An Empirical Evaluation of Perturbation-based Defenses

Adam Dziedzic 1  Sanjay Krishnan 1

Abstract
Recent work has extensively shown that randomized perturbations of a neural network can improve its robustness to adversarial attacks. The literature is, however, lacking a detailed compare-and-contrast of the latest proposals to understand what classes of perturbations work, when they work, and why they work. We contribute a detailed experimental evaluation that elucidates these questions and benchmarks perturbation defenses in a consistent way. In particular, we show five main results: (1) all input perturbation defenses, whether random or deterministic, are essentially equivalent in their efficacy, (2) such defenses offer almost no robustness to adaptive attacks unless these perturbations are observed during training, (3) a tuned sequence of noise layers across a network provides the best empirical robustness, (4) attacks transfer between perturbation defenses so the attackers need not know the specific type of defense only that it involves perturbations, and (5) adversarial examples very close to original images show an elevated sensitivity to perturbation in a first-order analysis. Based on these insights, we demonstrate a new robust model built on noise injection and adversarial training that achieves state-of-the-art robustness.

1. Introduction
The attacks on Convolutional Neural Networks, such as Carlini & Wagner (Carlini & Wagner, 2017) or PGD (Madry et al., 2017), generate strategically placed modifications to induce a deliberate misprediction. Recently, there have been a number of defenses against such attacks that use perturbation during inference time and/or training to improve robustness (Dziugaite et al., 2016; Roth et al., 2019; Yang et al., 2019). The success of perturbation-based defenses suggests that adversarial examples are not robust themselves, where small amount of noise can dominate the strategically placed perturbations rendering them ineffective. However, a detailed understanding of this phenomenon is lacking from the research literature including: (1) what types of perturbations work and what is their underlying mechanism, (2) is this property specific to certain attacks and architectures, and (3) whether such defenses are effective and how to make them effective.

Researchers began to notice this property a few years ago with the a number of inference-time “input perturbation” defenses, for example, feature squeezing (Xu et al., 2017), frequency or JPEG compression (Dziugaite et al., 2016), randomized smoothing (Cohen et al., 2019), and other types of structured perturbations (Jafarnia-Jahromi et al., 2018; Zhang & Liang, 2019; Guo et al., 2017). Our first set of experiments dive into inference-time input perturbations, where the input to the network is manipulated prior to inference. The main trade-off is choosing the perturbation strength to carefully mitigate prediction errors over true examples but maximize recovery of the adversarial examples. Interestingly enough, we find that this trade-off is consistent across very different families of perturbations, where the relationship between channel distortion (effective perturbation of the input) and robustness is very similar. One might think that a frequency-based or JPEG based defense should be more tuned to a natural image classification setting—but we do not find that this is the case. The particular distribution of noise added to the inference process is not as important as its magnitude.

The unification of perturbation-based defenses gives us an insight into how an attacker might avoid them even if they did not know the particular defense. Our experiments suggest that all the perturbation based defenses are vulnerable to the same types of attack strategy where the attacker simply finds an adversarial example further away from the original image. This optimization procedure can be tuned to find the smallest distance from the original image that closes a “recovery window” (demonstrated in our experiments). In fact, we can further optimize this distance with a generic attacker that assumes a particularly strong perturbation, based on the additive Laplace noise. Adaptive attacks designed on this channel are often successful against other defenses. This result implies that input perturbation defenses are simply not effective and attackers can easily circumvent them without much knowledge about the particular defense.
More effective perturbation-based defenses are possible by adding perturbations beyond the input to the internal layers (Liu et al., 2018; Lecuyer et al., 2018), model parameters (He et al., 2019), as well as placed in a separately trained auto-encoder added before a network (Lecuyer et al., 2018). A crucial question is how much noise should be added and where? Our experiments compare the different placements of the noise layers across the network. In general, we find that the most effective current defenses leverage a strategy used in RobustNet (Liu et al., 2018), which adds noise throughout the network. On the other hand, we show that simply adding noise in a single layer, as in (Lecuyer et al., 2018), is actually ineffective in practice. We also find that training the network with this noise is crucial for high accuracy, and there is a threshold of perturbation after which a network must be trained with the noise to achieve any reasonable performance (approximately the standard deviation of a perturbed tensor).

We believe that these experiments are valuable to the community as they highlight important properties of perturbation-based defenses and the principles of constructing new strong defenses. We present analysis of input gradients and Hessians of adversarial examples that suggest they exhibit a higher level of instability than natural examples. More importantly, as a conclusion of this experimental study, we demonstrate a new robust model based on noise injection and adversarial training that achieves state-of-the-art robustness to adversarial defenses. This design is based on our empirical findings that layer perturbation is an important and effective defense, as in (Liu et al., 2018), and adversarial training (Madry et al., 2017) independently improves robustness (to the best of our knowledge the combination has not been proposed).

2. Background

2.1. Related Work

Much of the community’s current understanding of adversarial sensitivity in neural networks is based on the seminal work by (Szegedy et al., 2014). Multiple contemporaneous works also studied different aspects of this problem, postulating linearity and over-parametrization as possible explanations (Goodfellow et al., 2014; Biggio et al., 2013). Since the beginning of this line of work, the connection between compression and adversarial robustness has been recognized. The main defense strategies include: the idea of defensive network distillation (Papernot et al., 2015), quantizing inputs using feature squeezing (Xu et al., 2017), the thermometer encoding as another form of quantization (Buckman et al., 2018), JPEG compression harnessed by (Dziugaite et al., 2016; Guo et al., 2017; Das et al., 2017; 2018; Aydemir et al., 2018; Liu et al., 2019).

Other highly related line of research leverages randomization for adversarial robustness: Pixel Deflection (Prakash et al., 2018), random resizing and padding of the input image (Xie et al., 2017), total variance minimization (Guo et al., 2017), and injection of noise into inputs and each of the layers of neural networks (Liu et al., 2018). Dropout randomization (Feinman et al., 2017) was used to create a defense as well. Some of these approaches can further give proven guarantees of robustness (Zhang & Liang, 2019; Cohen et al., 2019; Lecuyer et al., 2018). More recent work focuses on combination of randomized smoothing with adversarial training and achieves state of the art in terms of the provable robustness (Salman et al., 2019).

Despite all of this research and several theoretical results, the empirical success of these approaches has not completely been established. Many aforementioned defenses were later broken (Carlini & Wagner, 2017; Athalye et al., 2018), and many recent defenses have not been compared against each other in a standard setting. To further complicate matters, the community lacks consensus on why such approaches afford any robustness in the first place: (Szegedy et al., 2014) suggest that neural networks have blind spots, (Xu et al., 2017) propose that quantization makes the adversarial search space smaller, (Buckman et al., 2018) suggest the linearity is the main culprit and quantized inputs break up linearity, or is the mechanism simply gradient obfuscation (Papernot et al., 2017).

