PERCEPTUAL DEEP NEURAL NETWORKS: ADVERSARIAL ROBUSTNESS THROUGH INPUT RECREATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Adversarial examples have shown that albeit highly accurate, models learned by machines, differently from humans, have many weaknesses. However, humans’ perception is also fundamentally different from machines, because we do not see the signals which arrive at the retina but a rather complex recreation of them. In this paper, we explore how machines could recreate the input as well as investigate the benefits of such an augmented perception. In this regard, we propose Perceptual Deep Neural Networks (φDNN) which also recreate their own input before further processing. The concept is formalized mathematically and two variations of it are developed (one based on inpainting the whole image and the other based on a noisy resized super resolution recreation). Experiments reveal that φDNNs and their adversarial training variations can increase the robustness substantially, surpassing both state-of-the-art defenses and pre-processing types of defenses in 100% of the tests. φDNNs are shown to scale well to bigger image sizes, keeping a similar high accuracy throughout; while the state-of-the-art worsen up to 35%. Moreover, the recreation process intentionally corrupts the input image. Interestingly, we show by ablation tests that corrupting the input is, although counter-intuitive, beneficial. Thus, φDNNs reveal that input recreation has strong benefits for artificial neural networks similar to biological ones, shedding light into the importance of purposely corrupting the input as well as pioneering an area of perception models based on GANs and autoencoders for robust recognition in artificial intelligence.

1 INTRODUCTION

Recent work has revealed that albeit highly accurate, deep neural networks are far from robust Szegedy et al. (2014). The lack of robustness exist even for extremely small perturbations and simple transformations Madry et al. (2018); Engstrom et al. (2017); Su et al. (2019). A wide range of defenses were proposed in recent years Goodfellow et al. (2015); Grosse et al. (2017); Li & Li (2017); Metzen et al. (2017); Ma et al. (2018). However, most of them have shortcomings such as relying on obfuscated gradients Athalye et al. (2018) or being biased by the type of perturbation used to train (e.g., adversarial training) Kurakin et al. (2017); Kannan et al. (2018).

Humans are less affected by small changes in the input. Interestingly, this is true even when part of the input is completely removed; which happens every second. Each of our eyes have a blind spot1 where light cannot be perceived. Albeit this limitation, when we close one eye we do not see a black spot but a completely filled perception of an image De Weerd et al. (1995); Komatsu (2006). This is an example of how the brain is always predicting what it is viewing, revealing that biological perceptual systems are active rather than passive McClelland & Rumelhart (1981). Thus, the images we see every second is rather a creation than mere signals that arrived in the brain, also called perception filling-in and related to predictive coding Clark (2013); Rao & Ballard (1999); Ehinger et al. (2017). In this context, we raise the following question:

Could deep neural networks also benefit from actively creating its own input?

1The blind spot in each eye is where the optic nerve passes through the optic disc and therefore no photoreceptor cells are present.
Figure 1: Illustration of the proposed $\varphi$DNN architecture and its similarity to the filling-in phenomena in human vision. Input $x$ is initially corrupted by $\delta(x)$, removing some information while keeping contextual clues. $p(\delta(x); \theta)$ uses this corrupted image to predict a partial (or whole) recreation which is then aggregated with function $\mu$ to compose a complete recreated image. This recreated image is sent to the vanilla classifier.

To answer the question above we developed two perceptual systems that recreate the input image with predictions of it. One is based on inpainting all parts of the image while the other is based on recreating a super resolution of the image and then resizing it (Section 3.2). The recreated input is then fed to a deep neural network which has no access to the original input (Figure 1). Attacks on both systems suggest that by recreating the input, robustness against adversarial attacks increase. Furthermore, the input recreation is not mutually exclusive with many of the previous defenses. It can be used together with adversarial training, for example, to improve further robustness.

