DAD: Data-free Adversarial Defense at Test Time

Gaurav Kumar Nayak* Ruchit Rawal* Anirban Chakraborty

Department of Computational and Data Sciences
Indian Institute of Science, Bangalore, India
{gauravnayak, ruchitrawal, anirban}@iisc.ac.in

Abstract

*Deep models are highly susceptible to adversarial attacks. Such attacks are carefully crafted imperceptible noises that can fool the network and can cause severe consequences when deployed. To encounter them, the model requires training data for adversarial training or explicit regularization-based techniques. However, privacy has become an important concern, restricting access to only trained models but not the training data (*e.g. *biometric data). Also, data curation is expensive and companies may have proprietary rights over it. To handle such situations, we propose a completely novel problem of 'test-time adversarial defense in absence of training data and even their statistics'. We solve it in two stages: a) detection and b) correction of adversarial samples. Our adversarial sample detection framework is initially trained on arbitrary data and is subsequently adapted to the unlabelled test data through unsupervised domain adaptation. We further correct the predictions on detected adversarial samples by transforming them in Fourier domain and obtaining their low frequency component at our proposed suitable radius for model prediction. We demonstrate the efficacy of our proposed technique via extensive experiments against several adversarial attacks and for different model architectures and datasets. For a non-robust Resnet-*18 *model pretrained on CIFAR-10, our detection method correctly identifies* 91.42% *adversaries. Also, we significantly improve the adversarial accuracy from* 0% *to* 37.37% *with a minimal drop of* 0.02% *in clean accuracy on state-of-the-art 'Auto Attack' without having to retrain the model.*

1. Introduction

Deep learning models have emerged as effective solutions to several computer vision and machine learning problems. However, such models have become unreliable as they may yield incorrect predictions when encountered with input data added with a carefully crafted human-imperceptible noise, also termed as 'adversarial noise' [38]. These vulnerabilities of the deep models can also cause severe implications in applications involving safety and security concerns such as biometric authentication via face recognition in ATMs [33], mobile phones [1] etc. It can even become life crucial in self-driving cars [21, 10] where autonomous vehicles can be made to take incorrect decisions when the important traffic objects are manipulated, *e.g*. stop signs.

Several attempts have been made to make the deep models robust against the adversarial attacks. We can broadly categorize them as: i.) adversarial training [12, 31] and ii.) non-adversarial training methods [17]. These two family of approaches have their own limitations as the former one is more computationally expensive while the latter provides weaker defense, albeit at a lower computational overhead. Instead of making the model robust, there are also approaches to detect these attacks [26, 16, 48, 3, 47, 29, 43, 23]. These methods often require retraining of the network [16, 3, 23]. Few of them use statistical [48, 47, 43] and signal processing techniques [29, 26]. All these existing works have a strong dependency on either the training data or their statistics. However, recently several works (*e.g*. [36, 49, 52, 50]) have identified many use cases where the training data may not be freely available, but instead the trained models. For example, pretrained models on Google's JFT-300M [18] and Deepface model [39] of Facebook do not release their training data. The training data may not be shared for many reasons such as data privacy, proprietary rights, transmission limitations, etc. Even biometric data are sensitive, prohibiting their distribution due to privacy. Also, several companies would not prefer sharing their precious data freely due to competitive advantage and the expensive cost incurred in data curation and annotation. Thus, we raise an important concern: *'how to make the pretrained models robust against adversarial attacks in absence of original training data or their statistics'*.

One potential solution is to generate pseudo-data from the pretrained model and then use them as a substitute for the unavailable training data. Few works attempt to generate such data either directly via several iterations of backpropagations [36, 49] or through GAN based approaches involving complicated optimization [2]. However,

*denotes equal contribution.

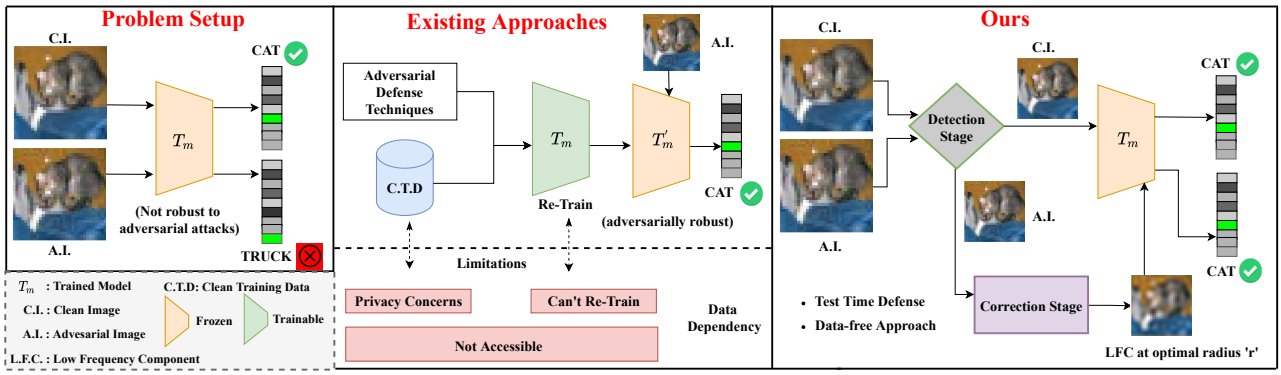

Figure 1. Comparison of existing approaches and ours. Traditional approaches fail to handle data privacy scenarios, Our method provides data-free robustness against adversarial attacks at test time without even retraining the pretrained non robust model T_m.

the pseudo-data generation process is computationally expensive. Furthermore retraining the model on the generated data using adversarial defense techniques is an added computation overhead. This motivates an alternative strategy that involves test time adversarial detection and subsequent correction on input space (data) instead of model, without generating any computationally expensive synthetic samples. However, even identifying the adversarial input samples with no prior knowledge about the training data is a non-trivial and difficult task. After detection, we aim to go a step ahead to even correct the adversarial samples which makes the problem extremely challenging and our proposed method is an initial attempt towards it . The problem set up and the major difference between existing methods and our proposed solution strategy is also summarized in Figure 1.

