Simple Post-Training Robustness using Test Time Augmentations and Random Forest

Gilad Cohen
Tel Aviv University
giladco1@post.tau.ac.il

Raja Giryes
Tel Aviv University
raja@tauex.tau.ac.il

Abstract

Although Deep Neural Networks (DNNs) achieve excellent performance on many real-world tasks, they are highly vulnerable to adversarial attacks. A leading defense against such attacks is adversarial training, a technique in which a DNN is trained to be robust to adversarial attacks by introducing adversarial noise to its input. This procedure is effective but must be done during the training phase. In this work, we propose Augmented Random Forest (ARF), a simple and easy-to-use strategy for robustifying an existing pretrained DNN without modifying its weights. For every image, we generate randomized test time augmentations by applying diverse color, blur, noise, and geometric transforms. Then we use the DNN's logits output to train a simple random forest to predict the real class label. Our method achieves state-of-the-art adversarial robustness on a diversity of white and black box attacks with minimal compromise on the natural images' classification. We test ARF also against numerous adaptive white-box attacks and it shows excellent results when combined with adversarial training. https://github.com/giladcohen/ARF.

1. Introduction

Deep neural networks (DNNs) achieve cutting edge performance in many problems and tasks. Yet, it has been shown that small perturbations of the network, which in many cases are indistinguishable to a human observer, may alter completely the network output [24, 60]. This phenomenon poses a great risk when using neural networks in sensitive applications and therefore requires a lot of attention.

Many defense techniques were developed to improve DNN's robustness to adversarial attacks. Yet, repeatedly, after the proposal of a new successful defense, a new attack was proposed that found new breeches in the DNNs [8, 61].

An example of a very common and successful strategy for improving DNN robustness is adversarial training [24, 40]. In this approach, adversarial examples are added in the network training process along with the regular examples. It is shown to reduce significantly the network vulnerability to attacks. A major disadvantage of this approach and most of the other existing defense strategies is that they require retraining the network. This puts an additional computational time, which might be significant in some cases, as one needs to update the network frequently to resist novel attacks.

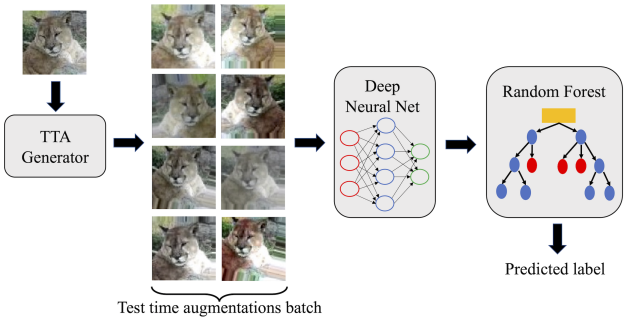

Figure 1. **ARF flow chart**. Test time augmentations are generated and fed into a pretrained DNN. Its logits are then passed to a random forest classifier to predict the class label.

Even in techniques that just fine tune DNNs, there is a need to have an access to all the training data. The same holds for the current leading detection methods that aim at just spotting attacks and alerting about them (without changing the DNN) [15, 36, 39]. Besides storage issues, having access to all data is a problem when a user wants to improve its network robustness to new attacks that were not present during the development of the DNN but does not have access to that data due to privacy or proprietary issues.

Contribution. To mitigate these issues, we propose a novel approach for improving the robustness against adversarial attacks, named **A**ugmented **R**andom **F**orest (ARF), which only requires storage of logits vectors and not the images themselves. Also, it is very simple to use, does not require retraining, and can be employed with any machine learning classifier that produces logits, including an adversarially trained DNN, to improve its robustness. In

addition to ARF, we also introduce a novel white-box attack, A-PGD, that combines the PGD attack [40] with image augmentations. We show that this attack is superior than current state-of-the-art (SOTA) attacks on DNN ensembles.

In our approach, for each input image we generate N Test Time Augmentations (TTAs) as shown in Figure 1. We feed this batch of image transformations to the DNN and collect its logits output. In the training phase (done only once), we fit a simple random forest classifier using logits obtained from both natural and adversarial TTAs. The random forest learns to be robust against adversarial images by training on the entire set of TTAs' logits distribution. In the inference phase, we generate N TTAs for a single image, obtain the DNN logits for it, and pass the logits to the random forest classifier. The inference executes a single forward pass on all the N TTAs at once, and thus is very fast.

We emphasize that the image transformations done in our pre-processing were used before for defense [26, 38, 45, 63]. These TTAs improve classifiers' robustness due to obfuscated gradients, but this was later shown to be fake robustness which can be circumvented using adaptive attacks in white-box settings [2] (see Section 2). Yet, we use TTAs to enrich the input augmentations for the random forest; we find that they improve accuracy on natural (non-adversarial) images and also enhance robustness.

We compare the adversarial robustness of ARF to SOTA baselines on a diverse set of attack strategies and threat models, for CIFAR-10, CIFAR-100, SVHN, and Tiny-ImageNet. ARF always succeeds to enhance the DNN's robustness by many folds, without the need to retrain the network (the random forest training is negligible compared to a neural network training) or store the training data. The best results are obtained when applying ARF with an adversarially robust network. We show that this combination achieves SOTA defense and is robust to adaptive white-box attacks.

2. Related Works

Various attacks and defense techniques have been proposed for DNNs. Defense techniques may be divided into strategies that aim at increasing the network robustness and approaches that try only detecting the adversarial attacks; In this work we focus on the former ones and describe some of them. For a more a comprehensive survey of the existing strategies one may refer to [18, 42, 67].

Adversarial attacks. The core strategy in adversarial attacks is to look for the smallest perturbation of an input that causes the network to change its prediction. The main difference between different existing attacks is the metric used to define the size of the change and the search strategy that is used for finding the perturbation. In addition, attacks may be targeted, i.e., aiming to change the output to a specific given class, or untargeted that just try to flip the network prediction. Another difference is the threat model of the attacks. Black-box adversaries can merely access the network outputs, while white-box adversaries have full access to the architecture and parameters, algorithm used for training and classification, and training data [11].

The fast gradient sign method (FGSM) changes the input in the direction of the gradient of the cross-entropy loss [24]. It is a fast single-step attack that is very easy to deploy. The Jacobian-based saliency map attack (JSMA) aims to find only a selected few input pixels, which induce the largest loss increase [49]. It is stronger but iterative and slow.

Deepfool is a non-targeted attack that searches for the closest decision boundary of the network for the given input example [44]. The work in [9] proposed a novel targeted attack (known as CW), which overcame the distillation defense method that was very successful till then [47]. Their approach was further improved in [8], where they formulated an optimization framework to construct loss functions for attacks that are defense specific. The work in [17] demonstrated using an ensemble of parameter-free attacks.

The work in [5] introduced the Boundary attack, a decision based attack used in black-box settings. Their method only requires the final model prediction and can be employed where the output logits do not exist or inaccessible. A more efficient black-box attack, the square attack [1], used much fewer queries than the Boundary attack, and it was shown to even outperform several gradient-based white-box attacks. These two attacks do not rely on gradient information and can be applied on any machine learning classifier.

