a saturating trend in robust accuracy at higher values of β. For the ResNet-18 model and CIFAR-10 dataset, the optimal value of β is 8 and 9 for ACAT and DAJAT respectively.

We further fix the value of β to the optimal setting of 9 and vary the coefficient of the JS divergence term in Fig.14(c,d). This term leads to a boost in the clean accuracy at the cost of a slight degradation in the robust accuracy. The optimal setting of the coefficient of JS divergence term is 2 in the given setting (Base, 2*AA) and ranges from 1 to 3 across all settings, datasets and model architectures.

We consider the Base, 2*AA as the main setting of DAJAT since its computational complexity matches with that of TRADES-AWP. However, we show the result of Base, 3*AA as well to highlight that the performance improves with a further increase in diversity. In this case, since the weight of the Base augmentations is considerably low in the overall loss (L16 in Algorithm-1), we give a weight of 1/3 to the TRADES loss on base augmentations and 2/3 to the TRADES loss on the AutoAugment based images. This mimics the setting of Base, 2*AA in terms of loss weighting, while introducing additional diversity due to the presence of a larger number of complex augmentations, yielding a small boost in performance.

We share the optimal set of hyperparameters across all datasets and model architectures in the ReadMe file that is submitted along with our Code in the Supplementary submission.

F.3 Performance on Larger Capacity Models

We find that the performance gains obtained using DAJAT are indeed higher on larger capacity models. In Table-16, we present results on CIFAR-100 dataset, on WideResNet-34-10 and WideResNet-34-20 model architectures using 110 epochs of training. We note that the improvements in clean accuracy increase from 6.01% to 7.37%, while the improvements in robust accuracy against AutoAttack increase from 1.71% to 2.72%.

F.4 Impact of Ascending Perturbation Radius in DAJAT

We present an ablation of DAJAT without ascending perturbation radius (DAJAT+Fixed-ε), where attacks are constrained within a fixed perturbation bound of 8/255 during training. Table-17 shows results on CIFAR-10 and CIFAR-100 datasets with WideResNet-34-10 and ResNet-18 architectures. Robust Accuracy is reported against the GAMA attack. As seen in the first half of the table, on ResNet-18 architecture, there is no gradient masking even with DAJAT+Fixed-ε. However, on the WideResNet-34-10 architecture, DAJAT+Fixed-ε results in a large drop in robust accuracy due to gradient masking. Although the use of weight averaging improves results, its

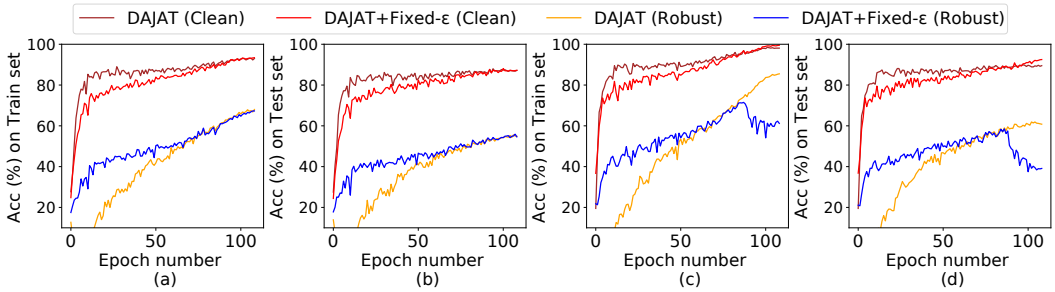


Figure 15: **Train and test Accuracy plots of DAJAT and DAJAT+Fixed-ε to show impact of varying ε training schedule:** The train and test plots of DAJAT are compared with an ablation experiment where a fixed ε training schedule is used (DAJAT+Fixed-ε). Clean and Robust accuracy against PGD-20 [31] attack are plotted on CIFAR-10 dataset using ResNet-18 (a,b) and WideResNet-34-10 (c,d) architectures. Although DAJAT+Fixed-ε is stable at lower model capacities (ResNet-18), there is a sudden drop in the robust accuracy due to Gradient masking on WideResNet-34-10 (c,d) after epoch 87, indicating the need for using varying ε schedule in DAJAT. Evaluations are done using an attack perturbation bound of $\varepsilon = 8/255$.