Our proposed detection method leverages adversarial detector classifier trained on any arbitrary data. In order to reduce the domain shift between that arbitrary data on which the detector is trained and our unabelled test samples, we frame our detection problem as an unsupervised domain adaptation (UDA). To further reduce the dependency on the arbitrary data, we use source-free UDA technique [28] which also allows our framework to even use any off-the-shelf pretrained detector classifier. Our correction framework is based on human cognition that human beings ignore the inputs that are outside a certain frequency range [45]. However, the model predictions are highly associated with high frequency components [42]. As adversarial attacks disrupt the model predictions, hence, we ignore the high frequency of the input beyond a certain radius. Selecting a suitable radius is crucial as removing high frequency components at low radius leads to lower discriminability while at high radius favours adversarial attacks. Adversarial noise creeps in when we aim for high discriminability. Therefore, we also propose a novel algorithm that finds a good trade off between these two. Our correction method is independent of the detection scheme. Hence, any existing data-dependent detection techniques can also benefit from our correction framework by easily plug in at test time to correct the detected adversaries. Also, as we do not modify the trained model (T_m in Figure 1), our method is architecture agnostic and works across a range of network architectures.

Our overall contributions can be summarized as follows:

- We are the first to attempt a novel problem of data-free adversarial defense at test time.
- We propose a novel adversarial detection framework based on source-free unsupervised domain adaptation technique (Sec. 4.1), which is also the first work that does not depend on the training data and their statistics.
- Inspired from human cognition, our correction framework analyzes the input data in Fourier domain and discards the adversarially corrupted high-frequency regions. A respectable adversarial accuracy is achieved by selecting the low-frequency components for each unlabelled sample at an optimal radius r^* proposed by our novel algorithm (Sec. 4.2).
- We perform extensive experiments on multiple architectures and datasets to demonstrate the effectiveness of our method. Without even retraining the trained non-robust Resnet-18 model on CIFAR-10 and in absence of training data, we obtain a significant improvement in the adversarial accuracy from 0% to 37.37% with a minimal drop of 0.02% in the clean accuracy on state-of-the-art Auto Attack [9].

2. Related Works

2.1. Adversarial Detection

Adversarial detection methods aim to successfully detect adversarially perturbed images. These can be broadly achieved by using trainable-detector or statistical-analysis based methods. The former usually involves training a detector-network either directly on the clean and adversarial

images in spatial [24, 30, 32] / frequency domain [14] or on logits computed by a pre-trained classifier [4]. Statistical-analysis based methods employ statistical tests like maximum mean discrepancy [13] or propose measures [11, 27] to identify perturbed images.

It is impractical to use the aforementioned techniques in our problem setup as they require training data to either train the detector (trainable-detector based) or tune hyper-parameters (for statistical-analysis based). We tackle this by formulating the detection of adversarial samples as a source-free unsupervised domain adaptation setup wherein we can adapt a detector trained on arbitrary (source) data to unlabelled (target) data at test-time.

2.2. Adversarial Robustness

While numerous defenses have been proposed to make a model robust to adversarial perturbations, adversarial training is arguably the only one that has stood the test of time. Szegedy *et al*. [38] first articulated adversarial training as augmenting the training data with adversaries to improve the robustness performance to a specific attack. Madry *et al*. [31] proposed the Projected Gradient Descent (PGD) attack, which when used in adversarial training provided robustness against a wide-class of iterative and non-iterative attacks. Apart from adversarial-training, other non-adversarial training based approaches primarily aim to regularize the network to reduce overfitting and achieve properties observed in adversarially trained models explicitly. Most notably, Jacobian Adversarially Regularized networks [8] achieve adversarial robustness by optimizing the model's jacobian to match natural training images.

Since we aim to make the model robust at test-time, it's not possible to apply either adversarial training or regularization approaches. We instead focus on reducing adversarial contamination in input data itself to achieve decent adversarial accuracy without dropping clean accuracy.

2.3. Frequency Domain

Wang *et al*. [42] in their work demonstrated that unlike humans, CNN relies heavily on high-frequency components (HFC) of an image. Consequently, perturbations in HFC cause changes in model prediction but are imperceptible to humans. More recently, Wang *et al*. [45] showed that many of the existing adversarial attacks usually perturb the high-frequency regions and proposed a method to measure the contribution of each frequency component towards model prediction. Taking inspiration from these recent observations we propose a novel detection and correction module that leverages frequency domain representations to improve adversarial accuracy (without plunging clean-accuracy) at test-time. The next section explains important preliminaries followed by proposed approach in detail.

3. Preliminaries

Notations: The target model T_m is pretrained on a training dataset D_{train}. The complete target dataset $D_{target} = \{D_{train}, D_{test}\}$. We assume no access to D_{train} but trained model T_m is available. $D_{test} = \{x_i\}_{i=1}^{N}$ is unlabelled testing data containing N test samples. We denote a set of adversarial attacks by $A_{attack} = \{A_j\}_{j=1}^{K}$ where K is the number of different attacks. The i^{th} test sample x_i is perturbed by any attack $A_j \in A_{attack}$ that fools the network T_m and the corresponding adversarial sample is $x_i^{'}$.

We denote any arbitrary dataset by $D_{arbitrary} = \{(x_{iA}, y_{iA})\}_{i=1}^{M}$ that has M labelled samples and the dataset $D_{arbitrary}$ is different from D_{target}. The model S_m is trained on arbitrary dataset $D_{arbitrary}$. The adversarial sample corresponding to x_{iA} is $x_{iA}^{'}$ which is obtained when the trained model S_m is attacked by any attack $A_j \in A_{attack}$.

