Adversarial Neural Pruning with Latent Vulnerability Suppression

Divyam Madaan 1  Jinwoo Shin 2 3  Sung Ju Hwang 1 3 4

Abstract
Despite the remarkable performance of deep neural networks on various computer vision tasks, they are known to be susceptible to adversarial perturbations, which makes it challenging to deploy them in real-world safety-critical applications. In this paper, we conjecture that the leading cause of adversarial vulnerability is the distortion in the latent feature space, and provide methods to suppress them effectively. Explicitly, we define vulnerability for each latent feature and then propose a new loss for adversarial learning, Vulnerability Suppression (VS) loss, that aims to minimize the feature-level vulnerability during training. We further propose a Bayesian framework to prune features with high vulnerability to reduce both vulnerability and loss on adversarial samples. We validate our Adversarial Neural Pruning with Vulnerability Suppression (ANP-VS) method on multiple benchmark datasets, on which it not only obtains state-of-the-art adversarial robustness but also improves the performance on clean examples, using only a fraction of the parameters used by the full network. Further qualitative analysis suggests that the improvements come from the suppression of feature-level vulnerability.

1. Introduction
In the last many years, deep neural networks (DNNs) have achieved impressive results on various computer vision tasks, e.g., image classification (Krizhevsky et al., 2012), face/object recognition (He et al., 2015; Deng et al., 2019), and semantic segmentation (He et al., 2017). The groundbreaking success of DNNs has motivated their use in safety-critical environments such as medical imaging (Esteva et al., 2017) and autonomous driving (Bojarski et al., 2016). However, DNNs are extremely brittle to carefully crafted imperceptible adversarial perturbations intentionally optimized to cause miss-prediction (Szegedy et al., 2013; Goodfellow et al., 2014).

While the field has primarily focused on the development of new attacks and defenses, a ‘cat-and-mouse’ game between attacker and defender has arisen. There has been a long list of proposed defenses to mitigate the effect of adversarial examples, e.g., defenses (Papernot et al., 2016; Xu et al., 2017b; Buckman et al., 2018; Dhillon et al., 2018), followed by successful attacks (Carlini & Wagner, 2016; Athalye et al., 2018; Uesato et al., 2018). This shows that any defense mechanism that once looks successful could be circumvented with the invention of new attacks. In this paper, we tackle the problem by identifying a more fundamental cause of the adversarial vulnerability of DNNs.

What makes DNNs vulnerable to adversarial attacks? Our intuition is that the adversarial vulnerability of a DNN comes from the distortion in the latent feature space incurred by the adversarial perturbation. If a perturbation at the input level is suppressed successfully at any layers of the DNN, then it would not cause miss-prediction. However, not all latent features will contribute equally to the distortion; some features may have more substantial distortion, by amplifying the perturbations at the input level, while others will remain relatively static. Under this motivation, we first formally define the vulnerability of the latent features and show that sparse networks can have a much smaller degree of network-level vulnerability in Figure 2(a). Then, we propose an effective way to suppress vulnerability by pruning the latent features with high vulnerability. We refer to this defense mechanism as Adversarial Neural Pruning (ANP). Further, we propose a novel loss, which we refer to as Vulnerability Suppression (VS) loss, that directly suppresses the vulnerability in the feature space by minimizing the feature-level vulnerability. To this end, we propose a novel defense mechanism coined, Adversarial Neural Pruning with Vulnerability Suppression (ANP-VS), that learns pruning masks for the features in a Bayesian framework to minimize both the adversarial loss and the feature-level vulnerability, as illustrated in Figure 1. It effectively suppresses the distortion in the latent feature space with almost no extra cost relative to adversarial training and yields light-weighted networks that are more robust to adversarial perturbations.
Adversarial Neural Pruning with Latent Vulnerability Suppression

Figure 1. Concept: We hypothesize that the distortion in the latent feature space is the leading cause of the adversarial vulnerability of deep neural networks and introduce a novel Vulnerability Suppression (VS) loss that explicitly aims to minimize feature-level distortions. Further, we prune the features with significant distortions by learning pruning masks with Adversarial Neural Pruning (ANP) that minimizes the adversarial loss.

In summary, the contributions of this paper are as follows:

- We hypothesize that the distortion in the latent features is the leading cause of DNN’s susceptibility to adversarial attacks and formally describe the concepts of the vulnerability of latent-features based on the expectation of the distortion of latent-features with respect to input perturbations.