Table 18: **Variance across reruns:** Variation in performance (%) of the proposed defenses ACAT and DAJAT on CIFAR-10 dataset and ResNet-18 architecture across three reruns. Based on the low standard deviation across runs, we note that both approaches are stable across reruns. Robust accuracy is evaluated against GAMA PGD-100 attack [42].

	ACAT		**DAJAT**	
	Clean Acc	**Robust Acc**	**Clean Acc**	**Robust Acc**
Run-1	82.41	50.00	86.13	51.37
Run-2	82.49	50.08	85.96	51.51
Run-3	82.54	50.12	85.94	51.48
Average	82.48	50.07	86.01	51.45
Standard Deviation	0.07	0.06	0.1	0.07

robust accuracy is still lower by around 5-10% when compared to DAJAT. The phenomenon of gradient masking in DAJAT+Fixed-ε can also be observed in Fig.15, where its robust accuracy suddenly drops after epoch 87 accompanied by an increase in clean accuracy on CIFAR-10 with WideResNet-34-10 architecture. We note that even with respect to the best epoch accuracy on DAJAT+Fixed-ε, we obtain an improvement of 2.57% on robust accuracy using DAJAT.

F.5 Variance across reruns

We present the variation across three reruns for the proposed defenses ACAT and DAJAT (Base, 2*AA) on CIFAR-10 dataset and ResNet-18 architecture in Table-18. Since weight averaging is known to improve the stability of the base method [43], we present results without the use of weight averaging to highlight the inherent variance of the base algorithm. The standard deviation of robust accuracy across reruns is low (0.06-0.07) across both approaches indicating stability of the proposed method. It can be noted that the standard deviation of clean accuracy in the proposed defense DAJAT is slightly higher (0.1) due to the randomness in the complex augmentations, which impacts clean accuracy more than the robust accuracy. Overall, we note that the standard deviation of clean and robust accuracy for both proposed defences ACAT and DAJAT is low .

F.6 Analysis on Compute, Flops and Performance

We present the FLOPs, number of parameters and training time on a single Nvidia RTX-3090 GPU along with our results when compared to the TRADES-AWP baseline on the CIFAR-10 dataset for ResNet18 and WRN-34-10 models in Table-19. Robust accuracy is reported against AutoAttack. We discuss our observations below:

Table 19: **Comparison of compute, FLOPs (per iteration) and performance** of the proposed approaches DAJAT and ACAT when compared to TRADES-AWP on CIFAR-10 dataset.

Model	Method	gigaFLOPS (GFLOPS)		Parameters (Million)		Train Time	Clean Acc	Robust Acc
		Training	Inference	Training	Inference	/epoch(sec)	(%)	(%)
ResNet18	TRADES-AWP	2707.6	71.251	11.174	11.174	299	80.47	49.87
	ACAT (**Ours**)	997.52	71.251	11.174	11.174	108	82.41	49.80
	DAJAT (**Ours**)	2422.5	71.251	11.184	11.174	264	85.99	51.48
WRN-34-10	TRADES-AWP	32417	853.08	46.160	46.160	1633	85.10	55.87
	ACAT (**Ours**)	11943	853.08	46.160	46.160	472	86.71	55.36
	DAJAT (**Ours**)	29005	853.08	46.183	46.160	1381	88.90	56.96

Table 20: **Comparison of the proposed augmentation scheme with CutMix based augmentations [37]:** Performance (%) of the proposed defense DAJAT (Base, 2*AA) when compared to the use of CutMix based augmentation proposed by Rebuffi et al. [37] against PGD 40-step attack [31]