**OUR CONTRIBUTIONS.** In this paper, we present input recreation as a novel paradigm to enhance robustness against adversarial samples. The key contributions can be summarized as follows:

- We introduce deep neural networks (DNN) that recreate their own input based solely on contextual hints, called perceptual DNNs ($\varphi$DNN). We describe $\varphi$DNNs formally and conduct experiments on two different implementations of it.

- We propose an inpainting based $\varphi$DNN. It works by predicting removed parts of the image and then joining the predicted parts together into a single completely recreated image. This recreated image is then used as input to a DNN.

- We propose a super resolution based $\varphi$DNN which recreates a higher resolution version of the input excluding at the same time any noise present in the original one. The image is later resized and inputted to a DNN.

- The results suggest that approaches with active perceptual systems recreating their own input can achieve higher robustness than their counterparts. This is true not only for the best performing system but most of its numerous variations, revealing a strength of the approach. Moreover, $\varphi$DNNs can be used jointly with other defenses to increase robustness further.

- Experiments reveal that, for neural networks able to recreate their own input, always purposely corrupting the input (for both training and testing) is mostly beneficial.
2 Related Works

Attack Methods. In this paper, we make use of attack methods for the sole purpose of evaluating the robustness of defenses and neural network models. Several attack models have been proposed in recent studies. They can be broadly categorized into white box [Szegedy et al. (2014); Goodfellow et al. (2015); Madry et al. (2018); Carlini & Wagner (2017)] and black box attacks [Papernot et al. (2017); Brendel et al. (2018); Ilyas et al. (2018); Tu et al. (2019); Dong et al. (2019)].

Many white box models can be summarized as follows. Given a target classifier \( C \) and an input pair \((x, y)\). Let \( L \) be the adversarial loss for the classifier \( C(x') \) e.g., the cross-entropy loss, and the \( \ell_p \) norm used to measure the distance between the legitimate input \( x \) and the adversarial input \( x' \).

Generally, white box attack methods have been proposed by solving the constrained optimization problem:

\[
\min_{x'} L(C(x'), y), \quad \text{s.t.} \quad \|x - x\|_p \leq \epsilon. \tag{1}
\]

Examples of white box attacks are FGSM [Goodfellow et al. (2015), one of the earliest white box attacks, which uses one-step approach to determine the direction to change the pixel value, and an improved method called projected gradient descent (PGD) with a multiple-step variant [Madry et al. (2018)]. In contrast, black box attacks have been proposed under more critical and practical conditions with the trade-off of being slower. Here, we are also interested in black box attacks which are not based on estimating gradients and therefore can find adversarial samples even when the gradient is masked [Athalye et al. (2018)]. Therefore, tests with more straightforward black box attack methods based on evolutionary strategy such as the one-pixel attack and few-pixel attack fits the purpose [Su et al. (2019)].

Defenses to Adversarial Attacks. Recent studies have proposed various defense mechanisms against the threat of adversarial attacks. Albeit recent efforts, there is not yet a completely effective method. Defensive distillation, for example, proposed a smaller neural network which squeezed the content learned by the original one [Papernot et al. (2016)], however, it was shown to lack robustness in a later paper [Carlini & Wagner (2017)]. Adversarial training which was firstly proposed by Goodfellow et al. [Goodfellow et al. (2015) increases the robustness by adding adversarial examples to the training set [Huang et al. (2015); Madry et al. (2018)]. Similarly, adversarial training was also shown vulnerable to attacks in [Tram`er et al. (2018)]. Other defenses include preprocessing defenses such as the feature squeezing (FS) and spatial smoothing (SS) [Xu et al. (2017)]. The objective here is to remove adversarial perturbation in a pre-processing stage. Recently there are a huge number of defenses proposed, however, they use mostly variations of gradient masking to avoid being attacked which do not confer greater security [Athalye et al. (2018)].