$T_m(x_i)$ and $S_m(x_{iA})$ are the logits predicted for i^{th} sample from D_{test} and $D_{arbitrary}$ respectively. The softmax function is denoted by $soft()$. The label predicted by network T_m and S_m on i^{th} sample is denoted by $label(T_m(x_i)) = \text{argmax}(soft(T_m(x_i)))$ and $label(S_m(x_{iA})) = \text{argmax}(soft(S_m(x_{iA})))$. Let A_{test} and $A_{arbitrary}$ are the complete set of adversarial samples that fools the model T_m and S_m respectively, such that $x_i^{'} \in A_{test}$ and $x_{iA}^{'} \in A_{arbitrary}$ for an i^{th} image. The set of layers that are used for adversarial detection are denoted by L_{advdet}.

The fourier transform and inverse fourier transform operations are denoted by $F(.)$ and $F^{-1}(.)$ respectively. The frequency component of an i^{th} sample is denoted by f_i. The low frequency component (LFC) and high frequency component (HFC) of a sample f_i which are separated by a radius (r) are denoted by fl_{ir} and fh_{ir} respectively.

Adversarial noise: Any adversarial attack $A_j \in A_{attack}$ fools the pretrained network T_m by changing the label prediction. An attack A_j on an i^{th} image x_i computes an adversarial image $x_i^{'}$ such that $label(T_m(x_i)) \neq label(T_m(x_i^{'}))$. To obtain $x_i^{'}$, the i^{th} image x_i is perturbed by an adversarial noise δ which is imperceptible such that $\|\delta\|$ is within some ϵ. We restrict to perturbations within the l_∞ ball of radius ϵ.

Unsupervised Domain Adaptation (UDA): A classifier F_s is trained on labelled source dataset D_s which comes from distribution S. The unlabelled target dataset D_t belongs to a different distribution T. UDA methods attempt to reduce the domain gap between S and T with an objective to obtain an adapted classifier F_t using F_s which can predict labels on the unlabelled samples from D_t. If we assume D_s to be unavailable for adaptation then this problem is referred to as source-free UDA [28, 25, 22].

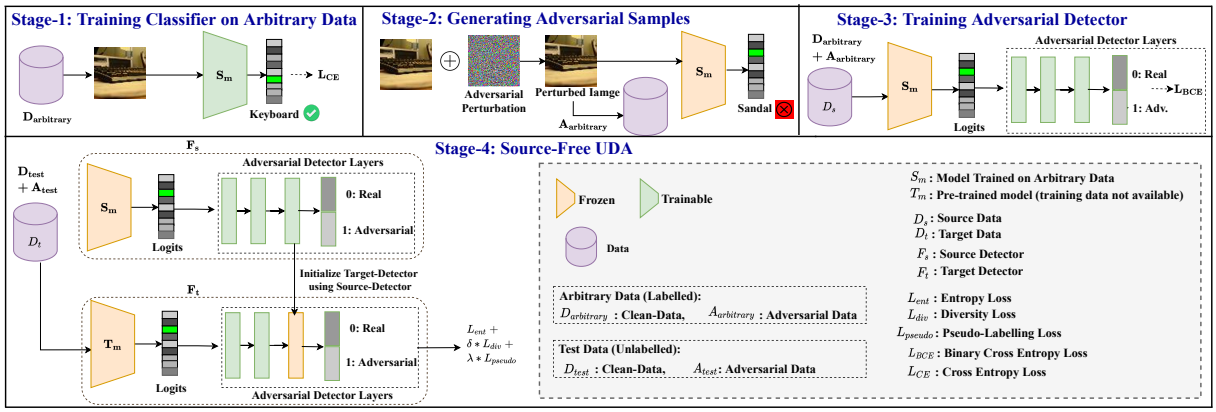

Figure 2. Our proposed detection module depicting different operations in each of the stages. We formulate adversarial detection as a source-free UDA problem where we adapt the source detector trained on arbitrary data to target detector using unlabelled clean and adversarial test samples.

Fourier Transform (FT): This operation is used in image processing to transform the input image from spatial to frequency domain [6]. For any i^{th} image x_i, its FT is defined as $F(x_i) = f_i$. At radius r,

$$\begin{aligned} fl_{ir} &= LFC(f_i, r) \\ fh_{ir} &= HFC(f_i, r) \end{aligned} \tag{1}$$

The inverse of FT helps to get back the spatial domain from the frequency domain. Thus, we have:

$$\begin{aligned} xl_{ir} &= F^{-1}(fl_{ir}) \\ xh_{ir} &= F^{-1}(fh_{ir}) \end{aligned} \tag{2}$$

where xl_{ir} and xh_{ir} are the LFC and HFC of i^{th} image in the spatial domain.

4. Proposed Approach

Test-time Adversarial Defense Set up: Given a pretrained target model T_m, our goal is to make the model robust against the set of adversarial attacks A_{attack} i.e. T_m should not change it's prediction on the set of adversarial samples A_{test}. The access to the dataset D_{train} is restricted due to data privacy. The objective is to maximize the performance of T_m on A_{test} without compromising much on D_{test}. As shown in Figure 1, we add a detection block before feeding the input to model T_m. The test samples which are detected as adversarial are passed to the correction module to minimize the adversarial contamination. The clean detected samples as well as corrected adversarial samples (output after correction stage) are then fed to pretrained model T_m to get the predictions. Next we explain in detail about our proposed detection and correction modules.

4.1. Detection module

The key idea is that if we have access to an adversarial detector which can classify samples from an arbitrary dataset as either clean and adversarial, then this binary classifier can be treated as a source classifier. The adversarial detector for the target model T_m can be thought of as target classifier with the target data being the collection of unlabelled test samples and their corresponding adversarial samples. Thus, as described in preliminaries of UDA, we can formulate the adversarial detection as a UDA problem where:

$F_s \leftarrow$ model S_m appended with detection layers L_{advdet}
$D_s \leftarrow$ mix of $D_{arbitrary}$ (clean) and $A_{arbitrary}$ (adversarial)
$D_t \leftarrow$ mix of D_{test} (clean) and A_{test} (adversarial)
$F_t \leftarrow$ model T_m appended with detection layers L_{advdet}

The goal of our detection module is to obtain the model F_t via UDA technique (to reduce the domain shift between D_s and D_t) that can classify samples from D_t as either clean or adversarial.