Method	Clean Acc	Robust Acc (PGD-40)
TRADES [57]	84.72	56.92
Rebuffi et al. [37]	87.24	57.60
TRADES-AWP [50]	85.35	59.13
DAJAT (**Ours**) (Base, 2*AA)	**88.90**	**60.97**

- FLOPs and number of parameters during inference are identical among the three training methods (TRADES-AWP, ACAT, DAJAT), since we use only a single batch-normalization layer (corresponding to Pad+Crop) during inference. As expected, these values are higher for WideResNet-34-10 model architecture when compared to ResNet-18.
- Since we use split batch-norm layers in DAJAT, the number of parameters increases by 0.05% during training, while it remains the same during inference.
- We compute FLOPs during training by considering that a single backward pass requires twice the number of FLOPs when compared to a forward pass. We also provide the number of forward and backward passes in each method for reference.
- Using ACAT, we achieve 63% reduction in FLOPs (training) and training time when compared to the TRADES-AWP baseline, while achieving 1.6-1.9% higher clean accuracy and comparable robust accuracy.
- The use of ACAT strategy in the proposed DAJAT defense enables us to achieve similar computational complexity as TRADES-AWP defense, while obtaining gains in performance. Using DAJAT, we achieve 10% reduction in FLOPs (training) and training time, while obtaining 3.8-5.5% higher clean accuracy and 1-1.6% higher robust accuracy.

F.7 Comparison against CutMix based augmentation

While we compare the performance of the proposed approach against various base adversarial training algorithms [50, 31, 33, 43] in the main paper, we additionally compare with a recent augmentation scheme that uses CutMix augmentations [37] to improve performance in this section. The authors of [37] show a significant boost in performance using 400 epochs of training and large model architectures. However, to ensure a fair comparison, we report the result of 110 epochs of training on WideResNet-34-10 architecture and CIFAR-10 dataset that has been shared by the authors with us upon request. We report the PGD 40-step accuracy as shared by the authors. As shown in Table-20, we obtain a significant boost in performance over the CutMix based augmentation as well as the TRADES-AWP [50] baseline using the proposed defense DAJAT.

Additionally, contrary to the claims by Rebuffi et al. [37], we show that it is indeed possible to effectively use augmentations that modify the low-level statistics of images for obtaining improved performance in Adversarial Training by using the proposed defense DAJAT.

As present in the github repository [6] of [37] we note that naively using cutmix doesn't give good results as shown in Table-6, therefore as suggested we use [35] as the base code and incorporate cutmix into it. We present the results for 200 epochs training with learning rate drop of 0.1 at 100 and 150 epochs, using the PreActResNet-18 model with Swish Activation and batch size of 128 in Table-21(C1). We observe significantly improved results

[6] https://github.com/deepmind/deepmind-research/tree/master/adversarial_robustness/pytorch

Table 21: Performance (%) by using [35] on CIFAR-10 dataset with Preact-ResNet18 model with Swish Activation trained using varying epsilon schedule and cosine learning rate unless specified otherwise.

Method	**Clean**	**GAMA**
[C1]: Cutmix - step schedule + fixed eps	81.67	49.18
[C2]: Cutmix	**83.34**	49.24
[C3]: Ours(Base, Cutmix)	82.67	51.99
[C4]: Ours(Base, Cutmix, Cutmix)	83.05	**52.22**
[C5]: C1 with Relu	81.03	46.6
[C6]: C1 with Weight decay for BN	70.66	36.36

Table 22: Performance of the proposed defense DAJAT when compared to some concurrent works on **CIFAR-10 and CIFAR-100 datasets** for ResNet18 and WideResNet-34-10 models. Robust evaluations are performed on Auto-Attack [8].