Regarding GAN based defenses, Defense-GAN [Samangouei et al. (2018)] is based on training a generative adversarial network (GAN) to learn the distribution of original images. Each input would then be used to search for the closest projected input image learned by the generator before proceeding to classification. One of the main shortcomings is that the distribution learned by the generator is strictly limited by the training data set and the input image might be mapped into an illegitimate space. Albeit using GANs in our proposed approach, it shares no other similarities to Defense-GAN. Here, GANs predict parts of the input using the contextual information present, and only after the input has been purposely corrupted.

Predictive Coding. Although \( \varphi \)DNNs do not necessarily use many of the components of predictive coding, it is loosely based on it. Predictive coding is a theory in neuroscience which postulates that the brain achieves high visual robustness by dynamically updating and predicting neural activities from the environment [Pennartz (2015)]. Previous studies have shown that the brain uses similar representations to CNNs, but CNNs are not as robust as the brain [Cichy et al. (2016); Wen et al. (2018b)]. Though there is still no perfect theoretical explanation for how it works, biological plausible models describe it as a recurrently connected hierarchical neural networks [Sporns & Zwi (2004)]. Recent research on predictive coding based CNNs imitating the feedforward, feedback, and recurrent connections performed well in object recognition tasks [Wen et al. (2018a)].

This work makes use of both Generative Adversarial Network (GAN) and AutoEncoder (AE) to recreate the images. They are described briefly as follows.
2.1 Generative Adversarial Network

Generative Adversarial Network (GAN) is a powerful generative model that consists of two neural networks: a generator network which learns the probability distribution of the input and a discriminator network which distinguishes between generated data and the input data [Goodfellow et al. (2014)].

Super-resolution GAN (SRGAN). Super-resolution GAN (SRGAN) generates a photo-realistic high-resolution (HR) image from its downsampled low-resolution (LR) input image. In [Ledig et al. (2017)], they used VGG-19 network to extract high dimension features and designed an alternative function, the perceptual loss function, which consists of content loss and adversarial loss to solve the following min-max optimization problem:

$$\min_G \max_D \mathbb{E}_{I_{HR} \sim p_{\text{train}}(I_{HR})}[\log D(I_{HR})] + \mathbb{E}_{I_{LR} \sim p_G(I_{LR})}[\log (1 - D(G(I_{LR})))]$$

Here, the generative model $G$ maps a given LR input $I_{LR}$ to its HR counterpart $I_{HR}$. The discriminator $D$ is trained to distinguish between the produced $I_{HR}$ images from real inputs.

2.2 Autoencoder Based Inpainting

Inpainting is defined as the synthesis of content to fill missing image parts. Here, we use an AE to predict the missing pixels with a simple UNET-like architecture [Ulyanov et al. (2018); Ronneberger et al. (2015)]. Let a masked image $x_0$ be represented as $x_0 = x \odot (1 - M)$, in which $M$ is a binary mask, $x$ is the original input image and $\odot$ is the element-wise product operation. Inpainting can be formulated as the following energy minimization problem:

$$\min_{\theta} E(F(x_0; \theta); x),$$

$$E(F(x_0; \theta); x) = |F(x_0; \theta) - x|,$$

where $F$ is the resulting function from the AE with parameters $\theta$.

3 $\varphi$DNNs

In this section, we describe formally the $\varphi$DNN architecture, its motivation as well as two different implementations of it.

3.1 Technical Motivation behind Input Recreation

Beyond the bio-inspired aspect, there are some technical importance for recreating the input in a similar way to humans and other animals. First, by recreating the input, the neural network and not the environment defines which input will be responsible for the output of the system. This type of actively modified input provides further control of the input to avoid contextual problems or other issues beyond adversarial samples. Second, it is now possible to constrain the probability distribution of the input further. This can be done in many ways and is only slightly explored here with added noise. Third, with perceptual changes happening all the time, attacking becomes a time-varying function which might be impossible to repeat. This would make calculated attacks near impossible. Fourth, when facing $\varphi$DNNs, the attacker has less information about the network for he/she does not know even the input now. Lastly, gradient-based and gradient estimation based approached tend to perform poorly if the input is changed substantially by the recreation process.