This lack of understanding motivates our experimental study. In particular, we show five main results: (1) all input perturbation defenses, whether random or deterministic, are essentially equivalent in their efficacy, (2) such defenses offer almost no robustness to adaptive attacks unless these perturbations are observed during training, (3) a tuned sequence of noise layers across a network provides the best empirical robustness despite not having a certifiable guarantee, (4) attacks transfer between perturbation defenses so the attackers need not know the specific type of defense only that it involves perturbations, and (5) adversarial examples very close to original images show an elevated sensitivity to perturbation in a first-order analysis.

2.2. Notation and Metrics

We consider convolutional neural networks that take $w \times h$ (width times height) RGB digital images as input, giving an example space of $\mathcal{X} \in (255)^{w \times h \times 3}$, where $(z)$ denotes the integer numbers from 0 to $z$. We consider a discrete label space of $k$ classes represented as a confidence value $\mathcal{Y} \in [0, 1]^k$. Neural networks are parametrized functions (by a weight vector $\theta$) between the example and label spaces $f(x; \theta) : \mathcal{X} \mapsto \mathcal{Y}$. 

\[ y(x) = \underbrace{f(x; \theta) \cdot \mathbf{1}}_{\text{Eq. } (5)} \]

\[ y(x) = \underbrace{f(x; \theta) \cdot \mathbf{1}}_{\text{Eq. } (5)} \]
An adversarial input $x_{adv}$ is a perturbation of a correctly predicted example $x$ that is incorrectly predicted by $f$.

$$f(x) \neq f(x_{adv})$$

The distortion is the $\ell_2$ error between the original example and the adversarial one:

$$\delta_{adv} = \|x - x_{adv}\|_2$$

We focus on white-box attacks where the adversary has access to a parametric description of the model he/she is attacking.

White-box adversarial examples can be synthesized in three main settings. In the adaptive setting, the attacker not only knows the model but also knows exactly what defense is used. In the non-adaptive setting, the attacker does not know about the defense technique and assumes no defense is used. Finally, our experiments consider a novel partially adaptive setting, where the attacker knows that a perturbation-based defense is being used but does not know precisely which one.

### 2.3. A Unified View on Perturbation

While there is extensive work on using randomization or compression as a defense, we find that all of the approaches essentially follow the same format. They approximate a trained network $f(\cdot)$ with a less precise version $\hat{f}(\cdot)$, such that the adversarial example reverts back to the original class:

$$f(x) = \hat{f}(x_{adv})$$

Intuitively, a lossy version of $f$ introduces noise into a prediction which dominates the strategic perturbations found by an adversarial attack procedure. It turns out that we can characterize a number of popular defense methodologies with this basic framework.

How and where to inject noise is the core question. The most simple approach is input perturbation: first transforming $x$ through a noise process $C(x) = C[x' | x]$, and then evaluating the neural network:

$$y = f(x') \quad x' \sim C(x)$$

One could also perturb the network itself: add $\epsilon$ noise to the parameters of the trained network:

$$y = f(x; \hat{\theta}) \quad \hat{\theta} = \theta + \epsilon$$

Or, one could add noise to any or all of the intermediate layers of a network:

$$f(x) = g(h(x)) \quad y = \hat{f}(x) = g(h(x + \epsilon_1) + \epsilon_2)$$

### 2.4. Perturbation Analysis

Part of the goal of this experimental study is to understand the mechanism that allows such defenses to work in the first place. We start with the hypothesis that synthesized adversarial examples have unstable predictions—meaning that small perturbations to the input space can change confidence values drastically—and measure quantities that will allow us to quantify this instability.

Let $f(x)$ be a function that maps an image to a single class confidence value (i.e., a scalar output). We want to understand how $f(x)$ changes if $x$ is perturbed by $\epsilon$. We can apply a Taylor expansion of $f$ around the given example $x$:

$$f(x + \epsilon) \approx f(x) + \epsilon^T \nabla_x f(x) + \frac{1}{2} \epsilon^T \nabla^2_x f(x) \epsilon + ...$$

where $\nabla_x f(x)$ denotes the gradient of the function $f$ with respect to $x$ and $\nabla^2_x f(x)$ denotes the Hessian of the function $f$ with respect to $x$. The magnitude of the change in confidence is governed by the Taylor series terms in factorially decreasing importance. $\|\epsilon\|_2$ is distortion measure $\delta_e$. The expression is bounded in terms of the operator norm, or the maximal change in norm that could be induced, of each of the terms (see Supplement B):

$$\epsilon^T \nabla_x f(x) + \frac{1}{2} \epsilon^T \nabla^2_x f(x) \epsilon + ... \leq \delta_e M_1(x) + \frac{1}{2} \delta_e^2 M_2(x) + ...$$

As $\nabla_x f(x)$ is a vector, this is simply the familiar $\ell_2$ norm, and for the second order term this is the maximal eigenvalue:

$$M_1(x) = \|\nabla_x f(x)\|_2 \quad M_2(x) = \lambda_{\text{max}}(\nabla^2_x f(x))$$

When $M_1$ and $M_2$ are larger this means there is a greater propensity to change the prediction for small perturbations. We will show experimentally that for certain types of attacks the $M_1$ and $M_2$ values around adversarial examples exhibit signs of instability compared to those around natural examples—suggesting a mathematical mechanism of why recovery is possible.

### 3. Experiments

Our experiments evaluate the efficacy of perturbation based defenses in a number of different adversarial problem settings.

#### 3.1. Experimental Setup

We run our experiments using ResNet-18 or ResNet-20 on CIFAR-10 and ResNet-50 on ImageNet dataset using P-100 GPUs (16GB memory). We explore a number of different attacks that are implemented in the foolbox library (Rauber et al., 2017). In each experiment, we measure the test accuracy (%), the confidence of predictions, and distances between the original images and either their adversarial
We first study the simplest class of perturbation defenses, (feature squeezing), Uniform & Gaussian noise, SVD compression. From (He et al., 2019). More details on the setup can be presented in (Liu et al., 2018). We also use the PGD attack so that the gradients are not obfuscated. We approximate the gradients for the backward pass on the compression layers as an identity function, similarly to (He et al., 2017; Athalye et al., 2018). For the RobustNet (Liu et al., 2018) and PNI (He et al., 2019), the C&W attack is aware of the randomization procedure and we use its adaptive version presented in (Liu et al., 2018). We also use the PGD attack from (He et al., 2019). More details on the setup can be found in Supplement, Section A.