In order to evaluate the performance of a novel defense approach, it is not sufficient to check robustness on the above attacks but rather design adaptive attacks to the developed defense [7, 23]. In our work, we evaluate our proposed defense against several tailored adaptive attacks.

Adversarial robustness. Many techniques where proposed to improve the adversarial robustness of DNNs. Some add a regularization during the network training such as penalizing the network input gradients [52] or Jacobian [29] to improve robustness; scaling the gradients in a batch based on their magnitude [54]; penalizing the network output so it has a smaller Lipschitz constant [28]; requiring similarity between logits of pairs of input examples [31]; requiring the linear and convoluional layers in the network to be approximately Parseval tight frames [14]; or using the mixup regularization [69, 71].

Other approaches rely on gradient masking [6, 19, 26, 55]. They make it harder for attacks (especially black-box) to find the gradient direction for producing the adversarial examples. However, masking the model gradient's cannot guarantee robustness against adaptive white-box attacks, as shown in [2]. They proposed BPDA, which estimates masked gradients in the classifier, and replaces them with approximated gradients in the backward pass by replacing any non-differential layer.

Another strategy to improve robustness is adding noise

to the data or perturbations to the network features during training [16, 30, 66]. A different approach performs a k-NN search, perhaps using external datasets or the web, to make a decision on the input [21, 58]. Knowledge distillation was also used to improve robustness [47]. It was improved by using gradient information [12, 48].

A leading method is adversarial (re)training with its many variants [24, 34, 40, 43, 57, 62, 64]. It trains the network using adversarial examples in addition to the regular data and thus improves robustness. Adding unlabeled data in the adversarial training improves performance on the clean data [10, 68], which is deteriorated many times due to the adversarial training.

One disadvantage of adversarial training is that it is computationally demanding. "Free adversarial training" propose an accelerated version [56]. The Virtual Adversarial Training (VAT) work [43] used a regularization term to smooth the output logits distribution of the model within a small environment surrounding the input image. The TRADES approach [70] added a regularizaion term to the cross-entropy loss in the training phase to improve robustness inside the ℓ_p ball $\mathbb{B}_p(x, \epsilon) = \{x' : ||x - x'||_p \leq \epsilon\}$. By adjusting this term one can control the trade-off between the accuracies on the normal and adversarial samples [59].

Unlike ARF, all the above methods require changing the DNN training and cannot be used for a trained network.

Test-time augmentation (TTA). Some works used transformations on the input image to yield a robust classifier [22, 25, 26, 37, 38, 41, 50, 63, 65]. Yet, the work in [3] found that these methods are susceptible to the EoT attack in white-box settings, where the transformation distribution is considered in the attack loss. Later, the BPDA attack was shown to circumvent non-differential transformations as well [2].

The approaches in [4, 26] compute the KL divergence between a pair of augmentations to detect adversarial attacks. The strategy in [53] proposed to utilize TTAs to detect adversarial images. The TTAs were used to aggregate statistics on the input image and detect anomalies associated with adversarial attacks. They showed that in some cases the correct label can also be predicted. Their approach requires an extensive statistical analysis on the dataset and tuning parameters and thresholds. Thus, it is not simple and easy to use with any arbitrary pretrained classifier.

The closest work to us added a random forest after a DNN for improving robustness to adversarial attacks [20]. Unlike ARF, their methodology includes a tedious analysis on the DNN layers. They searched for the "best" layer to start growing the random forest using a manual observation on the relative L_2 distance between original and adversarial samples, whereas ARF simply attaches the output of any learning classifier to a vanilla random forest. They showed robustness on simple MNIST and CIFAR-10 datasets, whereas we use more complex datasets (e.g., Tiny-ImageNet).

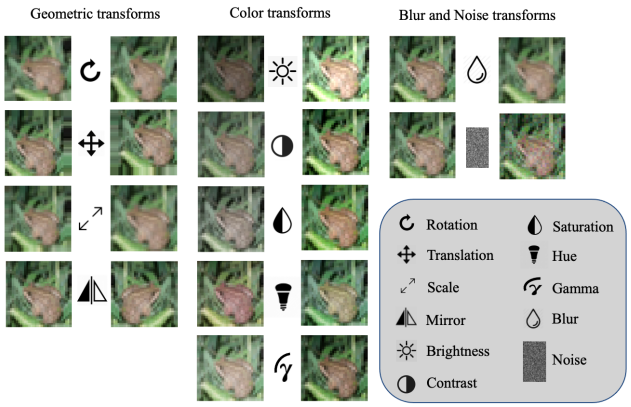

Figure 2. All the transforms used for the test-time augmentation (TTA). The left column illustrates the geometric transforms: Rotation, translation, scaling, and horizontal flips (not used on SVHN). The middle column illustrates the color transforms: Brightness, contrast, saturation, hue, and gamma. The right column illustrates a Gaussian blur and an addition of Gaussian white noise. All the above transforms are randomized to generate N TTAs samples.

3. Method

We turn to present ARF. We start by describing the TTAs generation prior to feeding them to the DNN. Then we show how their logits are used to train the random forest classifier.

3.1. Test-time Augmentations

We hypothesize that even if the adversary succeeds to attack a specific image, the close neighborhood around the image still holds enough information for reverting the predicted (wrong) label back to the correct label. To that end, for each image we generate N TTAs, using a variety of color, geometrical, blur, and noise transforms (see Fig. 2).

The color transforms include: Brightness, contrast, saturation, hue, and gamma; the geometric transforms include: Rotation, translation, scaling, and horizontal flipping; the blur transform convolutes the image with a 2D Gaussian kernel $G_{2D}(u, v; \sigma_b)$ where σ_b is uniformly distributed for every TTA image between 0.001 and a positive constant: $\sigma_b \sim U(0.001, \sigma_{bmax})$. The noise transform adds a white Gaussian noise n to the image, where $n \sim N(0, \sigma)$, and σ is uniformly distributed for every TTA image between 0 and a positive constant σ_{max}: $\sigma \sim U(0, \sigma_{max})$.

All the transforms including their parameters are randomized in test time. More details on the transforms definitions and parameters distributions appear in sup. mat. We chose to apply these transforms because they were shown to improve the classification accuracy significantly in self-supervised and semi-supervised learning [13]. Similarly to them, all the transforms parameters were chosen to alter the image until a human struggles to perceive the images on the dataset. We also added the Gamma transform since it showed small improvement (data not shown).

3.2. TTA Classifier

We generate N randomized TTAs and feed them to the DNN (Figure 1), and collect their logits output (of size N). Formally, we denote x as the original image, the generated TTAs are denoted as $\{x_t[i]\}_{i\in[0,N-1]}$, and the DNN outputs are $\{l[i,c]\}_{i\in[0,N-1]}^{c\in[0,\#\text{classes}-1]}$, where $l[i,c]$ is the logit corresponding to class c of the transformed image $x_t[i]$.