3.2 $\varphi$DNN’s Architecture

Consider the perceptual tuple $\varphi$ and its respective function $\varphi(x)$ as follows:

$$\varphi := \langle \delta, p(\delta(x); \theta), \mu \rangle,$$

$$\varphi(x) = \mu(p(\delta(x); \theta)),$$

where $\delta$ is a function that corrupts the input, removing some information from it and returning one or multiple corrupted images; $p(\delta(x); \theta)$ is the probability distribution learned by a model that predicts
\( x \) from the corrupted input \( \delta(x) \) based on its learned weights \( \theta \); and \( \mu \) is the aggregation function which joins partial recreations (when present) into a single recreated image.

\( \varphi \)DNN is defined as follows:

\[
\varphi \text{DNN} := C(\varphi(x)),
\]

(4)

in which \( C \) is a classifier that receives as input the output from the perceptual function \( \varphi(x) \).

**Noisy Super-Resolution Reconstruction (NSR)** Here we define an implementation of the \( \varphi \)DNN’s architecture using super resolution and images corrupted with noise. Note that images are always corrupted with noise (i.e., in both training and testing). Let \( x \) be a given input and \( R \) a function which resizes the high resolution image to the original resolution. The process can be defined as follows:

\[
\varphi_r := \langle \delta_r, p_r(\delta(x); \theta_r), \mu_r \rangle,
\]

(5)

where \( \delta_r = \text{Noise}, p_r = G(\delta(x); \theta), \mu_r = R \).

Note that \( \text{Noise} \) is an arbitrary noise function which returns a noisy image. \( G(\delta(x); \theta) \) is the generator of SRGAN which maps an image from low resolution to high resolution and tries to clean the always present noise (illustrated in Figure 2).

**Full Inpainting Reconstruction (FIR)** To demonstrate that \( \varphi \)DNNs can be developed in many forms, here, we propose a \( \varphi \)DNN based on inpainting the whole image. Specifically, \( \varphi_i \) is defined as follows.

\[
\varphi_i := \langle \delta_i, p_i(\delta(x); \theta_i), \mu_i \rangle,
\]

(6)

where

\[
\delta_i(x) = \bigcup_{k=0}^{9} \odot (1 - M_k) \cdot x,
\]

\[
p_i = F(\delta(x); \theta), \mu_i(x) = \sum_{k=0}^{9} M_k \cdot x,
\]

where \( M_k \) are masks such that their sum is equal to the identity matrix \( \left( \sum_{k=0}^{9} M_k = I \right) \) and their multiplication is equal to 0 \( \left( \prod_{k=0}^{9} M_k = 0 \right) \). Therefore, each of the masks hide a specific part of the image; and together they mask the whole image. \( \delta_i(x) \) (i.e., \( \bigcup_{k=0}^{9} \odot (1 - M_k) \cdot x \)) creates a set with 10 masked inputs. All masked inputs are then inpainted with \( F(\delta(x); \theta) \) and lastly all inpainted parts are joined together through \( \sum_{k=0}^{9} M_k \cdot x \). Figure 2 shows an illustration of the process.
4 EXPERIMENTS

To evaluate $\varphi$DNN architecture, we test here the robustness of two implementations of it (i.e., FIR and NSR) by attacking them with different types of attacks. The proposed architecture is also compared with other defenses in three datasets (CIFAR, SVHN and a subset of Imagenet called Imagenette [Shleifer & Prokop 2019]).