Our detection module consists of four stages which are shown in detail in Figure 2. Each stage performs a specific task which are described below:

Stage-1: Train model S_m with labelled data $D_{arbitrary}$ by minimizing the cross entropy loss $\min \sum_{i=1}^{M} L_{ce}(S_m(x_{iA}), y_{iA})$.

Stage-2: Generate a set of adversarial samples ($A_{arbitrary}$) by using any adversarial attack A_j from a set of adversarial attacks (A_{attack}) such that it fools the trained network S_m.

Stage-3: Train the adversarial detection layers (L_{advdet}) with input being the logits of network S_m and output is the soft score for the two classes (adversarial and clean). Similar to [4], the layers in L_{advnet} are composed of three fully connected layers containing 128 neurons and ReLU activations (except the last-layer). The first-two layers are followed by a dropout-layer with 25% dropout-rate. Additionally, the second (after-dropout) and third layer are followed by a Batch Normalization and Weight

Normalization layer respectively. The training is done using binary cross entropy loss L_{BCE} on data D_s. The label smoothing [35] is also done to further improve the discriminability of the source model F_s.

Stage-4: Perform UDA with the source network as F_s and the target network as F_t where the source data is D_s and target data is D_t. If we remove the dependency on the dataset D_s, then it can also easily facilitate any off-the-shelf pretrained detector classifier to be used in our framework and Stages 1 to 3 can be skipped in that case. Thus, to make our framework completely data-free and reduce its dependency on even arbitrary data, we adopt source-free UDA. To the best of our knowledge, the UDA setup for adversarial detection has not been explored previously even for data-dependent approaches.

The target network F_t is trained on the unlabelled dataset D_t where network T_m is frozen and entire layers of L_{advdet} of F_t is kept trainable expect the last classification layer and are initialized with weights of L_{advdet} of F_s. Inspired by [28], we also use three losses for training. We minimize the entropy loss (L_{ent}) to enforce that the network F_t predicts each individual sample with high confidence i.e. strong predictions for one of the classes (adversarial or clean). But this may result in a degenerate solution where only the same class gets predicted (i.e. same one-hot embedding). So, to avoid that we maximize diversity loss (L_{div}) that ensures high entropy across all the samples i.e. enforcing the mean of network predictions to be close to uniform distribution. Still, the unlabelled samples in D_t can get assigned a wrong label even if the ($L_{ent} - L_{div}$) is minimized. To overcome this, pseudo-labels are estimated via self-supervision [7, 28]. The initial centroid for a class is estimated by weighted mean of all samples where the individual sample weight is the predicted score of that class. Then the cosine similarity of the samples with respect to each centroid is calculated and the label of the nearest centroid is used as initial pseudo-labels. Next, the initial centroids and pseudo-labels are updated in a similar way as in K-means. L_{pseudo} is cross entropy loss on D_t where the assigned pseudo-labels are used as ground truth labels. Thus, overall loss $L = (L_{ent} - \delta L_{div} + \lambda L_{pseudo})$ is minimized where δ and λ are hyperparameters.

4.2. Correction module

The input images which are classified as adversarial by the trained adversarial detector model F_t are passed to this module. Here the objective is to minimize the contamination caused by adversarial attack on a given image such that the pretrained model T_m would retain its original prediction. In other words, an i^{th} sample $x_i^{'} \in A_{test}$ should be corrected as $x_{ic}^{'}$ so that $label(T_m(x_{ic}^{'})) = label(T_m(x_i))$.

The trained model gets fooled when its prediction gets altered on adversarial samples. Also, the deep model's prediction is highly associated with the HFC of an input image [42]. To lessen the effect of adversarial attack, we need to get rid of the contaminated HFC. This also resonates with human cognition as humans use LFC for decision making while ignoring the HFCs [45]. Consequently, the high frequency components are required to be ignored at some radius r. Thus, for an i^{th} adversarial image $x_i^{'}$ (refer FT operations in Preliminaries Sec. 3), we have:

$$\begin{gathered} f_i' = F(x_i^{'}) \\ \text{At any radius } r: fl_{ir}^{'} = LFC(f_i^{'}, r), fh_{ir}^{'} = HFC(f_i^{'}, r) \\ xl_{ir}^{'} = F^{-1}(fl_{ir}^{'}) \end{gathered} \tag{3}$$

However, $xl_{ir}^{'}$ obtained on a random selection of a radius r may yield poor and undesired results (refer Table 3). LFC's at high radius favours more discriminability (Disc) but high adversarial contamination (AdvCont). On the other hand, LFC's at small radius allows low AdvCont but low Disc. Hence, careful selection of suitable radius r^* is necessary to have a good trade off between Disc and AdvCont i.e. low AdvCont but at the same time having high Disc. Hence, the corrected i^{th} image in the spatial domain at radius r^* is given as:

$$x_{ic}^{'} = xl_{ir^*}^{'} = F^{-1}(fl_{ir^*}^{'}) \tag{4}$$

Estimating the radius r^* is not a trivial task especially when no prior knowledge about the training data is available and the value of suitable r^* can even vary for each adversarial sample. To overcome this problem, our proposed correction method estimates r^* for each sample and returns the corrected sample which is explained in detail in Algorithm 1. We define the minimum and maximum radius i.e. r_{min} and r_{max} (Line 2). As discussed, there are two major factors associated with the radius selection: Disc and AdvCont. Thus, we measure these two quantities present in LFC at each radius between r_{min} and r_{max} with a step size of 2. We quantify the Disc using SSIM [44] which is a perceptual metric. Specifically, the Disc score for an i^{th} adversarial sample (Line 6) is given as:

$$Disc_{score}(x_i^{'}, xl_{ir}^{'}) = SSIM(x_i^{'}, xl_{ir}^{'}) \tag{5}$$

As adversarial samples are perceptually similar to clean samples, hence SSIM score should be high to have high discriminability. We normalize SSIM between 0 to 1. Although increasing the radius leads to better $Disc_{score}$, it also allows the adversarial perturbations (usually located in HFC regions) to pass through. Thus, we also need to quantify AdvCont i.e. how much perturbations have crept in the LFC with respect to adversarial image $x_i^{'}$. To solve this, we compute the label-change-rate (LCR) at each radius. The key intuition of our method lies in the fact that if enough perturbations have passed through at some radius