When using only the TTAs for making the prediction, the inferred label is a simple argmax of the logits summation:

$$c_{pred} = \underset{c}{\operatorname{argmax}} \sum_{i=0}^{N-1} l[i,c]. \tag{1}$$

3.3. ARF Classifier

We split the official test set into two: *val* and *test* (see Sec. 4 - Random forest training). Let M be the *val* size. The augmented random forest (ARF) employs the aforementioned DNN logits of *val* to train a random forest classifier. We generate in *val* N TTAs for the normal (unperturbed) images and additional $10N$ TTAs for adversarial images generated using ten generic (non-adaptive) adversarial attacks (see Sec. 4 - Adversarial attacks), denoted by $\{x_t[k,i]\}_{k\in[0,M-1]}^{i\in[0,N-1]}$, $\{x'_t[k,i]\}_{k\in[0,M-1]}^{i\in[0,10N-1]}$, respectively. k indicates the image index in the *val* set, and i is the augmentation index. These TTAs are fed to the DNN and their logits output for the normal and adversarial images are denoted as $\{l[k,i,c]\}_{k\in[0,M-1],i\in[0,N-1]}^{c\in[0,\#\text{classes}-1]}$, $\{l'[k,i,c]\}_{k\in[0,M-1],i\in[0,10N-1]}^{c\in[0,\#\text{classes}-1]}$, respectively, or $\{l[k]\}$ and $\{l'[k]\}$ in short for clarity. We then fit the random forest classifier using the pairs $\{l[k],y[k]\}\bigcup\{l'[k],y[k]\}$ where $y[k]$ is the true label of the image $x[k]$, i.e., it learns to infer correct labels both from regular and adversarial logits.

The random forest training procedure needs to be carried out only once. For every new (unseen) image we generate TTAs, obtain their logits $l[i,c]$ (as in Section 3.2) and feed them to the random forest classifier to predict the class label.

3.4. Adversarial Attacks

To inspect our defense against adversarial images, we employed extensive and diverse attacks in a variety of threat models, and then evaluate them using our ARF classifier and compare them to the robustness obtained using equivalent adversarially trained TRADES/VAT networks, and to an ensemble of networks.

Black-box. A threat model where the adversary has access only to the DNN output, but neither to the DNN nor to the random forest classifier. In this setup we apply targeted Boundary [5] and untargeted Square [1] attacks on the DNN.

Gray-box. In this threat model the adversary has full access the DNN parameters, but is oblivious to the pre-processing (transformation) and post-processing (random forest) defenses. We apply FGSM, JSMA, PGD, Deepfool, and CW on the DNN. All, except Deepfool, are targeted.

Adaptive black-box. In this setting, the adversary does not have information on the DNN and random forest parameters. They can only query the final output (predicted label) of the random forest and perturb the input image without any gradients knowledge. We use the Square attack, which was shown to be more efficient compared to the popular Boundary attack, and achieved SOTA results, even compared to white-box attacks. Also, we set an untargeted setting since this attack excels on it [1].

Adaptive Gray-box. In this threat model the adversary has access to the DNN's parameters and has full knowledge about the distributions of the test time augmentations. The adversary is still oblivious to the post-processing (random forest). We formulate two adaptive attacks for this setting:

1) A-FGSM: This attack applies the FGSM attack on every one of the generated TTAs in $\{x_t[i]\}_{i\in[0,N-1]}$. All the gradients are then averaged and the mean gradient map is added to the original input image. Formally, we define X_t to be the distribution of the generated TTA transforms on an image x. Given a loss function $J(x,y;w)$, where x is the input image, y is the adversarial label and w are the DNN weights, the A-FGSM creates an adversarial image x' by:

$$\begin{aligned} x' &= x + \mathbb{E}_{x_t\sim X_t}\left[\epsilon\cdot\operatorname{sign}\left(\nabla_{x_t}J(x_t,y;w)\right)\right] \\ &= x + \frac{\epsilon}{N}\sum_{i=0}^{N-1}\operatorname{sign}\left(\nabla_{x_t[i]}J(x_t[i],y;w)\right). \end{aligned} \tag{2}$$

2) A-PGD: Similarly to the gradient averaging shown for A-FGSM, this adaptive attack employs PGD but in every iteration it projects the adversarial perturbations after the addition of the *averaged* TTAs gradients. Formally, let δ_k be the perturbation added to input image x in step k and α be the perturbation step size. The vanilla PGD attack is:

$$\delta_{k+1} = \mathcal{P}\left[\delta_k + \alpha\cdot\operatorname{sign}\left(\nabla_{\delta_k}J(x+\delta_k,y;w)\right)\right],$$

where $\mathcal{P}$ is the projection operator, clipping every perturbation inside a ball of interest defined by a given norm $\|\cdot\|$ (we use L_∞). For a general norm $\|\cdot\|$ it simply reads as:

$$\mathcal{P}(\delta) = \begin{cases} \frac{\delta}{||\delta||}\epsilon, & \text{if } ||\delta|| > \epsilon \\ \delta, & \text{otherwise.} \end{cases}$$

Our adaptive PGD attack is defined as:

$$\begin{aligned} \delta_{k+1} &= \mathcal{P}\left[\delta_k + \mathbb{E}_{x_t\sim X_t}\left[\alpha\cdot\operatorname{sign}\left(\nabla_{\delta_k}J(x_t+\delta_k,y;w)\right)\right]\right] \\ &= \mathcal{P}\left[\delta_k + \frac{\alpha}{N}\sum_{i=0}^{N-1}\operatorname{sign}\left(\nabla_{\delta_k}J(x_t[i]+\delta_k,y;w)\right)\right], \end{aligned} \tag{3}$$

i.e., similar to PGD but averaging the gradients over the batch of augmentations. The TTAs are generated randomly for each iteration. We initialize δ_0 as a zero gradient map and set the generated adversarial image as $x' = x + \delta_N$.

Adaptive white-box In this threat model the adversary has full knowledge about the DNN, the distribution of the transformations, and the random forest's parameters. The adversary knows everything about our defense method, including the training data (logits) used to fit the DNN and the random forest classifer. In this harsh settings we employ the BPDA attack [2] detailed in sup. mat.

4. Experimental Setup

We turn to detail the datasets we used, the DNN and random forest training, and inference computation time. Our hardware setup is thoroughly detailed in sup. mat.

Datasets. We perform our tests on 4 datasets: CIFAR-10, CIFAR-100 [33], SVHN [46] and Tiny ImageNet [35].

DNN training. We randomly split the training set of all the datasets into two subsets, *train* and *train-val*. The former is used to back-prop gradients from the loss to the inputs and train the DNN, whereas the latter is used for metric calculation to decay the learning rate. The size of the *train-val* set is chosen to be 5% of the official training set.

We trained three Resnet architectures [27], Resnet-34, Resnet-50, and Resnet-101, with global average pooling layer before the embedding space. The embedding vector was multiplied by a fully connected layer for the logits calculation. We trained CIFAR-10, CIFAR-100, SVHN, and Tiny ImageNet with 300, 300, 200, and 300 epochs, respectively; For the TRADES method (adversarial training) we trained them with 100, 100, 100, and 300 epochs, respectively, since we observed that fewer epochs obtain higher adversarial accuracy with TRADES (see sup. mat.). Early stopping was not used. All TRADES adversarial robust networks used $1/\lambda = 1$, $\epsilon = 0.031$, $\alpha = 0.007$ (ϵ step size), on L_∞ norm to match the settings in [70] for fair comparison. The VAT adversarial networks were also trained using ϵ=0.031, with $\alpha = 1$ and $\epsilon = 1, 1, 3, 1$ for CIFAR-10, CIFAR-100, SVHN, and Tiny ImageNet, respectively [43].