3.3. How to Attack Input Perturbation Defenses Non-adaptively?

Input perturbation defenses might seem like a simple, but effective trick to make a model adversarially robust, however, their similarity is actually a major pitfall. Even if the attacker does not know which particular defense is being used he/she can still attack the model with a “less-precise” attack—one that creates a larger but higher confidence adversarial image. The experiment in Section 3.2 shows that the defense philosophy is to generate a perturbation big enough to dominate the adversarial perturbations but small enough to generate valid predictions. This creates an obvious attack vector, where the attacker simply makes the adversarial perturbations large enough that the defender significantly hurts the accuracy of the model when trying to dominate the adversarially placed strategic perturbations. 

This interplay is visualized in Figure 2. We use an example from the ImageNet dataset, the ResNet-50 architecture, and set the stochastic channel to the Gaussian noise. We start from an adversarial example generated with the Carlini & Wagner (non-adaptive) $L_2$ attack and for consecutive subplots (in the top to bottom sequence), we increase the attack strength and incur higher distortion of the adversarial image.

An interesting way to compare very different classes of input perturbation defenses is to look at the “input distortion” ($L_2$ distance between the input and its perturbation, not to be confused with the distortion of the attack). We compare this metric against the test accuracy of a defended model, which indicates the ability of the perturbation to recover the original label.

For each image from CIFAR-10 test and ImageNet dev sets, we generate an adversarial attack with popular gradient-based methods. The attacks are tuned to minimize the distance between the adversarial input and the original example. We present the results in Figure 1. Each of the tested perturbation techniques is able to recover a substantial portion of correct labels from the adversarial examples. The curves are very similar in terms of where they achieve their peak robustness—the particular distribution of perturbations is not as relevant as their magnitude. The “optimally-tuned” defense essentially finds the same distortion parameter across very different stochastic and deterministic techniques. All these perturbation channels can be tuned to recover very similar maximum test accuracy of about 85% for CIFAR-10 and 70% for ImageNet after non-adaptive PGD and C&W attacks. This suggests that any form of perturbation with the right error magnitude is effective at defending against white-box non-adaptive attacks. If properly tuned, all such methods achieve a very similar level of robustness and their best parameter settings are essentially the same in terms of input distortion.

3.3. How to Attack Input Perturbation Defenses Non-adaptively?

Input perturbation defenses might seem like a simple, but effective trick to make a model adversarially robust, however, their similarity is actually a major pitfall. Even if the attacker does not know which particular defense is being used he/she can still attack the model with a “less-precise” attack—one that creates a larger but higher confidence adversarial image. The experiment in Section 3.2 shows that the defense philosophy is to generate a perturbation big enough to dominate the adversarial perturbations but small enough to generate valid predictions. This creates an obvious attack vector, where the attacker simply makes the adversarial perturbations large enough that the defender significantly hurts the accuracy of the model when trying to dominate the adversarially placed strategic perturbations.

This interplay is visualized in Figure 2. We use an example from the ImageNet dataset, the ResNet-50 architecture, and set the stochastic channel to the Gaussian noise. We start from an adversarial example generated with the Carlini & Wagner (non-adaptive) $L_2$ attack and for consecutive subplots (in the top to bottom sequence), we increase the attack strength and incur higher distortion of the adversarial image.
from the original image. For a single plot, we increase the $L_2$ distance of the output from the channel to the adversarial example by increasing the Gaussian noise (controlled by its standard deviation $\sigma$ and with mean $\mu = 0$). For $L_2$ distances incurred by different noise levels, we execute 100 predictions. We use the frequency count and report how many times the model predicts the original, adversarial, or other class.

The plot shows what range of distances from the adversarial image reveal the correct class. For the adversarial examples that are very close to the original image (e.g. adversarial distance of 0.006 for the top-left figure), the window of recovery (which indicates which strengths of the random noise can recover the correct label) is relatively wide, however, as we increase the distance of the adversarial image from the original image, the window shrinks and finally we are not enable to recover the correct label. Figure 16 in Supplement shows that these large distortion attacks are still imperceptible to the human eye. In conclusion, to avoid any sensitivity to input perturbation defenses an attacker must eliminate this recovery window by generating a larger distortion attack. In addition, statistical indications of adversarial examples should also be eschewed (Roth et al., 2019).

3.4. The Partially Adaptive Setting

The previous experiment shows that if the attacker makes the adversarial distortion large enough, perturbation-based defenses fail. Can an attacker still break such defenses with a low-distortion attack? One approach is to consider the adaptive setting. If the attacker has full knowledge of the defense, it is possible to construct an (adaptive) attack that is impervious to the defense. It has been known that many perturbation-based defenses are easily broken in the adaptive setting.

We go one step further. Since our experiments show that input transformations are so similar in their mechanisms, we find that an attacker does not need to be fully adaptive. To the best of our knowledge, such a problem setting has not been studied in the previous literature. The attacker simply assumes a particular strong perturbation-based defense and that same adversarial input often transfers to other defenses. Our attacker assumes that the defender is using a Laplace noise perturbation to defend the model, and generates an attack. We restrict the attacker to a single adaptive step (for details see Section D.5 in Supplement). Even in this weak adaptive setting, the deterministic channels are fully broken, and the randomized channels are mostly broken (with maximum accuracy of about 23.8%). We can strengthen this attack by giving the adversary a larger “compute budget” where the adversary is given an unlimited number of adaptive steps (Figure 11) thereby driving the accuracy to 0.

Laplace attacked images transfer the best to other defenses. They decrease the accuracy of the defense models by at least 44.3% (for the Laplace-based defense itself). Table 1 shows that FC attacked images do not transfer well to other defenses; the maximum drop in accuracy of the model protected by other defenses is 12.26%. Most adversarial images (against a given defense) transfer very well to the FC defense, i.e. an adversarial image against any defense (e.g. CD, SVD, Gauss, Uniform, or Laplace) is also adversarial against the FC defense. The adversarial images generated against the Uniform defense show better transfer to other defenses in comparison to the adversarial images generated against the Gaussian defense. This is because the higher noise level is applied in the Uniform defense. We observe analogous trends for the ImageNet dataset. In short, the input perturbation is not a sufficient defense for adversarial attacks even if the attacker does not know the particular details of the defense. The attacker can either synthesize higher distortion attacks or assume a Laplace-distributed perturbation.
Table 1. Transferability of the adversarial images created against a given noisy channel denoted as $A$ (adaptive attack specified in the first column) to the defense protected with a noisy channel denoted as $D$ (the defense with a noisy channel specified in the first row). Each result represents a recovery (%) of the adversarial examples (generated for $A$) to correct labels after applying the defense ($D$). We use 30% FC compression, 50% SVD compression, 4 bit values in CD, 0.03 noise level for Gauss and Laplace, and 0.04 noise level for the Uniform channel. We use ResNet-18 model, 2000 images from the CIFAR-10 test set and 100 attack iterations with 5 binary steps to find the $c$ value (with initial $c$ value set to 0.01) for the adaptive C&W $L_2$ attack. The baseline test accuracy is 93.56%. The test accuracy of the noisy channels on clean images is given in the first row denoted: Empty (an empty attack).