To evaluate the robustness of systems avoiding biases and the sole presence of gradient masking, we employ two white box attacks (FGSM and PGD) as well as non-gradient based black box attacks (one pixel and ten pixel attack). In this paper, every attack is repeated for 500 uniformly sampled random images of the test data set with the average attack accuracy being reported. For all experiments, the CIFAR-10 dataset is normalized to the range $[0, 1]$. The machines used in the experiments are equipped with NVIDIA GeForce RTX 2080 Ti and AMD Ryzen 9 3950x 16-core.

Regarding FIR, 10 masks are created, each of them removing 10% of the image. (Figure 3). To create the masks, a grid of a given size is set over the $32 \times 32$ image and then multiple pieces of this grid are randomly selected to form one mask. Pieces are selected until 10% of the image is covered. The inpainting model is trained with a corresponding mask size covering 10% of the image and with epochs and batch size of respectively 20 and 32.

![Figure 3: Three masks created for FIR with grid size of respectively (from left to right) 1, 4 and 8.](image)

Regarding NSR, to create the training dataset for SRGAN we resize CIFAR-10 dataset to $128 \times 128$ as the high resolution ground truth and add noise to the training dataset during training. The type of noise used is bi-linear interpolation for both up and downsizing. In order to match with the normalization, we replace tanh with sigmoid as the activation function for the last convolution layer in SRGAN’s generator. We train SRGAN with 1000 epochs and set batch size to 20 to ensure convergence.

4.1 COMPARISON WITH OTHER DEFENSES

Table 1 compares the last development in adversarial training, i.e. Feature Scattering [Zhang & Wang 2019], with the proposed algorithms and variations of them trained with a simple adversarial training. Other state-of-the-art defenses such as LLR and TRADES are also included in some of the tests, using the original results reported from their papers [Qin et al. 2019; Zhang et al. 2019].

Results show that both $\varphi$-DNNs with adversarial training surpass FScattering for all of the attacks (the only exception is the SVHN test in which $\text{FIR}_{adv}$ gets 80.9% against 81.4% achieved by FScattering). It is known that adversarial training methods such as FScattering perform poorly when the attacking distribution differ from the data used to learn. This applies to FScattering as well which can be attacked with more than 50% attack accuracy with 10px attack. Having said that, it is impressive that both NSR and FIR can surpass FScattering even on FGSM and PGD which are close to the augmented distribution of noisy images FScattering used to learn. Notice that the same adversarial training that has little change on the vanilla Resnet (i.e., $\text{ResNet}_{adv}$) is very effective on NSR and FIR. For example, in CIFAR under a PGD attack, $\text{FIR}_{adv}$ is 79.5% accurate against a 6.3% accuracy of the vanilla FIR and a 0% accuracy of the same adversarial training applied on a vanilla ResNet. Thus, it is expected that if a state-of-the-art adversarial training is applied to NSR and FIR, their robustness should improve even further. Pruned networks were also added to demonstrate that lower accuracy pruned neural networks are not comparable with current defenses.

In fact, if we take into account that FScattering and $\varphi$DNN are (a) different in nature and (b) can be also used together. It can be justified that $\varphi$DNNs should be compared with other pre-processing defenses and not adversarial training ones. We follow this rational and compare the proposed methodology in Table 2 with other pre-processing defenses such as FS, SS, JPEG compression defence...
Table 1: Comparison between Feature Scattering (the current state-of-the-art defense) and the two proposed $\varphi$-DNNs with a simple adversarial training ($\text{NSR}_{adv}$ and $\text{FIR}_{adv}$) on CIFAR10, SVHN and Imagenette. Other state-of-the-art defenses such as LLR and TRADES are also included. Results show the accuracy of defenses under attack. For reference, we include the $\varphi$-DNNs without adversarial training (NSR and FIR), only ResNet with the same simple adversarial training used on $\text{NSR}_{adv}$ and $\text{FIR}_{adv}$ (ResNet$_{adv}$); as well as vanilla ResNet. ResNet$_{pruned}$ means the trained ResNet is pruned 50%, retrained and then ultimately pruned to 80%.