We use an L_2 weight decay regularization of 0.0001 in all our DNN training, a stochastic gradient decent optimizer with momentum 0.9 with Nesterov updates, and a batch size of 100. The training starts with a learning rate of 0.1, which decreases by a factor of 0.9 after 3 epochs of no improvement on the *train-val* accuracy (2 epochs for SVHN).

Random forest training. We split the test set of all the datasets into two subsets, *val* and *test*. The *test* size is 2500, and the *val* is the official test set without these 2500 samples, i.e., 7500, 7500, 23500, and 7500 for CIFAR-10, CIFAR-100, SVHN, and Tiny ImageNet, respectively. The only exception is the Boundary attack. Due to long processing time, we selected for it only 750, 250 samples from *val*, *test*, respectively. The random forest classifier was trained with 1000 trees, using the Gini impurity criterion. The training time was 71 seconds and was done only once for all the normal/adversarial images on the *val* subset.

Adversarial attacks. The adversarial attacks detailed in Sec. 3.4 were set to the following norms and powers: (1) FGSM[1]: $(L_\infty, \epsilon = 0.01)$; (2) FGSM[2]: $(L_\infty, \epsilon = 0.031)$; (3) JSMA: $(L_0, \gamma = 0.01)$; (4) PGD[1]: $(L_\infty, \epsilon = 0.01)$; (5) PGD[2]: $(L_\infty, \epsilon = 0.031)$; (6) Deepfool: ($L_2$, ϵ is unconstrained); (7) CW_{L_2}: (L_2, ϵ is unconstrained); (8) CW_{L_∞}: $(L_\infty, \epsilon = 0.031)$; (9) Square: $(L_\infty, \epsilon = 0.031)$; and (10) Boundary: ($L_2$, ϵ is unconstrained).

PGD was applied with a step size of $\alpha = 0.003$ with 100 iterations. The above attacks were selected due to their norm diversity, effectiveness, and popularity. Many attacks employ $\epsilon = 0.031$ to match the settings in the TRADES baseline [70], which is the current SOTA. For all the targeted attack we randomly switched the ground-truth label to one of the different labels in the dataset.

We also apply the following adaptive attacks detailed in Sections 3.4: (i) A-FGSM($L_\infty, \epsilon = 0.031$); (ii) A-PGD($L_\infty, \epsilon = 0.031$); (iii) A-Square ($L_\infty, \epsilon = 0.031$); and (iv) BPDA: BPDA($L_\infty, \epsilon = 0.031$). A-FGSM, A-PGD, A-Square, and BPDA were set with $N = 256$ generated TTAs. A-PGD, A-Square and BPDA are very time consuming and thus were set with only 10 iterations; Therefore, we set their step size to $\alpha = 0.007$.

Testing. All the metrics we show in this work were calculated on the *test* subset. For both TTA and ARF we set $N = 256$ unless stated otherwise, i.e., we generate 256 TTAs in inference time, which allows the models to run in a single forward pass on the GPU. The majority of the computation time is devoted to the TTAs generation, which is done on the CPU and takes 3.32 ± 0.33 seconds for a single Tiny ImageNet image (calculated over 20 runs). The DNN and random forest forward pass times are negligible - 250 ms and 4 ms, respectively.

5. Results

We evaluate the performance of ARF on adversarial attacks and compare it to other robust methods. We also present ablation studies conducted to improve the model performance and computation time. Lastly, we show accuracies on adaptive black-box and white-box attacks. In sup. mat. we present the distortions created by the different attacks. Alternative simple machine learning classifiers such as logistic regression and SVM were found to be inferior to random forest; This comparison appears in sup. mat.

5.1. Adversarial Robustness

Table 1 shows the accuracy on the normal (not attacked) and adversarial examples obtained for all the non-adaptive

Table 1. Comparing accuracy (%) of various classifiers on non-adaptive attacks. Section 4 detail the tested attacks and datasets. We boldface best results. Ensemble is added just as a reference as it has an unfair advantage (explained in the text).

Dataset	Method	Normal	FGSM^1	FGSM^2	JSMA	PGD^1	PGD^2	Deepfool	CW$_{L_2}$	CW$_{L_\infty}$	Square	Boundary
CIFAR-10	Plain	94.92	68.52	55.28	68.68	13.72	0.00	4.00	3.12	23.44	59.36	18.40
	Ensemble	96.04	82.00	64.20	84.60	86.68	48.64	86.96	83.64	78.80	89.48	96.00
	TRADES	86.64	85.04	75.80	69.88	85.12	71.84	7.68	0.56	78.24	80.92	22.80
	VAT	**94.00**	82.68	70.36	80.48	82.12	20.08	4.04	4.24	49.80	81.32	15.20
	TTA	91.68	82.48	68.76	84.84	87.04	72.76	83.16	82.24	81.4	85.32	88.80
	ARF	93.76	83.72	70.20	85.28	90.32	77.88	87.36	84.36	85.64	87.84	**91.20**
	TRADES + ARF	84.28	82.56	76.72	79.64	82.56	76.24	69.40	68.00	80.48	80.56	81.20
	VAT + ARF	92.60	**89.44**	**81.24**	**90.60**	**90.28**	**82.36**	**87.72**	**85.12**	**88.92**	**89.00**	90.80
CIFAR-100	Plain	74.32	28.96	13.84	43.84	22.52	0.28	9.20	15.84	47.76	28.64	30.40
	Ensemble	78.04	58.20	29.16	52.48	69.64	33.28	76.60	51.08	68.68	65.36	74.40
	TRADES	53.36	51.80	41.52	46.84	52.88	46.44	10.88	5.28	51.04	45.96	24.00
	VAT	70.92	52.56	28.80	59.88	63.00	15.20	10.00	11.20	44.36	54.36	20.80
	TTA	70.76	52.24	28.80	56.36	62.92	42.08	65.92	46.72	62.60	56.28	65.20
	ARF	**71.52**	54.20	30.80	58.08	**66.72**	47.60	**68.12**	49.36	**64.44**	60.40	**68.00**
	TRADES + ARF	49.48	49.04	44.16	48.04	48.80	46.28	45.04	36.56	49.96	47.48	43.60
	VAT + ARF	68.96	**63.52**	**48.20**	**65.76**	66.24	**59.92**	65.56	**55.76**	62.72	**64.04**	61.60
SVHN	Plain	97.36	80.56	66.24	49.64	52.32	2.04	2.84	4.80	28.96	66.96	15.60
	Ensemble	98.12	89.52	74.84	85.36	92.80	71.20	71.88	79.76	83.04	93.52	97.20
	TRADES	92.48	90.60	81.20	39.44	90.28	70.88	5.08	0.72	82.36	83.84	19.20
	VAT	94.44	89.00	83.72	65.48	85.16	43.28	4.16	23.60	64.76	87.04	18.80
	TTA	97.08	87.16	73.76	86.36	89.68	61.32	66.84	80.08	80.64	92.24	95.20
	ARF	**96.92**	87.96	75.24	87.00	90.04	66.16	68.84	80.40	81.28	92.16	**96.00**
	TRADES + ARF	92.44	90.96	82.20	78.80	91.16	79.96	62.48	48.40	85.84	88.40	88.00
	VAT + ARF	95.64	**93.72**	**86.92**	**89.84**	**93.76**	**81.88**	**92.08**	**85.68**	**90.28**	**93.24**	91.60
Tiny ImageNet	Plain	59.24	25.48	9.92	28.88	30.72	0.40	10.08	16.24	34.76	28.04	19.60
	Ensemble	67.12	58.32	28.64	52.96	63.92	46.24	66.56	54.64	60.08	60.80	58.40
	TRADES	44.44	42.32	31.64	35.32	43.72	38.44	9.16	5.52	41.88	38.76	23.20
	VAT	**54.68**	**47.84**	24.52	44.92	**52.36**	31.80	9.92	6.64	**46.24**	**46.00**	23.20
	TTA	52.48	37.12	17.36	39.76	43.84	27.52	46.24	35.08	42.96	40.84	37.60
	ARF	53.36	40.76	21.52	43.76	48.88	33.48	**48.88**	**40.04**	45.80	45.92	**42.80**
	TRADES + ARF	39.24	37.68	33.44	36.20	38.36	35.20	35.12	27.80	38.72	36.80	32.00
	VAT + ARF	47.92	45.52	**36.28**	**45.76**	46.32	**41.96**	43.44	34.84	**46.24**	44.84	39.20