|        | FC  | CD  | SVD | Gauss | Laplace |
|--------|-----|-----|-----|-------|---------|
| Empty  | 93.32 | 93.01 | 93.12 | 92.53 | 91.35   |
| FC     | **0.20** | 80.75 | 83.05 | 81.15 | 78.70   |
| CD     | 3.85 | **0.70** | 43.60 | 47.30 | 62.35   |
| SVD    | 1.99 | 47.96 | **0.77** | 46.52 | 65.75   |
| Gauss  | 4.45 | 48.70 | 44.80 | 51.50 | 60.15   |
| Uniform| 3.45 | 30.30 | 30.60 | 30.15 | 51.55   |
| Laplace| 3.05 | 23.35 | 24.60 | **23.80** | **46.70** |

Figure 3. Adaptive vs non-adaptive attack. We use adaptive white-box C&W $L_2$ attack against: PlainNet, RobustNet (init noise 0.2, inner noise 0.1), and FC (frequency-based channel with 50% compression rate). On the left, we generate the attack on PlainNet and run against RobustNet and FC defenses. On the right, we generate the attack with knowledge about the defenses. We use VGG16 trained on CIFAR-10 data.

3.5. Robustness to Adaptive Attacks

The previous set of experiments show that simply adding post-hoc perturbations to a model is not a very effective defense. A crucial way to boost the robustness of perturbation approaches is to train the model observing training data perturbed by the future defense. This approach seems similar to adversarial training, but turns out to be much more computationally efficient as the perturbations are usually easy to calculate. While it has been previously noted that adversarial examples can fool multiple models (Tramèr et al., 2017), we find that perturbation defenses actually significantly affect transferability.

We evaluate, one such model based on the VGG-16 architecture, RobustNet model (Liu et al., 2018), in the same framework as before. RobustNet adds noise layers after each of the convolutional layers in the network. Figure 3 illustrates the results (and for reference we include FC on VGG-16, which is one of the better performing perturbations in this setting). We see that RobustNet is nearly perfect in the non-adaptive setting, i.e., an attack on a standard trained network does not transfer at all to RobustNet. This perturbation forces the adaptive attack to generate much more distorted examples in the adaptive setting as well. For example, for the same CW parameters ($c = 1.0$), the $L_2$ distortion is 0.87 for RobustNet, and only 0.27 for PlainNet (just a vanilla VGG-16 network). RobustNet performs much better than PlainNet or PlainNet with FFT layer (added before the network during inference). For the non-adaptive attack, where we generate adversarial examples on the PlainNet, the examples found have maximum $L_2$ distortions up to 0.47, so we cannot degrade the performance of the FFT-based and RobustNet defenses to the test accuracy of 0.0. Thus, attacks that could potentially succeed against most defenses have to incur much higher distortions.

3.6. Train vs Test Time Noise Addition

Neural networks can be perturbed in different ways. RobustNet adds noise layers, but an alternative is to perturb the parameters themselves, which a simplified version of (He et al., 2019) but without adversarial training. We compare robustness between networks with either parameter or feature perturbations. The pure parameter perturbation (ParamNet) is less robust than the network with feature perturbation (RobustNet). In Figure 5, for adversarial $L_2$ distortion of 0.5, the accuracy of ParamNet is 0.0, while RobustNet has accuracy above 45% (when comparing the two methods with similar accuracy on clean data). The addition of noise to features is more effective than to parameters because the number of parameters is an order of magnitude lower than number of elements in the input and intermediate data in current deep neural networks (Rhu et al., 2016). Moreover, the standard deviations of tensors for features are an order of magnitude higher than for parameters. Thus, less effective amount of variation can be added to the network via parameters.

The experiments with ParamNet illustrate another interesting phenomenon. As we increase the scale of the perturbation noise, there is a threshold after which training is necessary to achieve any reasonable accuracy. The parameters are initialized with Gaussian noise with mean 0 and standard deviation $\sigma = 0.02$. We observe that this $\sigma$ of the initialization noise persists through training and establishes a threshold for the perturbation noise. The perturbation noise with $\sigma \geq 0.02$ added during inference turns the network into a random classifier. Thus, the training is essential to recover the model test accuracy on clean data. For the
perturbation noise with $\sigma < 0.2$, there is a negligible difference between the test accuracy with or without training. For $\sigma > 0.2$, the network trained with noise is more robust but at the cost of lower accuracy on clean data.

Figure 5 demonstrates that where and how we add noise matters within RobustNet as well. In the RobustNet figure, we list different scales of mean 0 Gaussian noise. The first number is the noise introduced to the first layer, and the second number is noise introduced to all subsequent layers. Adding noise to the first layer is essentially the same as input perturbation. We have to add noise to all of the intermediate layers to achieve a high level of robustness. This result is significant because some recent defenses argue that noise injection early in a network is sufficient, our results do not suggest that is the case (Lecuyer et al., 2018).

3.7. Noise Injection and Adversarial Training

Adversarial training (Adv. Training) (Madry et al., 2017) is one of the most successful practical defenses. The RobustNet defense injects noise into input and intermediate feature maps, which gives similar gains in robustness to Adv. Training according to (Liu et al., 2018). PNI-W (He et al., 2019) combines noise injection into network parameters with Adv. Training and can give higher robustness than pure Adv. Training. Previous Section 3.6 shown that injecting noise into network layers performs better than noise injection solely into parameters. Thus, we propose to combine RobustNet with Adv. Training (denoted as RobustNet Adv.) and compare it to Adv. Training, PNI-W Adv., and RobustNet defense methods.

Current state of the art attacks are white-box and adaptive. For all compared defenses, adversarial training is done using PGD with 7 iterations (also aliased steps), similarly to (He et al., 2019). We use the same parameters as in the previous work by (Madry et al., 2017). The perturbation scale of 76.5/255 (0.3), and step size 2.55/255 (0.01). We only vary either number of iterations for the attack or perturbation scale $\epsilon$ (which influences the $L_\infty$ distortion of adversarial examples). Besides using 40 and 100 total attack iterations, we also increase the iterations to 1000 to further strengthen the adversary, similarly to (Yang et al., 2019). RobustNet employs the randomized transformations so we use modified version of C&W attack with EOT (Expectation over Transformation) (Athalye et al., 2018) as implemented in (Liu et al., 2018). We run the algorithm with 200 internal iterations for the gradient descent. The $\epsilon$ parameter is explored in broad range of values from 0 to 100. For RobustNet, we adjust the standard deviation $\sigma$ of the injected noise to 0.2 in the input layer and 0.1 in all the remaining internal layers. For RobustNet Adv., we keep the same $\sigma = 0.1$ for all the noisy layers. We train PNI-W in the same setup as in (He et al., 2019), with the equally weighted (by 0.5) sum of losses for clean and perturbed-data. To the best of our knowledge, the combination of RobustNet and Adv. Training has not been proposed before in the literature.