- Based on this finding, we introduce a novel defense mechanism, Adversarial Neural Pruning with Vulnerability Suppression (ANP-VS), to mitigate the feature-level vulnerability. The resulting framework learns a Bayesian pruning (dropout) mask to prune out the vulnerable features while preserving the robust ones by minimizing the adversarial and Vulnerability Suppression (VS) loss.

- We experimentally validate our proposed method on MNIST, CIFAR-10, and CIFAR-100 datasets, on which it achieves state-of-the-art robustness with a substantial reduction in memory and computation, with qualitative analysis which suggests that the improvement on robustness comes from its suppression of feature-level vulnerability.

While our primary focus is on improving the robustness of DNNs, we also found that ANP-VS achieves higher accuracy for clean/non-adversarial inputs, compared to baseline adversarial training methods (see the results of CIFAR datasets in Table 1). This is another essential benefit of ANP-VS as it has been well known that adversarial training tends to hurt the clean accuracy (Schmidt et al., 2018; Tsipras et al., 2019; Zhang et al., 2019).

2. Related work

Adversarial robustness. Since the literature on the adversarial robustness of neural networks is vast, we only discuss some of the most relevant studies. A large number of defenses (Papernot et al., 2016; Xu et al., 2017a; Buckman et al., 2018; Dhillon et al., 2018; Song et al., 2018) have been proposed and consequently broken by more sophisticated attack methods (Carlini & Wagner, 2016; Athalye et al., 2018; Uesato et al., 2018). Adversarial training based defenses (Madry et al., 2018; Liu et al., 2019; Hendrycks et al., 2019; Zhang et al., 2019) are widely considered to be the most effective since they largely avoid the obfuscated gradients problem (Athalye et al., 2018). There has also been previous work that studied robust and vulnerable features at the input level. Garg et al. (2018) established a relation between adversarially robust features and the spectral property of the geometry of the dataset and Gao et al. (2017) proposed to remove unnecessary features to get robustness. Recently, Ilyas et al. (2019) disentangle robust and non-robust features in the input-space. Our work is different from these existing works in that we consider and define the vulnerability at the latent feature level, which is more directly related to the model prediction.

Sparsification methods. Sparsification of neural networks is becoming increasingly important with the increased deployments of deep network models to resource-limited devices. The most straightforward way to sparsify neural networks is by removing weights with small magnitude (Ström, 1997; Collins & Kohli, 2014); however, such heuristics-based pruning often degenerates accuracy, and Han et al. (2015) proposed an iterative retraining and pruning approach to recover from the damage from pruning. Using sparsity-inducing regularization (e.g. $\ell_1$) is another popular approach for network sparsification. However elementwise sparsity does not yield practical speed-ups and Wen et al. (2016) proposed to use group sparsity to drop a neuron or a filter as a whole, that will reduce the actual network size. Molchanov et al. (2017) proposed to learn the individual dropout rates per weight with sparsity-inducing priors to completely drop out unnecessary weights, and Neklyudov et al. (2017) proposed to exploit structured sparsity by learning masks for each neuron or filter. Lee et al. (2018) proposed a variational dropout whose dropout probabilities are drawn from sparsity-inducing beta-Bernoulli prior. Information-theoretic approaches have been also shown to be effective, such as Dai et al. (2018) which minimizes the information theoretic bound to reduce the redundancy between layers.
3. Robustness of deep representations

We first briefly introduce some notations and the concept of vulnerability in the deep latent representation space. We represent a $L$-layer neural network by a function $f : X \rightarrow Y$ with dataset denoted by $D = \{X, Y\}$. Specifically, we denote any sample instance by $x \in X$, and its corresponding label by $y$. The output vector $f_\theta(x)$ can then be represented by $f_\theta(x) = f_{L-1}(f_{L-2}(\cdots(f_1(x))))$ where, $\theta = \{W_1, \ldots, W_{L-1}, b_1, \ldots, b_L\}$.