Algorithm 1: Correction of adversarial samples (A_{test}) by determining the best radius r^*

Input: Pretrained model T_m,
$x_i^{'}$: i^{th} adversarial sample
Output: $x_{ic}^{'}$: corrected i^{th} adversarial sample

1 Obtain prediction on the i^{th} adversarial sample:
$advpred \leftarrow label(T_m(x_i^{'}))$
2 Initialize:
$r_{min} \leftarrow 2, r_{max} \leftarrow 16, count \leftarrow 10, r^* \leftarrow r_{min}$
3 Set dropout in training mode for T_m
4 **for** $r = r_{min};\ r \leq r_{max};\ r = r + 2$ **do**
5 Obtain LFC $xl_{ir}^{'}$ for the i^{th} adversarial sample $x_i^{'}$ using equation 3
6 Compute discriminability score for $xl_{ir}^{'}$:
$Disc\_xl_{ir}^{'} \leftarrow Disc_{score}(x_i^{'}, xl_{ir}^{'})$ using equation 5
7 Initialize the label change rate: $lcr_r = 0$
8 **for** $k = 1 : count$ **do**
9 $advpred_r = label(T_m(xl_{ir}^{'}))$
10 **if** $advpred_r \neq advpred$ **then**
11 $lcr_r = lcr_r + 1$
12 **end**
13 **end**
14 $AdvCont\_xl_{ir}^{'} \leftarrow AdvCont_{score}(xl_{ir}^{'}) = (count - lcr_r)/count$
15 **if** $(Disc\_xl_{ir}^{'} - AdvCont\_xl_{ir}^{'}) > 0$ **then**
16 $r^* = r$
17 **else**
18 break;
19 **end**
20 **end**
21 Obtain LFC at best radius r^*:
$x_{ic}^{'} = xl_{ir^*}^{'} = F^{-1}(LFC(F(x_i^{'}), r^*))$

r, the adversarial component would be the dominating factor resulting in the low-pass spatial sample $xl_{ir}^{'}$ to have the same prediction as $x_i^{'}$. To get a better empirical estimate we compute the low-pass predictions repeatedly, after enabling the dropout. Enabling dropout perturbs the decision boundary slightly. Thus if adversarial noise has started dominating, the shift in decision boundary will have no/very-less effect on the model's prediction. To quantify this effect, the LCR (at a particular radius r) captures the number of times the LFC prediction differs w.r.t original adversarial sample (Lines 7- 13) i.e. higher LCR implies low adversarial contamination and vice-versa. Line 14 enforces max LCR (i.e. min adversarial score, denoted by $AdvCont_{score}$) as 0 and min LCR (i.e. max $AdvCont_{score}$) as 1. This allows us to directly compare the adversarial and discriminability score. The optimal radius for $x_i^{'}$ is the maximum radius at which $Disc_{score}$ is greater than $AdvCont_{score}$ (Lines 15-20). The corrected i^{th} sample i.e. $x_{ic}^{'}$ (Line 21) is then passed to the model T_m to get the predictions.

5. Experiments

Different architectures such as Resnet-18 [15] and Resnet-34 [15] are used as the target model T_m . Each of these architectures is trained on two different datasets i.e. CIFAR-10 [19] and Fashion MNIST (FMNIST) [46] using the standard cross-entropy loss. CIFAR-10 is a colored dataset containing RGB images while FMNIST is grayscale. Once the model T_m is trained, its corresponding training set is not used any further due to the assumption of data privacy. We perform three different types of adversarial attacks on trained model Tm, namely PGD [31], IFGSM [21], and Auto Attack [9] (state-of-the-art) at test time. The images are normalized between 0 to 1 before perturbing them to create adversarial samples. The settings for the attack parameter are followed from [41]. For the attack on a model trained on CIFAR, the attack parameter ϵ is taken as $8/255$. Similarly for FMNIST, the ϵ parameter is 0.2. In the case of a PGD attack, the ϵ_{step} is $2/255$ and 0.02 while the number of iterations (N) is 20 and 100 for CIFAR and FMNIST respectively. The value of N remains the same for IFGSM attack while ϵ_{step} is ϵ/N for both the datasets. We perform separate analysis for each of our proposed modules in Sec. 5.1 (analysis of detection module) and Sec. 5.2 (analysis of correction module). Results for more datasets are included in supplementary.

5.1. Performance Analysis of Detection Module

To evaluate our detection module we perform extensive experiments with TinyImageNet as our arbitrary dataset ($D_{arbitrary}$). The source model F_s comprises of a ResNet-18 classifier (S_m) followed by a three-layer adversarial detector module (L_{advdet}). The stage-1 begins by training S_m on $D_{arbitrary}$ with the standard cross-entropy loss and stochastic gradient descent optimizer with $1e-3$ and 0.9 as the learning rate and momentum respectively. For stage-2 we obtain adversarial samples ($A_{arbitrary}$) by attacking S_m with PGD attack, the parameters for which are described in detail in the supplementary. Stage-3 involves training of our adversarial detector where we train the layers (L_{advdet}) using L_{BCE} loss. The input to the loss is predicted soft scores i.e. $L_{advdet}(S_m(x))$ where $x \in D_{arbitrary} \cup A_{arbitrary}$ and the ground-truth i.e. 1 (adversarial) and 0 (clean) respectively. Finally, to make our approach completely data-free we further reduce the dependency on even arbitrary datasets by formulating a source-free domain adaptation setup. Thus, the first three stages can be skipped given we have access to the pretrained detector on arbitrary data. To adapt the source-model F_s on our unlabelled target-set, we train F_t (frozen T_m model followed by trainable L_{advdet}) with L_{ent}, L_{div}, L_{pseudo} as described in Sec. 4.1. We use fixed values of δ and λ as 0.8 and 0.3 for all our UDA-based detection experiments. We perform experiments over multiple target datasets (CIFAR-10, F-MNIST) and target-model