We represent the results in Fig. 4 as the test accuracy being a function of either strength of an attack or distortion incurred by an adversary. The former approach allows us to adjust attack parameters. The latter depiction gives us more absolute comparison between the defenses. We can observe that for the same $c$ parameter in CW, the attack generates different adversarial examples depending on the attacked network. A more robust network incurs higher distortion required to be used by adversary. For example, for parameter $c = 0.4$, the distortion required to attack RobustNet is 0.63, whereas to attack Adv. Trained network, the required distortion is 0.57.

For PGD, RobustNet performs worse than Adv. Training and PNI-W Adv. For CW, RobustNet gives higher accuracy only for highly distorted examples. However, the combination of RobustNet and Adv. Training provides improvement across the whole spectrum. PNI-W Adv. is on par with Adv. Training for CW for most distortion levels and slightly outperforms it for highly distorted examples, similarly to RobustNet. Clearly, combining Adv. Training with any form of noise injection is beneficial. Nevertheless, the type and placement of the noise injections are important and RobustNet Adv. outperforms PNI-W Adv. in most cases.

PNI-W defense injects noise into weights with trained scale factor and the initial noise magnitude based on the standard deviation of the weights. RobustNet is tuned manually and the standard deviation of each layer is a guiding value for the injected noise level. We find that training RobustNet in similar manner as PNI-W (so that the noise scale is a parameter) lowers the scale too much and the injected noise is too small. For average $\sigma = 0.25$ for the noise layers, the trained scale factor lowers the noise by two orders of magnitude, which makes network much less robust. This suggests that we have to transfer manual tuning from the noise level to adjusting the weights in ensemble loss. It gives fewer knobs but still requires human intervention. Another approach would be to train the noise layers in a similar way to Batch Normalization layers with the noise scale factor frozen for inference. We leave this as future work.

3.8. Adversarial Examples Are Unstable

The key question is why adversarial examples are more sensitive to perturbations than natural inputs, when there is evidence that from an input perspective they are statistically indistinguishable. Our experiments suggest that this sensitivity arises from the optimization process that generates adversarial examples.

Based on the operator-norm analysis presented in Section 2.4, we measure $L_2$ norm of the input gradients w.r.t.
An Empirical Evaluation of Perturbation-based Defenses

![Figure 4](image_url)

Figure 4. White-box adaptive attacks C&W and PGD used to generate adversarial examples and tested against the following defenses: adversarial training (Adv. Train), parameter noise injection to the weights that uses adversarial training (PNI-W Adv.), feature noise injection (RobustNet), and RobustNet combined with adversarial training (RobustNet Adv.). We train ResNet-20 on CIFAR-10 dataset.

![Figure 5](image_url)

Figure 5. We use the adaptive white-box C&W L2 attack and VGG16 trained on CIFAR-10. **Left: Training vs Inference.** The robustness of ParamNet with noise added either during training and inference or only during inference (test). **Right: Placement of noise layers.** The test accuracy of RobustNet with noise layers placed before every convolution layer is much higher than when the noise layer is placed only before the first convolution layer.

The non-adaptive attacks are not robust since small changes to an adversarial input often recovers its correct label. This is an obvious corollary to the very existence of adversarial examples that by definition are relatively close to correctly predicted examples in the input space. Random perturbations of the input can dominate the strategically placed perturbations synthesized by an attack. In fact, the results are consistent across both deterministic and stochastic channels that degrade the fidelity of the input example. From the perspective of the attacker, the recovery window can be closed to make the perturbation based recovery techniques ineffective. Moreover, the Laplace noise can be assumed a priori to achieve high transferability of the attacks. The most effective defenses perturb not only inputs but also the internal layers. The scale of the noise can be set as a parameter. We find that the combination of layer perturbations and adversarial training improves robustness against adaptive CW and PGD attacks in comparison to pure adversarial training or parameter noise injection with adversarial training. The training of networks with strong perturbation defenses (high level of added noise) is essential to recover the test accuracy on clean data. Our first and second order analysis shows that the perturbation based defenses are possible since the adversarial examples are not robust themselves.

4. Conclusion

The non-adaptive attacks are not robust since small changes to an adversarial input often recovers its correct label. This is an obvious corollary to the very existence of adversarial examples that by definition are relatively close to correctly predicted examples in the input space. Random perturbations of the input can dominate the strategically placed perturbations synthesized by an attack. In fact, the results are consistent across both deterministic and stochastic channels that degrade the fidelity of the input example. From the perspective of the attacker, the recovery window can be closed to make the perturbation based recovery techniques ineffective. Moreover, the Laplace noise can be assumed a priori to achieve high transferability of the attacks. The most effective defenses perturb not only inputs but also the internal layers. The scale of the noise can be set as a parameter. We find that the combination of layer perturbations and adversarial training improves robustness against adaptive CW and PGD attacks in comparison to pure adversarial training or parameter noise injection with adversarial training. The training of networks with strong perturbation defenses (high level of added noise) is essential to recover the test accuracy on clean data. Our first and second order analysis shows that the perturbation based defenses are possible since the adversarial examples are not robust themselves.
References

Athalye, A., Carlini, N., and Wagner, D. A. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In ICML, 2018.

Aydemir, A. E., Temizel, A., and Taskaya-Temizel, T. The effects of JPEG and JPEG2000 compression on attacks using adversarial examples. CoRR, abs/1803.10418, 2018. URL http://arxiv.org/abs/1803.10418.

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., and Roli, F. Evasion attacks against machine learning at test time. In Joint European conference on machine learning and knowledge discovery in databases, pp. 387–402. Springer, 2013.

Buckman, J., Roy, A., Raffel, C., and Goodfellow, I. Thermometer encoding: One hot way to resist adversarial examples. In International Conference on Learning Representations, 2018.

Carlini, N. and Wagner, D. Towards evaluating the robustness of neural networks. In 2017 IEEE Symposium on Security and Privacy (SP), pp. 39–57, May 2017.

Carlini, N. and Wagner, D. Adversarial examples are not easily detected: Bypassing ten detection methods. In Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, pp. 3–14. ACM, 2017.