Let $z_l$ denote the latent-feature vector for the $l$-th layer with rectified linear unit (ReLU) as the activation function, then $f_l(\cdot)$ can be defined as:

$$z_{l+1} = f_l(z_l) = \max\{W_lz_l + b_l, 0\}, \quad \forall l \in \{1, 2, \ldots, L-2\}$$

where $W_l$ and $b_l$ denote the weight and bias vector respectively. Let $x$ and $\tilde{x}$ denote clean and adversarial data points, respectively ($\tilde{x} = x + \delta$) for any $x \in X$ with $\ell_p$-ball $B(x, \varepsilon)$ around $x : \{\tilde{x} \in X : \|\tilde{x} - x\| \leq \varepsilon\}$. $z_l$ and $\tilde{z}_l$ as their corresponding latent-feature map vectors for the $l$-th layer of the network.

**Robustness and sparsity.** The sparsity and robustness have been explored together in various recent works. Guo et al. (2018) analyzes sparsity and robustness from a theoretical and experimental perspective and demonstrate that appropriately higher sparsity leads to a more robust model and Ye et al. (2018) experimentally discuss how pruning shall effect robustness with a similar conclusion. In contrary, Wang et al. (2018) derived opposite conclusions showing that robustness decreases with increase in sparsity. Ye & Xu (2019) proposed concurrent adversarial training and weight pruning to enable model compression while preserving robustness. However, ANP-VS is fundamentally different from all these methods; we sparsify networks while explicitly learning the pruning (dropout) mask to minimize the loss on adversarial examples.

**Vulnerability of a latent-feature.** The vulnerability of a $k$-th latent-feature for $l$-th layer can be measured by the distortion in that feature in the presence of an adversary. The vulnerability of a latent feature could then be defined as the expectation of the Manhattan distance between the feature value for a clean example ($z_{lk}$) and its adversarial example ($\tilde{z}_{lk}$). This could be formally defined as follows:

$$v(z_{lk}, \tilde{z}_{lk}) = \mathbb{E}_{(x, y) \sim D}[|z_{lk} - \tilde{z}_{lk}|]$$

**Vulnerability of a network.** We can measure the vulnerability of an entire network $f_\theta(X)$ by computing the sum of the vulnerability of all the latent features vectors of the network before the logit layer, then $V(f_\theta(X), f_\theta(\tilde{X}))$ can be defined as:

$$V(f_\theta(X), f_\theta(\tilde{X})) = \frac{1}{L - 2} \sum_{l=1}^{L-2} V_l$$

where $V_l$ represents the vulnerability of the layer $l$ composed of $N_l$ latent-features. Figure 2(a) shows the vulnerability of different networks across various datasets. It can be observed that although adversarial training suppresses the vulnerability at the input level, the latent feature space is still vulnerable to adversarial perturbation and that our proposed method achieves the minimum distortion.

**Adversarial training.** Adversarial Training (Goodfellow et al., 2014; Kurakin et al., 2016) was proposed as a data augmentation method to train the network on the mixture of clean and adversarial examples until the loss converges. Madry et al. (2018) incorporated the adversarial search inside the training process by solving the following non-convex outer minimization problem and a non-concave inner
maximization problem:
\[
\min_{\theta} \mathbb{E}_{(x,y) \sim \mathcal{D}} \left[ \max_{\delta \in \mathcal{B}(x,c)} \mathcal{L}(\theta, \tilde{x}, y) \right]
\]

Figure 2 shows that adversarial training can distinguish between robust and vulnerable features. While the standard training results in obtaining vulnerable features, adversarial training reduces the vulnerability of the latent-features and selects the robust features which are necessarily required to attain adversarial robustness.

Relationship between robustness and sparsity. Guo et al. (2018); Ye et al. (2018) shows that sparsifying networks leads to more robust networks, which is evident by our definitions: sparsity suppresses vulnerability to 0 and thus reduces the network vulnerability. Ye & Xu (2019) uses a pre-trained adversarial defense model and investigated different pruning schemes for robustness. However, the network still does not take into account the robustness of a latent-feature. To address these limitations, we introduce ANP-NS in the next section, which is a novel method to prune and suppress these vulnerable features and only requires a standard network. In the experiments section, we show that our method significantly outperforms their method in adversarial robustness and computational efficiency.