attacks (black-box and gray-box) we employed on Resnet-34. Tables for Resnet-50 and Resnet-101 are in sup. mat.

"Plain" corresponds to the non-robust, simple DNN accuracy, without any adversarial defense. "Ensemble" uses nine different DNNs with the same architecture and the predicted label is a majority voting amongst them. It should be emphasized that the adversary does not have access to any of these nine models. Thus, it has an unfair advantage. TTA classifier is applied on the DNN alone (w/o random forest), and ARF is evaluated in two setups, one on a regular (non adversarially robust) DNN, and another is combined with an adversarially robust DNN trained with VAT or TRADES.

Note that for all datasets, ARF has better robustness than TTA and both VAT and TRADES on JSMA, Deepfool, and CW$_{L_2}$. It is not surprising as TRADES employed a regularization term on a ball with an L_∞ norm and these attacks use other norms. Also, VAT regularizes logits distribution smoothness within the image's local surrounding, which is problematic when the norm is unconstrained in L_∞.

In the vast majority of the attacks and datasets, the ARF classifier outperforms the VAT networks. However, when combined together they usually achieve the highest adversarial robustness accuracy. This robust accuracy trumps even the ensemble score, except for Tiny Imagenet.

Lastly, observe that the normal accuracy obtained by ARF is much better than TRADES, and is comparable to the normal accuracy of VAT. ARF scores almost as the plain classifier for normal images on CIFAR-10, CIFAR-100 and SVHN.

Transferability. In the sup. mat. we show that our ARF defense is characterized with excellent transferability, being able to generalize to new (unseen) attacks.

5.2. Ablation Studies

We conducted two ablation studies to analyze ARF.

ARF classifier ablation. We tested three parameters governing the ARF accuracy:

1. *Features:* The inputs to the random forest classifier. We used three candidates: The DNN's logits, the DNN's probabilities (softmax over logits), and the embedding vectors in the DNN penultimate layer.

2. *Gaussian noise power:* We checked three different noise filters with max standard deviation (σ_{max}) of 0, 0.005, and 0.0125. The 0 value is equivalent to no noise.

3. *Strength of transforms:* We tested two sets of transforms from the transforms in Fig. 2: *soft* vs *hard*. The *soft* transforms span over shorter parameter intervals. For example, the *hard* brightness transform randomizes a brightness factor in the interval $U(0.6, 1.4)$ whereas the *soft* transform randomizes it in $U(0.8, 1.2)$. The full interval sets of these transforms are in sup. mat.

Table 2 shows the normal and adversarial accuracies ($\mathbb{A}_{norm}$ and $\mathbb{A}_{adv}$) on CIFAR-10, trained by Resnet-34, attacked by CW$_{L_2}$ and evaluated using ARF with $N = 1000$. The highest adversarial accuracy was obtained for logits vectors, *hard* transforms, and $\sigma_{max} = 0.005$. Thus, these were the parameters we used in this work. It is interesting to

Table 2. Ablation study on 3 parameters used for ARF. 1) Random forest input features: Logits, softmax probabilities, and DNN embeddings. 2) Randomization level of transforms: *hard* for a larger randomization range (coarse transforms) and *soft* for a smaller range (mellow transforms). 3) Noise transform max power (σ_{max}). The adversarial score is computed for CW_{L_2}.

Features	Transforms	σ_{max}	Accuracy (%)	
			$\mathbb{A}_{norm}$	$\mathbb{A}_{adv}$
Logits	*soft*	0	**94.24**	83.56
Logits	*soft*	0.005	94.20	83.72
Logits	*soft*	0.0125	93.88	83.96
Logits	*hard*	0	93.72	84.64
Logits	*hard*	0.005	93.80	**85.00**
Logits	*hard*	0.0125	93.08	84.96
Probs	*soft*	0	94.16	83.36
Probs	*soft*	0.005	94.04	83.64
Probs	*soft*	0.0125	93.80	84.00
Probs	*hard*	0	93.60	84.52
Probs	*hard*	0.005	93.80	84.72
Probs	*hard*	0.0125	93.00	84.96
Embeddings	*soft*	0	94.16	83.56
Embeddings	*soft*	0.005	93.96	83.64
Embeddings	*soft*	0.0125	93.56	83.88
Embeddings	*hard*	0	93.68	84.88
Embeddings	*hard*	0.005	93.60	84.80
Embeddings	*hard*	0.0125	93.04	84.92

point out that the best normal accuracy was obtained for *soft* transforms with $\sigma_{max} = 0$ (for all features). This observation conforms with the high normal accuracy presented in Table 1, as the plain DNN does not apply any transform.

TTA size ablation. The computational bottleneck in our TTA and ARF classifiers is the generation of the N TTAs. Using $N = 1000$ images as done for Table 2 requires a long computation time so we searched for the minimal N, which achieves sufficient adversarial robustness. We select $N = 256$ for our experiments since it achieves good robustness with very high confidence.

Figure 3 shows the adversarial accuracy on CIFAR-10 for three selected attacks: PGD[1], Deepfool, and CW_{L_2} in a logarithmic scale. The width of each line corresponds to the measured standard deviation of five repeated experiments. We select $N = 256$ for our experiments since it achieves good robustness with very high confidence (narrow interval). Ablation of the TTA size on CIFAR-100 and SVHN is presented in sup. mat.