	CIFAR-10, ResNet-18		**CIFAR-10, WRN-34-10**		**CIFAR-100, ResNet-18**		**CIFAR-100, WRN-34-10**	
Method	**Clean**	**AA @ 8/255**	**Clean**	**AA @ 8/255**	**Clean**	**AA @ 8/255**	**Clean**	**AA @ 8/255**
AWP [50]	81.99	51.45	85.36	56.17	59.88	25.81	62.73	29.59
HAT [36]	85.63	49.54	86.21	51.46	59.19	23.26	59.95	24.55
SEAT [47]	83.7	51.3	86.44	55.67	56.28	**27.87**	-	-
SEAT+Cutmix [47]	81.53	49.1	84.81	56.03	-	-	-	-
TRADES + TE [14]	83.86	49.77	-	-	59.35	25.27	-	-
UDR + TRADES [5]	84.4	49.9	84.93	54.45	-	-	-	-
DAJAT (**Ours**)	**85.71**	**52.50**	**88.71**	**57.81**	**65.45**	27.69	**68.75**	**31.85**

as compared to Table-6 on using [35] as the base code. We observe that the key differences in [35] as compared to [57] are:

- Use of swish activation function in the PreActResNet18 model
- Weight decay not used for batch normalization layers

To study the impact of these changes, we investigate the use ReLU instead of Swish activation (Table-21(C5)) and the use of weight decay for all the parameters of the model including the batch normalization layers (Table-21(C6)). In both cases, we observe a significant drop with respect to C1. Thus based on this ablation, the use of swish activation, and avoiding weight decay for batch normalization layers seems important to obtain benefits using Cutmix.

Further we incorporate linearly increasing varying epsilon schedule along with cosine learning rate schedule and get improved results in Table-21(C2). Next we incorporated our method DAJAT with C2 and present the results in Table-21(C3,C5), where we can observe significant gains over C2, thus showing the effectiveness of DAJAT.

F.8 Comparison of DAJAT with Concurrent Works

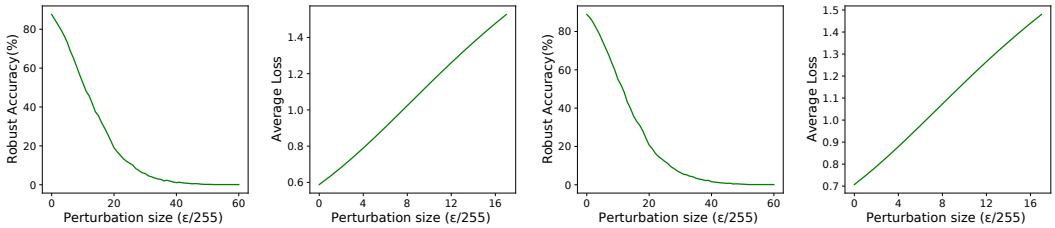
Here we compare the proposed approach DAJAT trained for 200 epochs with recent works that appeared at ICLR 2022. The comparison of the proposed method DAJAT with AWP [50], HAT [36], Self Ensemble Adversarial Training (SEAT) [47] with (SEAT+cutmix) and without cutmix (SEAT), Unified distributional robustness for TRADES (UDR + TRADES) [5], temporal ensembling with TRADES (TRADES + TE) [14] on CIFAR10 and CIFAR100 datasets for ResNet18 and WideResNet-34-10 models is shown in Table-22. Since these are very recent works, we only present results reported in the paper, and leave the remaining entries in the table blank.

F.9 Combining the proposed approach with different adversarial training methods

We explore combining the proposed defense DAJAT with some existing methods like [50], [1] and [36] in Table-23. We observe that combining DAJAT with all three existing works leads to significant gains both in clean as well as adversarial accuracies (AA, 8/255), especially on CIFAR-100 where the number of images per class is low. Although OAAT [1] shows improved results over AWP [50], combining DAJAT with OAAT leads to further gains of $\sim 1.5\%$ in both clean and adversarial accuracy on CIFAR10 and $1-1.5\%$ gains in both clean and adversarial accuracy on CIFAR100. Further, since OAAT [1] claims to achieve robustness at larger epsilon bounds, we evaluate using Auto-Attack at $\varepsilon = 16/255$. Using OAAT+DAJAT we observe gains over OAAT on AutoAttack with $\varepsilon = 16/255$ as well, which further confirms the effectiveness of DAJAT. Finally we combine DAJAT with HAT [36] and we observe consistent gains over HAT [36] on all models and datasets. While HAT