Cohen, J. M., Rosenfeld, E., and Kolter, J. Z. Certified adversarial robustness via randomized smoothing. arXiv preprint arXiv:1902.02918, 2019.

Das, N., Shanbhogue, M., Chen, S., Hohman, F., Chen, L., Kounavis, M. E., and Chau, D. H. Keeping the bad guys out: Protecting and vaccinating deep learning with JPEG compression. CoRR, abs/1705.02900, 2017. URL http://arxiv.org/abs/1705.02900.

Das, N., Shanbhogue, M., Chen, S., Hohman, F., Li, S., Chen, L., Kounavis, M. E., and Chau, D. H. Shield: Fast, practical defense and vaccination for deep learning using JPEG compression. CoRR, abs/1802.06816, 2018. URL http://arxiv.org/abs/1802.06816.

Dziugaite, G. K., Ghahramani, Z., and Roy, D. M. A study of the effect of jpg compression on adversarial images. arXiv preprint arXiv:1608.00853, 2016.

Feinman, R., Curtin, R. R., Shintre, S., and Gardner, A. B. Detecting adversarial samples from artifacts. arXiv preprint arXiv:1703.00410, 2017.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572, 2014.

Guo, C., Rana, M., Cisse, M., and van der Maaten, L. Countering Adversarial Images using Input Transformations. arXiv e-prints, art. arXiv:1711.00117, Oct 2017.

He, W., Wei, J., Chen, X., Carlini, N., and Song, D. Adversarial example defense: Ensembles of weak defenses are not strong. In 11th USENIX Workshop on Offensive Technologies (WOOT 17), 2017.

He, Z., Rakin, A. S., and Fan, D. Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019.

Jafarnia-Jahromi, M., Chowdhury, T., Wu, H.-T., and Mukherjee, S. Ppd: Permutation phase defense against adversarial examples in deep learning. arXiv preprint arXiv:1812.10049, 2018.

Lecuyer, M., Atlidakis, V., Geambasu, R., Hsu, D., and Jana, S. Certified Robustness to Adversarial Examples with Differential Privacy. arXiv e-prints, art. arXiv:1802.03471, Feb 2018.

Liu, X., Cheng, M., Zhang, H., and Hsieh, C.-J. Towards robust neural networks via random self-ensemble. In Ferrari, V., Hebert, M., Sminchisescu, C., and Weiss, Y. (eds.), Computer Vision – ECCV 2018, pp. 381–397, Cham, 2018. Springer International Publishing. ISBN 978-3-030-01234-2.

Liu, Z., Liu, Q., Liu, T., Xu, N., Lin, X., Wang, Y., and Wen, W. Feature distillation: Dnn-oriented jpeg compression against adversarial examples. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083, 2017.

Papernot, N., McDaniel, P., Wu, X., Jha, S., and Swami, A. Distillation as a defense to adversarial perturbations against deep neural networks. arXiv preprint arXiv:1511.04508, 2015.

Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., and Swami, A. Practical black-box attacks against machine learning. In Proceedings of the 2017 ACM on Asia conference on computer and communications security, pp. 506–519. ACM, 2017.

Prakash, A., Moran, N., Garber, S., DiLillo, A., and Storer, J. Deflecting adversarial attacks with pixel deflection. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018.
An Empirical Evaluation of Perturbation-based Defenses

Rauber, J., Brendel, W., and Bethge, M. Foolbox: A python toolbox to benchmark the robustness of machine learning models. *arXiv preprint arXiv:1707.04131*, 2017.

Rhu, M., Gimelshein, N., Clemons, J., Zulfiqar, A., and Keckler, S. vdn: Virtualized deep neural networks for scalable, memory-efficient neural network design. pp. 1–13, 10 2016. doi: 10.1109/MICRO.2016.7783721.

Roth, K., Kilcher, Y., and Hofmann, T. The odds are odd: A statistical test for detecting adversarial examples. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5498–5507, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/roth19a.html.

Salman, H., Yang, G., Li, J., Zhang, P., Zhang, H., Razenshteyn, I. P., and Bubeck, S. Provably robust deep learning via adversarially trained smoothed classifiers. *CoRR*, abs/1906.04584, 2019. URL http://arxiv.org/abs/1906.04584.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., and Fergus, R. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2014.

Tabacof, P. and Valle, E. Exploring the space of adversarial images. *CoRR*, abs/1510.05328, 2015. URL http://arxiv.org/abs/1510.05328.

Tramèr, F., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. The space of transferable adversarial examples. *arXiv preprint arXiv:1704.03453*, 2017.

Xie, C., Wang, J., Zhang, Z., Ren, Z., and Yuille, A. Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991*, 2017.

Xu, W., Evans, D., and Qi, Y. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.

Yang, Y., Zhang, G., Katabi, D., and Xu, Z. ME-net: Towards effective adversarial robustness with matrix estimation. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7025–7034, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/yang19e.html.

Yao, Z., Gholami, A., Lei, Q., Keutzer, K., and Mahoney, M. W. Hessian-based analysis of large batch training and robustness to adversaries. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 4949–4959. Curran Associates, Inc., 2018.

Zhang, Y. and Liang, P. Defending against whitebox adversarial attacks via randomized discretization. *AISTATS*, 2019.
A. Details on the experimental setup

We use the foolbox library (Rauber et al., 2017) in our experiments. We borrowed the nomenclature used for the attacks from the library. For example, the name for the attack initially proposed in (Szegedy et al., 2014) and extended in (Tabacof & Valle, 2015) is LBFGS. In most of our experiments, we also use the default foolbox parameters for the attacks. For example, for PGD the initial limit on the perturbation size epsilon is set to 0.3, step size to 0.01, default number of iterations is 40. For Carlini & Wagner, we set maximum number of iterations to 1000, learning rate to 0.005, initial value of the constant c to 0.01. Note that for the Carlini & Wagner, we use the c parameter as described in (Carlini & Wagner, 2017) and also the code from (Liu et al., 2018). For the LBFGS attack, we use the epsilon parameter set to 0.00001 and up to 150 iterations. For the FGSM attack, different epsilons starting from 50 and up to 1000 are tried until the adversarial image is found. For the BIM L1 attack, we set epsilon to 0.3, step size to 0.05, and number of iterations to 10.