5.3. Is ARF a Masking Gradient Approach?

Table 3 shows the adversarial accuracies for different robust classifiers for all the adaptive attacks in Section 3.4. For easy comparison, the performance on the corresponded non-adaptive attack is shown next to each accuracy result. Note that our A-PGD attack is much more effective against the

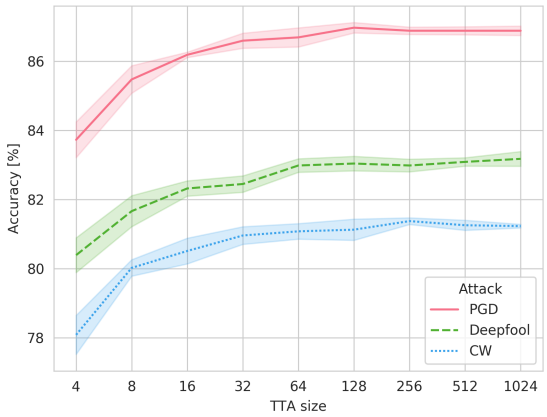

Figure 3. Ablation study on the number of generated TTAs (N). We calculate the adversarial accuracies on CIFAR-10 for three attacks as a function of N (logarithmic scale).

ensemble and TTA classifiers, surpassing all other adaptive and non-adaptive attacks by a large margin. Alas, it is not as powerful as the vanilla PGD against plain adversarial robust DNNs (TRADES/VAT). For all the other robust classifiers it achieves comparable results to the strong BPDA attack.

Observe that ARF is robust against the black-box adaptive attack, but fails when attacked with an adaptive gray-box or white-box attack. For example, the white-box BPDA attack decreases the ARF accuracy on CIFAR-10 to 8.8%. The VAT+ARF combination demonstrates SOTA robustness for all non-adaptive attacks, however, the vanilla TRADES or VAT perform better on adaptive attacks. Notice that these results may suggest that ARF may be considered as a gradient masking approach as revealing the gradients of the random forest, degrade the performance of our defense model significantly. Yet, we observe that the combination of TRADES+ARF is usually preferred against adaptive attack, obtaining high robustness for these attacks for all datasets. Also, although the performance of ARF degrades significantly when the BPDA harsh attack is applied, when combined with adversarial training this degradation is minimal.

Visual Perceptiblity. Although the ARF defense is susceptible to the BPDA attack, an adaptive white-box attack that was customly tailored to circumvent our specific random forest classifier, we show that BPDA fails to generate imperceptible images. We display some images generated using BPDA against ARF and demonstrate that a human observer can easily detect an unusual distortion in them.

Figure 4 exhibits clean images and adversarial images generated by BPDA for CIFAR-10, CIFAR-100, SVHN, and Tiny ImageNet. "Clean" column corresponds to natural (undistorted) images; "ARF" column denotes images that fool our ARF defense; "TRADES+ARF" and "VAT+ARF" columns display images that fool our ARF defense when combined with TRADES and VAT adversarially trained

Table 3. Adversarial accuracies (%) for various robust classifiers on adaptive attacks: A-Square (black-box), A-FGSM and A-PGD (gray-box) and BPDA (white-box), and their non-adaptive correspondents. FGSM^2 and PGD^2 are abbreviated to FGSM and PGD for clarity. ARF can maintain robustness only when combined with an adversarially trained DNN. Ensemble is presented just as a reference as it has an unfair advantage (see text).

Dataset	Method	FGSM	A-FGSM	PGD	A-PGD	Square	A-Square	BPDA
CIFAR-10	Ensemble	64.20	41.60	48.64	5.76	89.48	90.80	10.40
	TRADES	75.80	**79.20**	71.84	**77.76**	80.92	89.60	**84.00**
	VAT	70.36	68.84	20.08	54.84	81.32	**95.20**	58.80
	TTA	68.76	33.32	72.76	5.12	85.32	87.60	8.40
	ARF	70.20	37.80	77.88	5.48	87.84	89.20	8.80
	TRADES + ARF	76.72	74.88	76.24	73.48	80.56	87.20	82.80
	VAT + ARF	**81.24**	64.68	**82.36**	54.64	**89.00**	92.00	63.60
CIFAR-100	Ensemble	29.16	25.04	33.28	19.24	65.36	66.40	23.60
	TRADES	41.52	**48.76**	46.44	**48.44**	45.96	52.80	**50.40**
	VAT	28.80	40.32	15.20	39.32	54.36	64.00	43.60
	TTA	28.80	13.20	42.08	10.16	56.28	56.00	12.00
	ARF	30.80	16.60	47.60	11.08	60.40	58.00	12.80
	TRADES + ARF	44.16	46.44	46.28	47.60	47.48	47.20	46.00
	VAT + ARF	**48.20**	37.80	**59.92**	43.16	**64.04**	**64.80**	42.40
SVHN	Ensemble	74.84	62.44	71.20	36.04	93.52	96.80	45.60
	TRADES	81.20	**82.00**	70.88	**78.40**	83.84	92.40	**75.60**
	VAT	83.72	79.52	43.28	58.32	87.04	92.40	62.80
	TTA	73.76	56.92	61.32	19.92	92.24	**96.40**	32.80
	ARF	75.24	59.80	66.16	20.96	92.16	95.60	34.80
	TRADES + ARF	82.20	80.96	79.96	76.44	88.40	91.60	72.80
	VAT + ARF	**86.92**	79.68	**81.88**	65.36	**93.24**	94.00	66.40
Tiny ImageNet	Ensemble	28.64	30.04	46.24	38.16	60.80	56.80	39.60
	TRADES	31.64	40.76	38.44	40.00	38.76	40.80	37.60
	VAT	24.52	**43.12**	31.80	**46.28**	**46.00**	**49.60**	**43.60**
	TTA	17.36	6.96	27.52	6.72	40.84	40.00	11.60
	ARF	21.52	10.40	33.48	9.16	45.92	45.60	15.20
	TRADES + ARF	33.44	35.24	35.20	36.12	36.80	38.00	35.60
	VAT + ARF	**36.28**	36.16	**41.96**	40.56	44.84	38.00	36.40

DNNs, respectively. For a fair comparison, we show only images that successfully fool all the three defenses, meaning, the DNN classified the clean image successfully but the adversarial image was able to flip the label despite our random forest classifier.

Note that the most visible noises correspond to attacks on the vanilla ARF method, without incorporating TRADES/VAT into it. This observation is counterintuitive to the reported accuracies in Table 3 that show better robustness of ARF when combined with TRADES/VAT. Moreover, in the sup. mat. we show lower L_2 distortion for the vanilla ARF defense on BPDA compared to TRADES+ARF and VAT+ARF. Nonetheless, these visible distortions decrease the efficacy of BPDA towards our defense as it can be easily spotted by a naked eye.