Table 23: Performance (%) of DAJAT when **combined with other Adversarial training methods**, OAAT [1] and HAT [36] on **CIFAR-10 and CIFAR-100** with 110 epochs of training. Robust evaluations are performed on Auto-Attack(AA) [8] at $\varepsilon = 8/255$ and $16/255$.

	CIFAR-10, ResNet-18			CIFAR-10, WRN-34-10			CIFAR-100, ResNet-18			CIFAR-100, WRN-34-10		
Method	**Clean**	**AA, 8/255**	**AA, 16/255**	**Clean**	**AA, 8/255**	**AA, 16/255**	**Clean**	**AA, 8/255**	**AA, 16/255**	**Clean**	**AA, 8/255**	**AA, 16/255**
AWP [57, 50]	80.47	49.87	**19.23**	85.10	55.87	**23.27**	59.88	25.81	8.28	62.73	29.59	**11.04**
AWP+DAJAT	**85.99**	**51.48**	16.33	**88.90**	**56.96**	19.73	**66.84**	**27.32**	**8.97**	**68.74**	**31.30**	9.91
OAAT [1]	80.24	50.88	22.05	85.67	55.93	24.05	61.70	26.77	9.91	65.73	30.35	12.01
OAAT+DAJAT	**82.05**	**52.21**	**22.78**	**86.22**	**57.64**	**24.56**	**62.50**	**28.47**	**10.67**	**66.03**	**31.15**	**12.67**
HAT [36]	85.63	49.54	14.96	86.21	51.46	**16.76**	59.19	23.26	6.96	59.95	24.55	7.13
HAT+DAJAT	**86.68**	**51.47**	**16.38**	**86.71**	**53.85**	16.50	**62.78**	**26.49**	**8.72**	**64.88**	**27.37**	**8.71**



(a) ACAT (Accuracy) (b) ACAT (Loss) (c) DAJAT (Accuracy) (d) DAJAT (Loss)

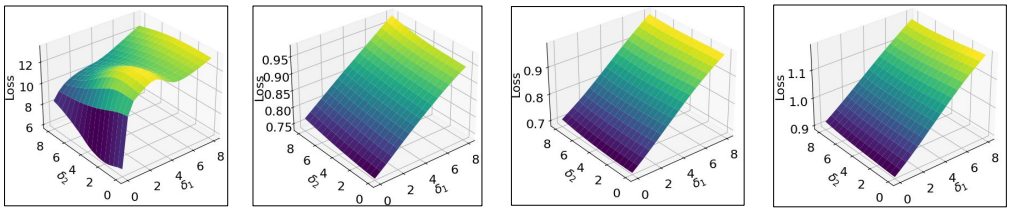
Figure 16: **Robust Accuracy and Loss on FGSM attack against variation in perturbation size:** (a,c) Robust accuracy (%) of the proposed defenses ACAT and DAJAT against PGD 7-step attacks across variation in attack perturbation bound. Attacks within larger perturbation bounds are able to bring down the robust accuracy of the model to 0, indicating the absence of gradient masking. (b,d) Cross-entropy loss on FGSM adversarial samples across variation in attack perturbation bound. The linearly increasing trend of loss indicates the absence of gradient masking. The models are trained on CIFAR-10 dataset using ResNet-18 architecture.

[36] proposes to improve the robustness-accuracy trade-off, combining DAJAT with HAT further improves this trade-off and shows gains of $\sim 1\%$ on clean accuracy and $\sim 2\%$ on robust accuracy for CIFAR-10, and $3-5\%$ on clean accuracy and $\sim 3\%$ on robust accuracy for CIFAR-100 dataset.