Dataset	Model (T_m)	PGD			IFGSM			Auto-Attack		
		Overall	Clean	Adv.	Overall	Clean	Adv.	Overall	Clean	Adv.
CIFAR-10	ResNet-18	94.01	95.37	92.64	85.28	90.85	79.7	95.69	99.96	91.42
	Resnet-34	92.33	93.35	91.31	82.13	94.21	70.05	94.86	98.24	91.49
FMNIST	Resnet-18	86.23	97.24	75.22	89.17	96.28	82.05	85.52	98.28	72.75
	Resnet-34	89.65	99.43	79.87	90.06	99.51	80.62	86.37	98.32	74.41

Table 1. Proposed Detection Module Performance: Detection Accuracy (in %) on both clean and adversarial ('Overall'), clean samples ('Clean') and adversarial samples ('Adv.') for non robust target models trained on CIFAR-10 and Fashion MNIST with different architectures (Resnet-18 and Resnet-34).

Dataset	Model	Clean (B. A.)	PGD		IFGSM		Auto-Attack	
			Adv. (A. A.)	Ours (A. C.)	Adv. (A. A.)	Ours (A. C.)	Adv. (A. A.)	Ours (A. C.)
CIFAR	vgg-16	94.00	4.64	41.04 (36.4↑)	22.03	38.37 (16.34↑)	0.00	40.06 (40.06↑)
	resnet18	93.07	0.45	39.39 (38.94↑)	5.9	38.49 (32.59↑)	0.00	40.25 (40.25↑)
	resnet34	93.33	0.18	41.71 (41.53↑)	4.76	40.62 (35.86↑)	0.00	42.40 (42.40↑)
F-MNIST	vgg-16	91.79	0.62	33.09 (32.47↑)	6.43	33.83 (27.40↑)	0.00	37.99 (37.99↑)
	resnet18	90.31	2.95	32.22 (29.27↑)	7.64	32.38 (24.74↑)	0.00	35.80 (35.80↑)
	resnet34	90.82	1.85	33.23 (31.38↑)	5.57	33.73 (28.16↑)	0.00	35.82 (35.82↑)

Table 2. Notations: B.A. - Before Attack, A.A. - After Attack, A.C. - After Correction. Performance (in %) without and with our proposed correction module across VGG and Resnet architectures on different datasets i.e. CIFAR 10 and Fashion MNIST. The non-robust trained models performs poorly on adversarial samples which is significantly recovered through our correction method by achieving major boost in adversarial accuracy. The symbol (↑) denotes increment in performance by our method (A.C.) over (A.A.).

classifiers i.e. T_m (ResNet-18, ResNet-34) to demonstrate the effectiveness of our detection module. Refer the supplementary for results on other arbitrary datasets.

The experiment results are shown in Table 1. We can observe a high overall detection accuracy on a broad range of attacks, architectures (T_m), and datasets. We split the overall detection accuracy into adversarial and clean detection accuracies to better investigate the detector's performance. Our detection setup is a binary-classification problem, the adversarial detection accuracy can be understood as the True-Positive-Rate of our detector i.e. proportion of samples that are adversarial and correctly classified. Similarly, the clean detection accuracy is the True-Negative-Rate i.e. proportion of samples that are clean and classified correctly. In our experiments, we observe that we achieve a very high True-Negative-Rate (≈90%-99%) which is highly desirable in order to preserve the clean accuracy on T_m. The clean samples are directly passed to T_m, whereas the adversarially detected samples are first processed through the correction module that we describe in the next section.

5.2. Performance Analysis of Correction Module

Our correction module is training-free as we correct an incoming adversarially perturbed image by obtaining its LFC at optimal radius r^*. To reduce the computational complexity in calculating LFC using FT, we use FFT operations in the experiments. We calculate (as described in 4.2) the $AdvCont_{score}$ and $Disc_{score}$ to measure the discriminability and adversarial contamination for each LFC obtained at different radii. For obtaining the $Disc_{score}$ (eq. 5), we calculate the normalized SSIM-score using the open-source implementation provided by [40]. Our best radius r^* is selected as the maximum radius where $Disc_{score}$ is greater than $AdvCont_{score}$. The corrected image at r^* is obtained using eq. 4 where the imaginary part of F^{-1} output is discarded to enable the output to be fed to the trained model T_m to get the predictions.

In Table 2, we present the results for our correction module. The performance of the non-robust model T_m on clean and adversarially perturbed data is denoted by B.A. (Before Attack) and A.A. (After Attack) respectively. Assuming ideal detector, we pass each adversarial sample through our correction module and report the performance of corrected data as A.C. (After Correction). Most notably, we achieve a performance gain of $\approx 35 - 40\%$ on the state-of-the-art auto-attack across different architectures on multiple datasets. To further investigate the efficacy of our correction algorithm across different adversarial attacks, we also perform experiments on the widely popular PGD and IFGSM attacks, and obtain a similar boost in adversarial accuracy.

Estimating r^* accurately is especially important to our correction module's performance as we assume no knowledge about the training data. We verify this intuition by performing ablations on a "Random Baseline" (R.B.) wherein r^* is chosen randomly (within our specified range) for each sample. As shown in Table 3, R.B. although slightly higher than A.A., is significantly lower than A.C. that indicates the usefulness of selecting r^* appropriately.

Dataset	Model	PGD		IFGSM		Auto-Attack	
		R. B.	Ours	R. B.	Ours	R. B.	Ours
CIFAR	vgg-16	28.29	41.04	36.22	38.37	28.51	40.06
	resnet18	23.15	39.39	28.12	38.49	24.82	40.25
	resnet34	23.67	41.71	25.39	40.62	21.64	42.4
FMNIST	vgg-16	10.95	33.09	13.84	33.83	9.64	37.99
	resnet18	8.55	32.22	12.87	32.38	9.50	35.8
	resnet34	9.82	33.23	10.69	33.73	9.47	35.82

Table 3. Ablation on radius selection: Our proposed technique for radius selection leads to significant adversarial accuracy against Random baseline (R. B.) where the radius is selected randomly. The reported baseline performance is the mean over five trials.