6. Conclusions

This work proposes a simple, fast, and easy to use method to classify adversarial images, named ARF. Our approach is applied on pretrained DNNs without the need to carry out adversarial training or updating the model's parameters. ARF first generates many test-time augmentations, applying a wide variety of random color, geometrical, blur and noise transforms on the input image, and feeds these augmentations to a pretrained DNN. Then it collects the DNN's logits and feeds them to a vanilla random forest classifier which yields SOTA robust classification when combined with an adversarially trained DNN (VAT). This improvement in robustness comes at the cost of training the random forest

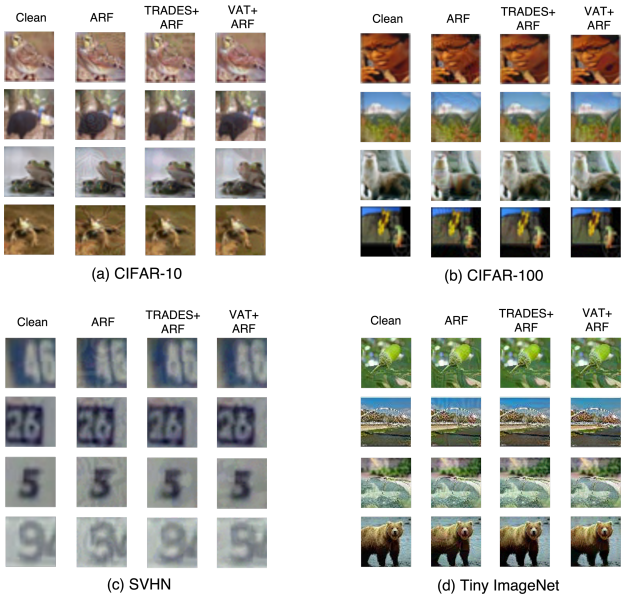

Figure 4. Adversarial images generated by BPDA circumventing our ARF defense. TRADES+ARF and VAT+ARF correspond to our ARF defense when applied on top of an adversarially trained DNN, TRADES/VAT, respectively. Adversarial images that fool our ARF defense can be easily spotted by the naked eye. Therefore, while they fool our network they do not meet the perceptual criterion of adversarial attacks. This shows that ARF is indeed a strong defense against adversarial attacks.

model (only once). We tested ARF with a variety of attacks, where some of them were especially designed against ARF. One of them, A-PGD, which we proposed, is of interest by itself as it is very effective against DNN ensemble while not having access to any of its networks.

When tested on adaptive attacks, ARF applied on a non-robust DNN shows inferior robust accuracies compared to a plain adversarial training (VAT/TRADES), suggesting that ARF's robustness is attributed to gradient masking. However, it was shown to perform well under the adaptive white-box threat model when combined with TRADES. Also, the white-box setting assumes full knowledge about our defense parameters, which can be easily changed by quickly re-training the simple ARF model upon every classification. Thus, hiding the ARF model can be considered as holding a secret key for "security through obscurity" [7, 32]. In addition, defending against new adaptive attacks is feasible by including them into the ARF fitting. Therefore, the use of ARF should be favored over adversarial training alone (although in the white-box setting tailored to ARF it was better alone) as in the non white-box setting ARF leads to a significant improvement. We believe that integrating ARF within the adversarial training can further boost the robustness as was shown for data augmentations in a very recent work [51].

References

[1] Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square Attack: a query-efficient black-box adversarial attack via random search. In *ECCV*, 2020. 2, 4

[2] Anish Athalye, Nicholas Carlini, and David A Wagner. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples. In *ICML*, 2018. 2, 3, 5

[3] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing Robust Adversarial Examples. *ArXiv*, abs/1707.0, 2018. 3

[4] Yuval Bahat, Michal Irani, and Gregory Shakhnarovich. Natural and Adversarial Error Detection using Invariance to Image Transformations. *ArXiv*, abs/1902.0, 2019. 3

[5] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-Based Adversarial Attacks: Reliable Attacks Against Black-Box Machine Learning Models. In *ICLR*, 2018. 2, 4

[6] Jacob Buckman, Aurko Roy, Colin Raffel, and Ian Goodfellow. Thermometer Encoding: One Hot Way To Resist Adversarial Examples. In *ICLR*, 2018. 2

[7] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, and Alexey Kurakin. On Evaluating Adversarial Robustness. *CoRR*, abs/1902.0, 2019. 2, 8

[8] Nicholas Carlini and David A Wagner. Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods. In *AISec@CCS*, 2017. 1, 2

[9] Nicholas Carlini and David A Wagner. Towards Evaluating the Robustness of Neural Networks. *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57, 2017. 2

[10] Y. Carmon, Aditi Raghunathan, Ludwig Schmidt, Percy Liang, and John C. Duchi. Unlabeled data improves adversarial robustness. In *NeurIPS*, 2019. 3

[11] Anirban Chakraborty, Manaar Alam, Vishal Dey, A Chattopadhyay, and Debdeep Mukhopadhyay. Adversarial Attacks and Defences: A Survey. *ArXiv*, abs/1810.0, 2018. 2

[12] Alvin Chan, Yi Tay, and Yew-Soon Ong. What it thinks is important is important: Robustness transfers through input gradients. In *CVPR*, June 2020. 3

[13] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E Hinton. A Simple Framework for Contrastive Learning of Visual Representations. *ArXiv*, abs/2002.0, 2020. 3

[14] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *ICML*, 2017. 2

[15] Gilad Cohen, Guillermo Sapiro, and Raja Giryes. Detecting adversarial samples using influence functions and nearest neighbors. In *CVPR*, June 2020. 1

[16] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *ICML*, volume 97, pages 1310–1320, 09–15 Jun 2019. 3

[17] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning*, volume 119, pages 2206–2216, 2020. 2

[18] B. Darvish Rouani, M. Samragh, T. Javidi, and F. Koushanfar. Safe machine learning and defeating adversarial attacks. *IEEE Security Privacy*, 17(2):31–38, 2019. 2

[19] Guneet S. Dhillon, Kamyar Azizzadenesheli, Jeremy D. Bernstein, Jean Kossaifi, Aran Khanna, Zachary C. Lipton, and Animashree Anandkumar. Stochastic activation pruning for robust adversarial defense. In *ICLR*, 2018. 2

[20] Yifan Ding, Liqiang Wang, Huan Zhang, Jinfeng Yi, Deliang Fan, and Boqing Gong. Defending against adversarial attacks using random forest. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 105–114, 2019. 3

[21] Abhimanyu Dubey, Laurens van der Maaten, Zeki Yalniz, Yixuan Li, and Dhruv Kumar Mahajan. Defense against adversarial images using web-scale nearest-neighbor search. *CVPR*, pages 8759–8768, 2019. 3

[22] Gintare Karolina Dziugaite, Zoubin Ghahramani, and Daniel M Roy. A study of the effect of JPG compression on adversarial images. *ArXiv*, abs/1608.0, 2016. 3

[23] Yue Gao, Ilia Shumailov, Kassem Fawaz, and Nicolas Papernot. On the limitations of stochastic pre-processing defenses. In *Thirty-Sixth Conference on Neural Information Processing Systems*, 2022. 2

[24] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining And Harnessing Adversarial Examples. In *ICLR*, 2015. 1, 2, 3

[25] Abigail Graese, Andras Rozsa, and Terrance E Boult. Assessing Threat of Adversarial Examples on Deep Neural Networks. *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 69–74, 2016. 3