B. Perturbation analysis: addendum

\[ e^T \nabla_x f(x) + \frac{1}{2} e^T \nabla^2_x f(x) e + \cdots \leq \delta_c M_1(x) + \frac{1}{2} \delta_c^2 M_2(x) + \cdots \]

\[ M_1(x) = \| \nabla_x f(x) \|_2 \quad M_2(x) = \lambda_{\max}(\nabla^2_x f(x)) \]

(1) \[ e^T \nabla_x f(x) \leq \delta_c M_1(x) \]

From the Cauchy-Schwarz inequality:

\[ \| e \|_2 M_1(x) = \delta_{adv} M_1(x) \leq \delta_c M_1(x) \quad (\text{since } \delta_{adv} \ll \delta_c) \]

(2) \[ \nabla^2_x f(x) e \leq \delta_c^2 M_2(x) \]

From the definition of maximum eigenvalue:

\[ \lambda_{\max} \geq \frac{e^T \nabla^2_x f(x) e}{e^T e} \]

\[ e^T \nabla^2_x f(x) e \leq \| e \|_2^2 \lambda_{\max} = \delta_{adv} \lambda_{\max} \leq \delta_c^2 \lambda_{\max} \]

C. Compression techniques

C.1. FFT-based compression

We apply compression in the frequency domain to reduce the precision of the input images. Let x be an input image, which has corresponding Fourier representation that re-indexes each tensor in the frequency domain:

\[ F[\omega] = F(x[n]) \]

This Fourier representation can be efficiently computed with an FFT. The mapping is invertible \( x = F^{-1}(F(x)) \). Let \( M_f[\omega] \) be a discrete indicator function defined as follows:

\[ M_f[\omega] = \begin{cases} 1, & \omega \leq f \\ 0, & \omega > f \end{cases} \]

\( M_f[\omega] \) is a mask that limits the \( F[\omega] \) to a certain band of frequencies. \( f \) represents how much of the frequency domain is considered. The band-limited spectrum is defined as, \( F[\omega] \cdot M_f[\omega] \), and the band-limited filter is defined as:

\[ x' = F^{-1}(F[\omega] \cdot M_f[\omega]) \]

C.2. SVD-based compression

Analogously to the FFT-based method, we decompose an image with SVD transformation and reconstruct its compressed version with dominant singular values. The basis used in SVD are adaptive and determined by an image, as opposed to pre-selected basis used in FFT. This can result in higher quality for the same compression rate in case of SVD, however it is more computationally intensive than FFT-based compression.

D. Additional experiments for White-box Attacks

D.1. Accuracy of Perturbation Defenses on Clean Data

One pitfall of the input perturbation defense is that it introduces errors whether or not there are any adversarial examples. The errors act as an upper-bound for the best possible test accuracy we can get under adversarial perturbations. For frequency compression, color depth compression, and uniform noise injection, we compare the test accuracy for different levels of imprecision.

The test accuracy of the models can be increased by training with compression, e.g., by using FFT based convolutions with 50% compression in the frequency domain increases the accuracy to 92.32%. We present the results for six different noisy channels; three of them are compression based: FC, CD, SVD, and other three add different type of noise: Gauss, Uniform, and Laplace. For each of the compression based approach, we increase the compression rate systematically from 0 to about 90% (in case of the CD, the compression rate is computed based on how many bits are used per value). For the noise based approaches, we increase the strength of the noise by controlling the epsilon parameter \( \epsilon \) (in case of the Gaussian noise, it corresponds to the sigma parameter \( \sigma \)). The full result is presented in Figure 7.
D.2. Distributions of input perturbations

We emphasize in the paper that the distribution of noise does not matter as much as the magnitude. To illustrate this point, we plot the distribution of $deltas$ for six imprecise channels in Figure 8. We compute the $deltas$ by subtracting an original image from the perturbed adversarial image and plot the histograms of differences. We use an image from the ImageNet dataset. For all the examples, the correct labels were recovered. We use the C&W attack with 1000 iterations and the initial value $c = 0.01$. All of the distributions are varied, yet they achieve a similar robustness in the non-adaptive setting.

D.3. More on Recovery Windows

In Figure 9, we expand the experiments described in Section 3.3. We run the experiment for more sets of values and plot the class frequencies post-perturbation for different $c$ parameters (how close we are to the original image). As we increase $c$, our ability to tune a perturbation-based defense goes down, namely, there are no range of distortions for which we are more likely to be in the original class.

D.4. Multiple Trials For Stochastic Perturbations

Another compelling reason to use a stochastic perturbations (like Uniform noise) as a defense is that it can be run repeatedly in a number of random trials. We show that doing so slightly improves the efficacy of the defense. We randomly choose 1000 images from the CIFAR-10 test set. For each of these images, we generate an adversarial attack. We then pass each image through the same stochastic perturbation multiple times. We take the most frequent prediction. Figure 10 illustrates the results.

Only 16 trials are needed to get a relatively strong defense. We argue that this result is significant. Randomized defenses are difficult to attack. The attacker cannot anticipate which particular perturbation to the model will happen. The downside is a potential of erratic predictions. We show that a relatively small number of trials can greatly reduce this noise. Furthermore, the expense of running multiple trials of a randomized defense is small relative to the expense of synthesizing an attack in the first place.

D.5. More Details on The White-Box Adaptive Attack

The stochastic perturbations are harder to attack with a gradient-based adaptive methods. Our strategy is to send an output from the adversarial algorithm through the imprecise network and the network at least as many times as set in the defense. We mark the attack as successful if the most frequent output label is different from the ground truth. The more passes through the noisy channel we optimize for, the stronger the attack. Furthermore, we run many iterations of the attack to decrease the $L_2$ distortion. An attack that always evades the noise injection defense is more difficult to generate because of randomization. The other randomized approach was introduced in dropout (Feinman et al., 2017). The attacks against randomized defenses require optimization of complex loss functions, incur higher distortion, and the attacks are not fully successful (Carlini & Wagner, 2017). In the Figure 11, we present the result of running attacks and defenses on CIFAR-10 data with single and many iterations. The defense with many trials can be drawn to 0% accuracy, however, the defense not fully optimized by the
adversary (single noise injection) can result in about 40% or higher accuracy.

**D.6. Gradient-based Analysis**

We run the experiment on the ImageNet dataset. We analyze only the clean images that were classified correctly. We present how the gradient of the loss w.r.t. the input image changes for the correct class as we add the Gaussian noise to the original image in Figure 12. The norm of the gradient smoothly increases. In Figure 13, we start from an adversarial image found with the default C&W attack from the foolbox library. Then, we systematically add Gaussian noise to the adversarial image and collect data on the norm of gradients for the original and adversarial classes. The norm of the gradients for the adversarial class increases while the norm of the gradients for the original class decreases. We cross the decision boundary to the correct class very early and recover the correct labels for images. Then, as we add more and more Gaussian noise, the predictions of the classifier become random and the norm of the gradients converge to a single value. In Figure 14, we plot the gradients also for a random class. We observe that for an untargeted attack, the gradients for the original and adversarial classes are larger than for the other classes. The targeted attack decreases the loss for the target class and the gradients for the adversarial classes are lower when compared with gradients from the untargeted attacks, so fewer images can be recovered in the former case. The targeted attack causes a smaller increase of the norms of gradients for the original class than the untargeted attack. However, it is still higher than for a random class.