6. Combined Detection and Correction

In this section, we discuss the performance on clean (D_{test}) and adversarially perturbed data (A_{test}) after combining our detection and correction modules as shown in Figure 1. Our combined module provides **D**ata-free **A**dversarial **D**efense (dubbed as '**DAD**') at test time in an end-to-end fashion without any prior information about the training data. DAD focuses on detecting and correcting adversarially perturbed samples to obtain correct predictions without modifying the pretrained classifier (T_m) which allows us to achieve a significant gain in adversarial accuracy without compromising on the clean accuracy, as shown in Figure 3. For instance, we improved the adversarial accuracy on Auto Attack by 39.63% with a minimal drop of 1.58% in the clean accuracy for ResNet-34 architecture on CIFAR-10 dataset. The combined performance for other attacks is provided in supplementary.

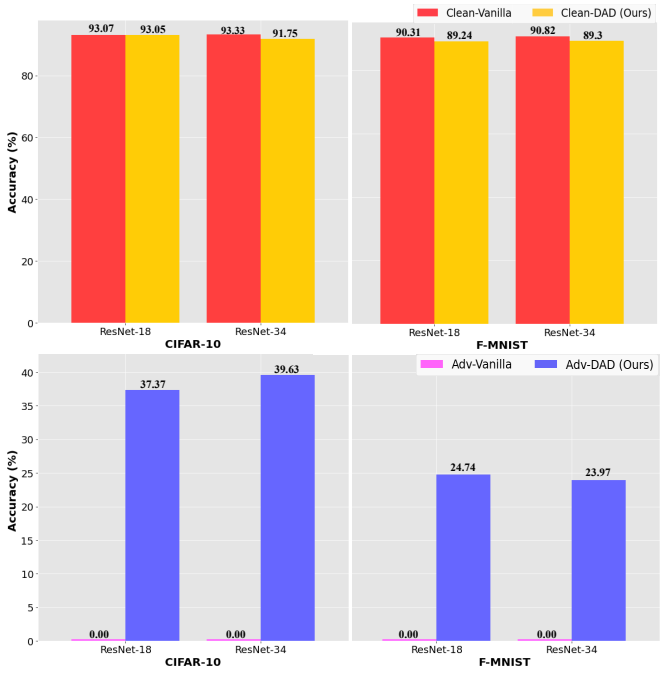

Figure 3. Performance comparison w/o (vanilla) and w/ our proposed detection and correction module to provide data-free adversarial defense (DAD) across different architectures and datasets on state-of-the-art Auto Attack.

We compare our proposed method, DAD (no training data used) with existing state-of-the-art data-dependent approaches in Table 4. We observe that our proposed method achieves decent adversarial accuracy in comparison to most of the data-dependent methods while maintaining a higher clean accuracy, entirely at test-time.

Method	Data-Free	Clean	Auto Attack
Zhang *et al*., 2019 [51]	✗	82.0	48.7
Sehwag *et al*., 2021 [37]	✗	84.38	54.43
Kundu *et al*., 2020 [20]	✗	87.32	40.41
Atzmon *et al*., 2019 [5]	✗	81.30	40.22
Moosavi-Dezfooli *et al*., 2019 [34]	✗	83.11	38.50
Baseline (at test time w/o our framework)	-	93.07	0
DAD (**Ours**)	✓	93.05	37.37

Table 4. Comparison of our **d**ata-free **a**dversarial **d**efense (DAD) with recent data-dependent approaches for resnet18 on CIFAR-10.

7. Conclusion

We presented for the first time a complete test time detection and correction approach for adversarial robustness in absence of training data. We showed the performance of each of our proposed modules: detection and correction. The experimental results across adversarial attacks, datasets, and architectures show the efficacy of our method. The combined module performance does not compromise much on the clean accuracy besides achieving significant improvement in adversarial accuracy, even against state-of-the-art Auto Attack. Our data-free method even gives quite competitive results in comparison to data-dependent approaches. Apart from these, there are other benefits associated with our proposed framework, as described below:

- Our detection module is independent of the correction module. Thus, any state-of-the-art classifier-based adversarial detector can be easily adopted on our source-free UDA-based adversarial detection framework.
- Any data-dependent detection approach can benefit from our correction module at test time to correct adversarial samples after successfully detecting them.

However, our adversarial detection method requires logits as input and hence is not strictly a black-box defense. Along with this, improving our radius selection algorithm to better estimate the optimal radius (r^*) are our future directions.

8. Acknowledgements

This work is supported by a Start-up Research Grant (SRG) from SERB, DST, India (Project file number: SRG/2019/001938).

References

[1] Face id security. https://www.apple.com/business-docs/FaceID_Security_Guide.pdf. accessed: November 2017.

[2] Sravanti Addepalli, Gaurav Kumar Nayak, Anirban Chakraborty, and Venkatesh Babu Radhakrishnan. Degan: Data-enriching gan for retrieving representative samples from a trained classifier. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3130–3137, 2020.

[3] Jonathan Aigrain and Marcin Detyniecki. Detecting adversarial examples and other misclassifications in neural networks by introspection. *arXiv preprint arXiv:1905.09186*, 2019.

[4] Jonathan Aigrain and Marcin Detyniecki. Detecting adversarial examples and other misclassifications in neural networks by introspection. *CoRR*, abs/1905.09186, 2019.

[5] Matan Atzmon, Niv Haim, Lior Yariv, Ofer Israelov, Haggai Maron, and Yaron Lipman. Controlling neural level sets. *arXiv preprint arXiv:1905.11911*, 2019.

[6] Ronald Newbold Bracewell and Ronald N Bracewell. *The Fourier transform and its applications*, volume 31999. McGraw-Hill New York, 1986.

[7] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018.

[8] Alvin Chan, Yi Tay, Yew Soon Ong, and Jie Fu. Jacobian adversarially regularized networks for robustness. *arXiv preprint arXiv:1912.10185*, 2019.