[26] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens van der Maaten. Countering Adversarial Images using Input Transformations. In *ICLR*, 2018. 2, 3

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *CVPR*, pages 770–778, 2016. 5

[28] Matthias Hein and Maksym Andriushchenko. Formal Guarantees on the Robustness of a Classifier against Adversarial Manipulation. In *NIPS*, 2017. 2

[29] Daniel Jakubovitz and Raja Giryes. Improving DNN Robustness to Adversarial Attacks Using Jacobian Regularization. In *ECCV*, 2018. 2

[30] Ahmadreza Jeddi, Mohammad Javad Shafiee, Michelle Karg, Christian Scharfenberger, and Alexander Wong. Learn2perturb: An end-to-end feature perturbation learning to improve adversarial robustness. In *CVPR*, June 2020. 3

[31] Harini Kannan, Alexey Kurakin, and Ian Goodfellow. Adversarial logit pairing. *arXiv preprint arXiv:1803.06373*, 2018. 2

[32] A Kerckhoffs. La cryptographie militaire. *Journal des Sciences Militaires*, pages 161–191, 1883. 8

[33] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009. 5

[34] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial Machine Learning at Scale. *CoRR*, abs/1611.0, 2017. 3

[35] Ya Le and X Yang. Tiny ImageNet Visual Recognition Challenge. 2015. 5

[36] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *NeurIPS*, 2018-Decem:7167–7177, 2018. 1

[37] Jiajun Lu, Hussein Sibai, Evan Fabry, and David Alexander Forsyth. NO Need to Worry about Adversarial Examples in Object Detection in Autonomous Vehicles. *ArXiv*, abs/1707.0, 2017. 3

[38] Yan Luo, Xavier Boix, Gemma Roig, Tomaso A Poggio, and Qi Zhao. Foveation-based Mechanisms Alleviate Adversarial Examples. *ArXiv*, abs/1511.0, 2015. 2, 3

[39] Xingjun Ma, Bo Li, Yisen Wang, Sarah M Erfani, Sudanthi N R Wijewickrema, Michael E Houle, Grant Schoenebeck, Dawn Song, and James Bailey. Characterizing Adversarial Subspaces Using Local Intrinsic Dimensionality. *CoRR*, abs/1801.0, 2018. 1

[40] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. In *ICLR*, 2018. 1, 2, 3

[41] Dongyu Meng and Hao Chen. Magnet: A two-pronged defense against adversarial examples. In *ACM SIGSAC Conference on Computer and Communications Security*, page 135–147, 2017. 3

[42] D. J. Miller, Z. Xiang, and G. Kesidis. Adversarial learning targeting deep neural network classification: A comprehensive review of defenses against attacks. *IEEE*, 108(3):402–433, 2020. 2

[43] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, and Shin Ishii. Distributional Smoothing with Virtual Adversarial Training. In *ICLR*, 2016. 3, 5

[44] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. DeepFool: A simple and accurate method to fool deep neural networks. *CVPR*, pages 2574–2582, 2016. 2

[45] Federico Nesti, Alessandro Biondi, and Giorgio C Buttazzo. Detecting Adversarial Examples by Input Transformations, Defense Perturbations, and Voting. *IEEE transactions on neural networks and learning systems*, PP, 2021. 2

[46] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading Digits in Natural Images with Unsupervised Feature Learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. 5

[47] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *IEEE Symposium on Security and Privacy (SP)*, 2016. 2, 3

[48] Nicolas Papernot, Patrick D McDaniel, and Ian J Goodfellow. Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples. *CoRR*, abs/1605.0, 2016. 3

[49] Nicolas Papernot, Patrick D McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The Limitations of Deep Learning in Adversarial Settings. *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 372–387, 2016. 2

[50] J. C. Perez, M. Alfarra, G. Jeanneret, L. Rueda, A. Thabet, B. Ghanem, and P. Arbelaez. Enhancing adversarial robustness via test-time transformation ensembling. In *IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 81–91, 2021. 3

[51] Sylvestre-Alvise Rebuffi, Sven Gowal, Dan Andrei Calian, Florian Stimberg, Olivia Wiles, and Timothy Mann. Data Augmentation Can Improve Robustness. In *NeurIPS*, 2021. 8

[52] Andrew Slavin Ross and Finale Doshi-Velez. Improving the Adversarial Robustness and Interpretability of Deep Neural Networks by Regularizing Their Input Gradients. In *AAAI*, 2017. 2

[53] Kevin Roth, Yannic Kilcher, and Thomas Hofmann. The Odds are Odd: A Statistical Test for Detecting Adversarial Examples. In *ICML*, 2019. 3

[54] Andras Rozsa, Manuel Gunther, and Terrance E. Boult. Towards robust deep neural networks with bang. In *WACV*, 2018. 2

[55] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-GAN: Protecting classifiers against adversarial attacks using generative models. In *ICLR*, 2018. 2

[56] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *NeurIPS*, pages 3358–3369, 2019. 3

[57] Uri Shaham, Yutaro Yamada, and Sahand Negahban. Understanding adversarial training: Increasing local stability of supervised models through robust optimization. *Neurocomputing*, 307:195–204, 2018. 3

[58] Chawin Sitawarin and Dávid Wágner. Defending against adversarial examples with k-nearest neighbor. *ArXiv*, abs/1906.09525, 2019. 3

[59] David Stutz, Matthias Hein, and Bernt Schiele. Disentangling Adversarial Robustness and Generalization. In *CVPR*, 6 2019. 3

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

[61] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In *NeurIPS*, 2020. 1

[62] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble Adversarial Training: Attacks and Defenses. In *ICLR*, 2018. 3

[63] Qinglong Wang, Wenbo Guo, Kaixuan Zhang, Alexander Ororbia, Xinyu Xing, C Lee Giles, and Xue Liu. Learning Adversary-Resistant Deep Neural Networks. *ArXiv*, abs/1612.0, 2016. 2, 3

[64] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. In *ICLR*, 2019. 3

[65] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating Adversarial Effects Through Randomization. In *ICLR*, 2018. 3

[66] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *CVPR*, pages 501–509, 2019. 3

[67] X. Yuan, P. He, Q. Zhu, and X. Li. Adversarial examples: Attacks and defenses for deep learning. *IEEE Transactions on Neural Networks and Learning Systems*, 30(9):2805–2824, 2019. 2

[68] Runtian Zhai, Tianle Cai, Di He, Chen Dan, Kun He, John Hopcroft, and Liwei Wang. Adversarially robust generalization just requires more unlabeled data. *arXiv preprint arXiv:1906.00555*, 2019. 3

[69] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 2

[70] Hongyang Zhang, Yaodong Yu, J Jiao, E Xing, L Ghaoui, and Michael I Jordan. Theoretically Principled Trade-off between Robustness and Accuracy. In *ICML*, 2019. 3, 5

[71] Linjun Zhang, Zhun Deng, Kenji Kawaguchi, Amirata Ghorbani, and James Zou. How does mixup help with robustness and generalization? *arXiv preprint arXiv:2010.04819*, 2020. 2