<table>
<thead>
<tr>
<th>Defense</th>
<th>test acc</th>
<th>1px attack</th>
<th>10px attack</th>
<th>FGSM ($\epsilon = 8$)</th>
<th>PGD ($\epsilon = 8$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Imagenette</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ours: NSR$_{adv}$</td>
<td>77.0</td>
<td>75.3</td>
<td>75.6</td>
<td>57.3</td>
<td>52.7</td>
</tr>
<tr>
<td>Ours: FIR$_{adv}$</td>
<td>78.1</td>
<td>73.7</td>
<td>62.9</td>
<td>75.0</td>
<td>66.8</td>
</tr>
<tr>
<td>FScattering</td>
<td>72.4</td>
<td>59.5</td>
<td>52.2</td>
<td>42.3</td>
<td>43.2</td>
</tr>
<tr>
<td>NSR</td>
<td>81.7</td>
<td>77.5</td>
<td><strong>77.6</strong></td>
<td>43.4</td>
<td>48.9</td>
</tr>
<tr>
<td>FIR</td>
<td>85.4</td>
<td><strong>81.1</strong></td>
<td>76.9</td>
<td>13.4</td>
<td>0.2</td>
</tr>
<tr>
<td>CIFAR10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ours: NSR$_{adv}$</td>
<td>82.7</td>
<td>78.1</td>
<td><strong>73.9</strong></td>
<td>80.3</td>
<td><strong>79.6</strong></td>
</tr>
<tr>
<td>Ours: FIR$_{adv}$</td>
<td>88.5</td>
<td>73.6</td>
<td>50.4</td>
<td><strong>86.2</strong></td>
<td>79.5</td>
</tr>
<tr>
<td>FScattering</td>
<td>90.0</td>
<td>72.2</td>
<td>47.7</td>
<td>78.4</td>
<td>70.5</td>
</tr>
<tr>
<td>TRADES</td>
<td>87.4</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>51.6</td>
</tr>
<tr>
<td>LLR</td>
<td>86.8</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>54.2</td>
</tr>
<tr>
<td>SVHN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ours: NSR$_{adv}$</td>
<td>92.4</td>
<td>89.1</td>
<td>87.7</td>
<td>85.6</td>
<td>87.9</td>
</tr>
<tr>
<td>Ours: FIR$_{adv}$</td>
<td>93.0</td>
<td>80.9</td>
<td>53.9</td>
<td><strong>93.0</strong></td>
<td><strong>91.5</strong></td>
</tr>
<tr>
<td>FScattering</td>
<td>96.2</td>
<td>81.4</td>
<td>46.0</td>
<td>83.5</td>
<td>52.0</td>
</tr>
<tr>
<td>NSR</td>
<td>93.0</td>
<td><strong>90.2</strong></td>
<td><strong>90.1</strong></td>
<td>80.1</td>
<td>88.4</td>
</tr>
<tr>
<td>FIR</td>
<td>96.4</td>
<td>80.4</td>
<td>57.1</td>
<td>52.6</td>
<td>9.95</td>
</tr>
</tbody>
</table>

Table 2: Comparison of proposed methods with other preprocessing based defenses. NSR and FIR models use the best setting from Tables 3 and 4 while the other ones use AllConv and the best settings out of a couple of experiments.