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

[10] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1625–1634, 2018.

[11] Reuben Feinman, Ryan R. Curtin, Saurabh Shintre, and Andrew B. Gardner. Detecting adversarial samples from artifacts. *CoRR*, abs/1703.00410, 2017.

[12] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[13] Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick D. McDaniel. On the (statistical) detection of adversarial examples. *CoRR*, abs/1702.06280, 2017.

[14] Paula Harder, F. Pfreundt, Margret Keuper, and Janis Keuper. Spectraldefense: Detecting adversarial attacks on cnns in the fourier domain. *ArXiv*, abs/2103.03000, 2021.

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

[16] Akinori Higashi, Minoru Kuribayashi, Nobuo Funabiki, Huy H Nguyen, and Isao Echizen. Detection of adversarial examples based on sensitivities to noise removal filter. In *2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1386–1391. IEEE, 2020.

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

[18] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 491–507. Springer, 2020.

[19] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Canadian Institute for Advanced Research, 2009.

[20] Souvik Kundu, Mahdi Nazemi, Peter A Beerel, and Massoud Pedram. Dnr: A tunable robust pruning framework through dynamic network rewiring of dnns. In *Proceedings of the 26th Asia and South Pacific Design Automation Conference*, pages 344–350, 2021.

[21] Alexey Kurakin, Ian Goodfellow, Samy Bengio, et al. Adversarial examples in the physical world, 2016.

[22] Vinod K Kurmi, Venkatesh K Subramanian, and Vinay P Namboodiri. Domain impression: A source data free domain adaptation method. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 615–625, 2021.

[23] Hyun Kwon, Yongchul Kim, Hyunsoo Yoon, and Daeseon Choi. Classification score approach for detecting adversarial example in deep neural network. *Multimedia Tools and Applications*, 80(7):10339–10360, 2021.

[24] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[25] Rui Li, Qianfen Jiao, Wenming Cao, Hau-San Wong, and Si Wu. Model adaptation: Unsupervised domain adaptation without source data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9641–9650, 2020.

[26] Bin Liang, Hongcheng Li, Miaoqiang Su, Xirong Li, Wenchang Shi, and Xiaofeng Wang. Detecting adversarial image examples in deep neural networks with adaptive noise reduction. *IEEE Transactions on Dependable and Secure Computing*, 2018.

[27] Bin Liang, Hongcheng Li, Miaoqiang Su, Xirong Li, Wenchang Shi, and Xiaofeng Wang. Detecting adversarial image examples in deep neural networks with adaptive noise reduction. *IEEE Trans. Dependable Secur. Comput.*, 18(1):72–85, 2021.

[28] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for un-

supervised domain adaptation. In *International Conference on Machine Learning*, pages 6028–6039. PMLR, 2020.

[29] Peter Lorenz, Paula Harder, Dominik Straßel, Margret Keuper, and Janis Keuper. Detecting autoattack perturbations in the frequency domain. In *ICML 2021 Workshop on Adversarial Machine Learning*, 2021.

[30] Xingjun Ma, Bo Li, Yisen Wang, Sarah M. Erfani, Sudanthi N. R. Wijewickrema, Grant Schoenebeck, Dawn Song, Michael E. Houle, and James Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[31] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[32] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[33] Charlotte Middlehurst. China unveils world's first facial recognition atm. `http://www.telegraph.co.uk/news/worldnews/asia/china/11643314/China-unveils-worlds-first-facial-recognition-ATM.html`. June 2015.

[34] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. Robustness via curvature regularization, and vice versa. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9078–9086, 2019.

[35] Rafael Müller, Simon Kornblith, and Geoffrey Hinton. When does label smoothing help? *arXiv preprint arXiv:1906.02629*, 2019.

[36] Gaurav Kumar Nayak, Konda Reddy Mopuri, Vaisakh Shaj, Venkatesh Babu Radhakrishnan, and Anirban Chakraborty. Zero-shot knowledge distillation in deep networks. In *International Conference on Machine Learning*, pages 4743–4751. PMLR, 2019.

[37] Vikash Sehwag, Saeed Mahloujifar, Tinashe Handina, Sihui Dai, Chong Xiang, Mung Chiang, and Prateek Mittal. Improving adversarial robustness using proxy distributions. *arXiv preprint arXiv:2104.09425*, 2021.

[38] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[39] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708, 2014.

[40] VainF. Vainf/pytorch-msssim: Fast and differentiable ms-ssim and ssim for pytorch. `https://github.com/VainF/pytorch-msssim`.

[41] BS Vivek, Ambareesh Revanur, Naveen Venkat, and R Venkatesh Babu. Plug-and-pipeline: Efficient regularization for single-step adversarial training. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 138–146. IEEE, 2020.

[42] Haohan Wang, Xindi Wu, Zeyi Huang, and Eric P Xing. High-frequency component helps explain the generalization of convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8684–8694, 2020.

[43] Si Wang, Wenye Liu, and Chip-Hong Chang. Detecting adversarial examples for deep neural networks via layer directed discriminative noise injection. In *2019 Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*, pages 1–6. IEEE, 2019.

[44] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[45] Zifan Wang, Yilin Yang, Ankit Shrivastava, Varun Rawal, and Zihao Ding. Towards frequency-based explanation for robust cnn. *arXiv preprint arXiv:2005.03141*, 2020.

[46] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[47] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.

[48] Puyudi Yang, Jianbo Chen, Cho-Jui Hsieh, Jane-Ling Wang, and Michael Jordan. Ml-loo: Detecting adversarial examples with feature attribution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6639–6647, 2020.

[49] Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deepinversion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8715–8724, 2020.

[50] Jaemin Yoo, Minyong Cho, Taebum Kim, and U Kang. Knowledge extraction with no observable data. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 2705–2714, 2019.

[51] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, pages 7472–7482. PMLR, 2019.

[52] Yiman Zhang, Hanting Chen, Xinghao Chen, Yiping Deng, Chunjing Xu, and Yunhe Wang. Data-free knowledge distillation for image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7852–7861, 2021.