4. Adversarial neural pruning with vulnerability suppression

In this section, we propose our method to reduce the vulnerability in the latent space. Let \( \mathcal{L}(\theta \odot M, x, y) \) be the loss function at data point \( x \) with class \( y \) for any \( x \in \mathcal{X} \) for the model with parameters \( \theta \) and mask parameters \( M \), we use Projected Gradient Descent (PGD) (Madry et al., 2018) to generate the adversarial examples:
\[
\tilde{x}^{k+1} = \prod_{B(x,\varepsilon)} (\tilde{x}^{k} + \alpha \cdot \text{sgn}(\nabla_x \mathcal{L}(\theta \odot M, \tilde{x}^{k}, y)))
\]

where \( \alpha \) is the step size, \( \text{sgn}(\cdot) \) returns the sign of the vector, \( \odot \) is the hadamard product, \( \prod(\cdot) \) is the projection operator on \( \ell_{\infty} \) norm-ball \( B(x,\varepsilon) \) around \( x \) with radius \( \varepsilon \) for each example, and \( x^{k+1} \) denotes the adversarial example at the \( k \)-th PGD step. Specifically, PGD perturbs the clean example \( x \) for \( K \) number of steps, and projects the adversarial example \( \tilde{x}^{k+1} \) onto the norm-ball of \( x \), if it goes beyond \( B(x,\varepsilon) \) after each step of perturbation.

In order to minimize the vulnerability of the network, we first propose Adversarial Neural Pruning (ANP). The basic idea of ANP is to achieve robustness while suppressing the distortion, by explicitly pruning out the latent features with high distortion. Specifically, ANP learns pruning masks for the features in a Bayesian framework to minimize the following adversarial loss:
\[
\min_{M} \mathbb{E}_{(x,y) \sim \mathcal{D}} \left[ \max_{\delta \in \mathcal{B}(x,c)} \mathcal{L}(\theta \odot M, \tilde{x}, y) \right]
\]

We further combine our proposed pruning scheme with our novel vulnerability suppression loss (VS) that minimizes the network vulnerability and further improves the robustness of the model. Let \( J(\theta \odot M, x, y) \) be the adversarial training loss on a batch of samples (comprised of the cross-entropy loss on the clean and adversarial samples, plus any weight decay, etc.). In particular, Adversarial Neural Pruning with Vulnerability Suppression (ANP-NS) optimizes the network parameters \( \theta \) using the following loss:
\[
\min_{\theta} \mathbb{E}_{(x,y) \sim \mathcal{D}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left( J(\theta \odot M, x, y) + \lambda \cdot V(f_{\theta}(x), f_{\theta}(\tilde{x})) \right) \right\}
\]

where \( \lambda \) is the hyper-parameter determining the strength of the vulnerability suppression loss. The overall training algorithm is displayed in Algorithm 1.

Intuition behind ANP-NS. ANP in Equation (5) encourages the pruning mask to be optimized by minimizing the adversarial loss and preserving the robustness of the model. The VS loss encourages the suppression of the distortion of the latent features in the presence of adversarial perturbations via minimizing the difference between latent-features of clean examples and those of adversarial examples. We empirically found that ANP-NS also has an effect of increasing the smoothness of the model’s output and its loss surface, which is conceptually consistent with the indispensable properties of robust models (Cisse et al., 2017; Tsipras et al., 2019) (see Figure 5).

Algorithm 1 Adversarial training by ANP-NS

input Dataset \( \mathcal{D} \), training iterations \( T \), trained model \( f_{\theta} \) with parameters \( \theta \), mask parameters \( M \), batch size \( n \), PGD iterations \( K \), and PGD step size \( \alpha \) for some norm-ball \( B \).

output Pruned state of network \( f_{\theta} \)

1: for \( t = \{1,\ldots,T\} \) do
2: Sample mini-batch \( B = \{x_1,\ldots,x_m\} \subset \mathcal{D} \)
3: for \( i = \{1,\ldots,n\} \) do
4: // Run PGD adversary
5: for \( k = \{1,\ldots,K\} \) do
6: \( \tilde{x}^{k+1} = \prod_{B(x_i,\varepsilon)} (\tilde{x}^{k} + \alpha \cdot \text{sgn}(\nabla_x \mathcal{L}(\theta \odot M, \tilde{x}^{k}, y))) \)
7: end for
8: Optimize \( \theta \) by Equation 6 and \( M \) by Equation 5 using gradient descent.
9: end for
10: end for
Adversarial beta-Bernoulli dropout. Beta-Bernoulli Dropout (Lee et al., 2018) learns to set the dropout rate by generating the dropout mask from sparsity-inducing beta-Bernoulli prior to each neuron. Let $\theta \in \mathbb{R}^{K \times L \times M}$ be a parameter tensor of neural network layer with $K$ channels and $M = \{m_1, \ldots, m_n\}$ for any $m \in \{0, 1\}^K$ be the binary mask sampled from the finite-dimensional beta-Bernoulli prior to be applied for the $n$-th observation $x_n$.