<table>
<thead>
<tr>
<th>Defense</th>
<th>test acc</th>
<th>1px attack</th>
<th>10px attack</th>
<th>FGSM ($\epsilon = 8$)</th>
<th>PGD ($\epsilon = 8$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours: NSR</td>
<td>83.8</td>
<td>78.3</td>
<td>73.9</td>
<td><strong>68.0</strong></td>
<td>74.2</td>
</tr>
<tr>
<td>Ours: FIR</td>
<td>89.4</td>
<td>73.2</td>
<td>54.2</td>
<td>25.5</td>
<td>6.3</td>
</tr>
<tr>
<td>FS</td>
<td>79.2</td>
<td>42.7</td>
<td>10.5</td>
<td>23.8</td>
<td>11.2</td>
</tr>
<tr>
<td>SS</td>
<td>78.6</td>
<td>69.6</td>
<td>38.8</td>
<td>42.8</td>
<td>28.9</td>
</tr>
<tr>
<td>JPEG</td>
<td>73.0</td>
<td>21.0</td>
<td>1.5</td>
<td>47.1</td>
<td>41.2</td>
</tr>
<tr>
<td>LS</td>
<td>91.4</td>
<td>48.6</td>
<td>15.0</td>
<td>54.4</td>
<td>4.1</td>
</tr>
</tbody>
</table>

(JPEG) [Das et al. (2017)] and Label Smoothing (LS) [Hazan et al. (2016)]. All defenses used ResNet with the same type of augmentation. Note that we also tried to include DefenseGAN but it failed to learn properly on CIFAR10.

Both $\varphi$-DNNs surpass all others in most of the attacks. The result is expected since $\varphi$-DNNs do not only pre-process images, they recreate them based on contextual information and previous learned distribution. The only exception is FIR for FGSM and PGD (Table 2). FIR’s poor results on PGD and FGSM are less obvious. It is related to the grid size which is discussed in Section 4.2.
Table 3: Attack accuracy for both NSR and SR (NSR without the added noise $\delta_r(x)$) trained with different types of noise and connected to ResNet. We tested Gaussian noise with 0 mean ($\mu$), and variances ($\sigma^2$) of 0.01. For Panda noise, the scalar number (0.01) represents the probability ($\alpha$ and $\beta$) of white and black pixels present in the image. $A + B$ represents that two types of noises $A$ and $B$ are summed together. The subscript $T$ means that the classifier was retrained with a data set made of recreated images (i.e., images from $\varphi_r(x)$).

<table>
<thead>
<tr>
<th>Defense</th>
<th>Noise</th>
<th>test acc</th>
<th>10px attack</th>
<th>FGSM ($\epsilon = 8$)</th>
<th>PGD ($\epsilon = 8$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NSR</td>
<td>+ResNet[Guassian0.01]</td>
<td>79.2</td>
<td>60.5</td>
<td>73.0</td>
<td>75.5</td>
</tr>
<tr>
<td>SR</td>
<td>+ResNet[Guassian0.01]</td>
<td>77.4</td>
<td>16.2</td>
<td>73.0</td>
<td>75.0</td>
</tr>
<tr>
<td>NSR</td>
<td>+ResNet[Panda0.01]</td>
<td>91.0</td>
<td>49.1</td>
<td>68.2</td>
<td>72.2</td>
</tr>
<tr>
<td>SR</td>
<td>+ResNet[Panda0.01]</td>
<td>91.5</td>
<td>30.8</td>
<td>33.2</td>
<td>17.7</td>
</tr>
<tr>
<td>NSR</td>
<td>+ResNet[Guassian+Panda]</td>
<td>77.0</td>
<td>64.5</td>
<td>70.0</td>
<td>72.2</td>
</tr>
<tr>
<td>SR</td>
<td>+ResNet[Guassian+Panda]</td>
<td>81.9</td>
<td>37.9</td>
<td>72.4</td>
<td>78.4</td>
</tr>
<tr>
<td>NSR</td>
<td>+ResNet[Guassian+Panda]_T</td>
<td>83.8</td>
<td>74.6</td>
<td>68.6</td>
<td>74.2</td>
</tr>
<tr>
<td>SR</td>
<td>+ResNet[Guassian+Panda]_T</td>
<td>84.5</td>
<td>44.6</td>
<td>67.6</td>
<td>76.5</td>
</tr>
</tbody>
</table>

Table 4: Comparing the difference of grid size on FIR’s accuracy and robustness. ResNet is the vanilla classifier while FIR$^+_1$, FIR$^+_4$ and FIR$^+_8$ means using ResNet in the FIR’s architecture with grid size of respectively 1, 4 and 8. Each inpainting model is trained with the corresponding grid size only, and the classifier model is trained with corresponding inpainting image from $\varphi_i(x)$.