F.10 Sanity checks to verify the absence of gradient masking

We perform several sanity checks as recommended by Athalye et al. [3] to ensure the absence of gradient masking in the proposed defenses ACAT and DAJAT.

- From Table-24 we note that Black-Box attacks are weaker than White-Box attacks, indicating that the gradients from the model are reliable.
- We further note from Table-24 that attacks with higher number of steps are stronger than those with lower steps. Further, PGD multi-step attacks are stronger than FGSM white-box attacks.
- From Table-25 we note that robust accuracy against targeted and untargeted attacks saturates as the number of attack steps increase from 500 to 1000, indicating that the evaluation is robust.
- We also note from Table-25 that the drop in accuracy with 1000 random restarts is marginal.



(a) Normal Training (b) TRADES-AWP (c) ACAT (**Ours**) (d) DAJAT (**Ours**)

Figure 17: **Loss Surface Plots:** Plot of cross-entropy loss in the local neighborhood of images along the gradient direction (δ_1) and a random direction perpendicular to the gradient (δ_2). The loss surface of the proposed defenses ACAT and DAJAT are smooth similar to the TRADES-AWP defense, indicating the absence of gradient masking.

Table 24: **Evaluation against Black-Box and White-Box FGSM [17] attacks and multi-step PGD attacks [31]:** Performance (%) of the proposed DAJAT (Base + 2*AA) defense on CIFAR-10 dataset with ResNet-18 architecture

Method	**Clean Acc**	**BB FGSM**	**FGSM**	**PGD-20**	**PGD-100**	**PGD-500**
NuAT2-WA	86.32	84.71	63.48	58.09	57.74	57.74
ACAT	86.71	85.29	64.08	58.76	58.64	58.53
TRADES-AWP	85.36	83.93	63.49	59.22	59.11	59.08
DAJAT(Base, 3*AA)	**88.64**	**87.19**	**66.99**	**61.09**	**60.80**	**60.74**

Table 25: **Evaluation against multi-step Targeted and Untargeted PGD attacks [31] with single and multiple random restarts:** Performance (%) of the proposed defense DAJAT (Base, 2*AA) across different datasets with ResNet-18 architecture

	CIFAR-10		**CIFAR-100**		**IN-10**	
Attack	500-step	1000-step	500-step	1000-step	500-step	1000-step
PGD-Targeted (Least Likely Class)	85.01	85.01	66.02	65.98	85.06	85.01
PGD-Targeted (Random Class)	80.56	80.55	63.96	63.96	80.13	80.13
PGD-Untargeted	55.21	55.20	32.89	32.89	65.07	65.07
	1-RR	1000-RR	1-RR	1000-RR	1-RR	1000-RR
PGD 50-step, r-RR	55.30	54.55	32.98	32.09	65.20	65.02

- We note from Fig.16 that an increase in perturbation bound increases the effectiveness of PGD 7-step attacks, and is able to bring down the accuracy of the model to 0 at large bounds. Further, the loss on FGSM samples monotonically increases in the vicinity of the data samples. These trends indicate the absence of gradient masking.
- We present results against AutoAttack [8] in Tables-2 and 3 of the main paper. AutoAttack is an ensemble of several gradient-based attacks and a gradient-free attack Square [2]. The robust accuracy against AutoAttack is similar to the accuracy against gradient-based attack GAMA [42] indicating that gradient-free attacks are not significantly stronger than gradient based attacks.
- We show the loss surface plots of the proposed defenses ACAT and DAJAT in the vicinity of data samples in Fig.17. We note that the loss surface of the proposed defenses is smooth similar to the TRADES-AWP defense, indicating the absence of gradient masking.

We finally compare the robust accuracy against various attacks in Tables-24 and 25 with the robust accuracy against GAMA attack [42] and AutoAttack [8] in Tables-2 and 3 in the main paper. The latter evaluations are significantly stronger, indicating that the evaluation presented in the main paper is robust.