**D.6.1. Controlling the Gradient Magnitude**

The gradient magnitude can be controlled with the confidence parameter in the C&W attack. There is a high dependence between what confidence level is set for the attack and what the gradient norm is. If we set the confidence to 0 (zero), then we only intend to mis-classify an example, and...
Figure 11. For the CIFAR-10 dataset, we run multiple trials of the uniform noise perturbation and take the most frequent prediction in the defense (many noise iterations). We also run just a single noise injection and return the predicted label. The attacker runs the same number of many uniform trials as the defender. The experiment is run on 100 images, with 100 C&W $L_2$ attack iterations.

Figure 12. The changes in the $L_2$ norm of the gradient of the loss w.r.t. the input image $x$ for the correct class $c_{org}$ as we add Gaussian noise to the original image. The experiment is run on 1000 images from the ImageNet dataset.

Figure 13. The changes in the $L_2$ norm of the gradient for the correct class $c_{org}$ and the adversarial class $c_{adv}$, as we add Gaussian noise to the adversarial image generated with C&W $L_2$ attack. The experiment is run on 1000 images from the CIFAR-10 dataset.

Figure 14. The changes in the $L_2$ norm of the gradient of the loss for the correct class $c_{org}$, the adversarial class $c_{adv}$, and a random class $c_{ran}$ as we add Gaussian noise to the adversarial image generated with C&W $L_2$ attack. The experiment is run on 1000 images from the CIFAR-10 dataset.

The softmax probabilities and norms of gradients are correlated for original images but not necessarily for adversarial examples. We run experiment for 1000 images from the CIFAR-10 dataset, the classification accuracy is 93.9%. We take into account only the correctly classified images. For each image we generate an adversarial example using default C&W attack from the foolbox library (where confidence parameter is zeroed out). We record the softmax probabilities and norms of the gradients for each of the 10 classes. For 99% of the original images, the lowest gradients are for the original class. Only for 1% of the adversarial examples, the lowest gradients are for the adversarial class.

D.7. Visualizations of attacks and imprecise channels

Figure 16 presents a sample image from ImageNet for the Carlini-Wagner $L_2$ attack.
An Empirical Evaluation of Perturbation-based Defenses

Figure 15. Dependence between confidence levels of the CW attack and gradients of the generated adversarial examples.

E. Additional Experiments for black-box attacks

As a black box attack, we define an attack that does not need the knowledge about the gradient or the model.

E.1. Decision-based attacks

The attacks require neither gradients nor probabilities. They operate directly on the images.

E.1.1. Robustness to Uniform and Gaussian Noise

We evaluate the robustness of band-limited CNNs. Specifically, models trained with more compression discard part of the noise by removing the high frequency Fourier coefficients (FC channel). In Figure 17, we show the test accuracy for input images perturbed with different levels of uniform and Gaussian noise, which is controlled systematically by the sigma parameter, fed into models trained with different compression levels (i.e., 0%, 50%, or 85%) and methods (i.e., band-limited vs. RPA-based).

Our results demonstrate that models trained with higher compression are more robust to the inserted noise. Interestingly, band-limited CNNs also outperform the RPA-based method and under-fitted models (e.g., via early stopping), which do not exhibit the robustness to noise.

Input test images are perturbed with uniform or Gaussian noise, where the sigma parameter is changed from 0 to 1 or 0 to 2, respectively. The more band-limited model, the more robust it is to the introduced noise.

E.1.2. Contrast Reduction Attack

This black-box attack gradually distorts all the pixels:

$$ target = \frac{\max + \min}{2} $$

$$ perturbed = (1 - \epsilon) \times \text{image} + \epsilon \times target $$

where min and max values are computed across all pixels of images in the dataset.

We can defend the attack with CD (Color Depth reduction) until certain value of epsilon, but then every pixel is perturbed in a smooth way so there are no high-frequency coefficients increased in the FFT domain of the image. The contrast reduction attack becomes a low-frequency based attack when considered in the frequency domain. Another way to defend the attack is to run a high-pass filter in the frequency domain instead of the low-pass filter.

We run the experiments for different models with CD and two band-limited models (the model with full spectra and no compression as well as model with 85% of compression - with FC layers). The CD does defend the attack to some extent and the fewer pixels per channel (the stronger the CD in a model), the more robust the model is against the contrast reduction attack.

Test accuracy as a function of the contrast reduction attack for ResNet-18 on CIFAR-10 (after 350 epochs) is plotted in Figure 17. We control the strength of the attack with parameter epsilon that is changed systematically from 0.0 to 1.0. We use the whole test set for CIFAR-10. R denotes the number of values used per channel (e.g., R=32 means that we use 32 values instead of standard 256).

The Reduced Precision Arithmetic, where operations on 16 bit floats are used instead of on 32 or 64 bit float numbers.
E.1.3. **Multiple pixels attack**

The foolbox library supports a single pixel attack, where a given pixel is set to white or black. A certain number of pixels (e.g., 1000) is chosen and each of them is checked separately if it can lead to the misclassification of the image. The natural extension is to increase the number of pixels to be perturbed, in case where the single pixel attack does not succeed. We present results for the multiple pixel attack in Figure 17.

E.2. **Spatial-based attacks**

Spatial attacks apply adversarial rotations and translations that can be easily added to the data augmentation during training. However, these attacks are defended neither by removing the high frequency coefficients nor by quantization (CD). We separately apply rotation by changing its angle from 0 to 20 degrees and do the translations within a horizontal and vertical limit of shifted pixels (Figure 17).

E.3. **Score-based attacks**

The score based attack require access to the model predictions and its probabilities (the inputs to the softmax) or the logits to estimate the gradients.

E.3.1. **Local Search Attack**

The local search attack estimates the sensitivity of individual pixels by applying extreme perturbations and observing the effect on the probability of the correct class. Next, it perturbs the pixels to which the model is most sensitive. The procedure is repeated until the image is misclassified, searching for additional critical pixels in the neighborhood of previously found ones. We run the experiments for the attack on 100 test images from CIFAR-10, since the attack is relatively slow (Figure 17).
An Empirical Evaluation of Perturbation-based Defenses

Figure 16. We plot a sample image from the ImageNet dataset in its original state, after adversarial (white-box, non-adaptive C&W $L_2$) attack, and then after recovery via imprecise channels: CD (color depth reduction with 32 bits), FC (30% compression in the frequency domain), Gaussian, and uniform noise ($\epsilon = 0.03$).

Figure 17. Test accuracy as a function of the strengths of the attacks for ResNet-18 on CIFAR-10.