<table>
<thead>
<tr>
<th></th>
<th>Test accuracy</th>
<th>1px attack</th>
<th>10px attack</th>
<th>FGSM ($\epsilon = 8$)</th>
<th>PGD ($\epsilon = 8$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet</td>
<td>93.0</td>
<td>56.0</td>
<td>18.2</td>
<td>13.6</td>
<td>0.0</td>
</tr>
<tr>
<td>FIR$^+_1$</td>
<td>89.4</td>
<td>73.2</td>
<td>54.2</td>
<td>25.5</td>
<td>6.3</td>
</tr>
<tr>
<td>FIR$^+_4$</td>
<td>82.7</td>
<td>69.2</td>
<td>43.2</td>
<td>47.9</td>
<td>47.5</td>
</tr>
<tr>
<td>FIR$^+_8$</td>
<td>73.5</td>
<td>59.0</td>
<td>25.9</td>
<td>49.0</td>
<td>53.9</td>
</tr>
</tbody>
</table>

NSR corrupts the input image possibly losing some information. Here we will investigate if this loss of information has any deleterious consequences. We will also analyze the behavior of NSR on adversarial samples. To analyze the influence of the initial input corruption by $\delta_r(x)$, an ablation test is made, in which $\delta_r(x)$ is removed from $\varphi_r(x)$ (this algorithm is called SR in Table 3). Results show an increased robustness and similar accuracy. Specifically, in 8 out of 12 tests, the robustness of NSR surpassed the ablated algorithm SR. Regarding the accuracy, both NSR and SR performed similarly.

These results reveal, perhaps counter-intuitively, that always adding noise ($\delta_r(x)$) to the input is mostly beneficial for neural networks that recreate their input. On average, it usually improves robustness while leaving accuracy unchanged. There are two reasons for such a behavior: (a) always adding noise constrains the image distribution to non-smooth pixel transitions and (b) an always changing input is harder to attack.

4.2 FIR Analysis

In this section, FIR will be analyzed with relation to its grid size. For L0 attacks (1px, 5px and 10px attacks), FIR performs better with lower grid values while higher grid values are better suited to L$\infty$ attacks (PGD and FGSM) (Table 4). This is expected since L0 attacks perturbs fewer pixels and therefore punctual corrections are better. The opposite is true for L$\infty$. Attacking FIR is difficult because for a pixel to be modified in the final image $\varphi_i(x)$, many pixels around it must be changed (for lower grid values) or pixels near the mask needs to be changed (for higher grid values). This creates a bigger burden on the attacker and causes many attacks to fail. A simple version of adversarial training (with FGSM created adversarial samples) improves substantially the advantages of FIR, allowing it to even surpass the state-of-the-art (Table 1).
5 CONCLUSIONS

In this paper, we proposed ϕDNNs which make use of corrupting functions and context-based prediction to recreate the input. We showed that, perhaps surprisingly, corrupting the input is beneficial to robustness. Moreover, two implementations of ϕDNNs surpassed the state-of-the-art while possessing much better scaling for datasets with bigger image sizes. Note that the implementations used here do not utilize state-of-the-art GANs or AEs as well as only a classic adversarial training scheme. Therefore, state-of-the-art GANs/AEs/adversarial training should increase further the results reported here, which are already surpassing the state-of-the-art in all of the tests.

Thus, this paper proposes a novel paradigm for robust neural networks with state-of-the-art results which should, hopefully, incentive further investigation into ϕDNNs and other perception models. It also opens new paths for robust artificial intelligence, towards safer applications.

REFERENCES


Under review as a conference paper at ICLR 2021