The goal of the variational inference is to compute the posterior distribution $p(\theta, M, \pi | D)$ and approximate this posterior using an approximate variational distribution $q(\theta, M, \pi | X)$ of known parametric form. For $\pi$, we use the Kumaranwamy distribution (Kumaranwamy, 1980) with parameters $a$ and $b$ following the Kumaranwamy distribution with known parametric form. For $\theta$, we use the Adversarial Neural Pruning (Lee et al., 2018); Nalisnick & Smyth (2016), and $M$.

Concurrent weight pruning and adversarial training on a pre-trained base network (Liu et al., 2019).

5. Experiments

5.1. Experimental setup

Baselines and our model. We first introduce various baselines and our model. We compare against following adversarial robustness and compression baselines:

2. Bayesian Pruning (BP). Base network with Beta-Bernoulli dropout (Lee et al., 2018).
4. Adversarial Bayesian Network (AT BNN). Adversarial Bayesian trained network (Liu et al., 2019).
5. Pre-trained Adversarial Training (Pretrained AT). Adversarial training on a pre-trained base model (Hendrycks et al., 2019).
7. Theoretically Principled Trade-off between Robustness and Accuracy (TRADES). Explicit trade off between natural and robust generalization (Zhang et al., 2019).

Datasets. We validate our method on following benchmark datasets for adversarial robustness:

1. MNIST. This dataset (LeCun, 1998) contains 60,000 grey scale images of handwritten digits sized $28 \times 28$, where there are 5,000 training instances and 1,000 test instances per class. As for the base network, we use LeNet 5-Caffe 1 for this dataset.
2. CIFAR-10. This dataset (Krizhevsky, 2012) consists of 60,000 images sized $32 \times 32$, from ten animal and vehicle classes. For each class, there are 5,000 images for training and 1,000 images for test. We use VGG-16 (Simonyan & Zisserman, 2014) for this dataset with 13 convolutional and two fully connected layers with pre-activation batch normalization and Binary Dropout.
3. CIFAR-100. This dataset (Krizhevsky, 2012) also consists of 60,000 images of $32 \times 32$ pixels as in CIFAR-10 but has 100 generic object classes instead of 10. Each class has 500 images for training and 100 images for test. We use VGG-16 (Simonyan & Zisserman, 2014) similar to CIFAR-10 dataset as the base network for this dataset.

1https://github.com/BVLC/caffe/tree/master/examples/mnist
### Evaluation setup

We validate our model with three metrics for computational efficiency: i) **Memory footprint** (Memory) - The ratio of space for storing hidden feature maps in pruned model versus original model. ii) **Floating point operations** (xFLOPs) - The ratio of the number of floating-point operations for the original model versus pruned model. iii) **Model size** (Sparsity) - The ratio of the number of zero units in the original model versus the pruned model. We report the clean, adversarial accuracy and vulnerability (Equation 2) for \( \ell_\infty \) white box and black box attack (Papernot et al., 2017). For generating black-box adversarial examples, we used AT and standard network for adversarial training and standard methods respectively. We utilize the clean and adversarial examples for adversarial training methods, all our results are measured by computing mean and standard deviation across 5 trials with random seeds. We list the hyper-parameters in the supplementary material, and the code is available online.²

### 5.2. Comparison of robustness and generalization

#### Evaluation on MNIST

For MNIST, we consider a Lenet-5-Caffe model with a perturbation radius of \( \varepsilon = 0.3 \), perturbation step of 0.01, 20 PGD steps for training, and 40 PGD steps with random restarts for evaluating the trained model. Our pretrained standard Lenet 5-Caffe baseline model reaches over 99.29% accuracy after 200 epochs averaged across five runs. The results in Table 1 show that ANP-VS significantly improves the adversarial accuracy and vulnerability of the network over all the competing baselines. Namely, with the robust-features, ANP-VS achieves \( \sim 3\% \) improvement in adversarial accuracy and \( \sim 65\% \) reduction in the vulnerability against white-box and black-box attacks over the state-of-the-art adversarial training frameworks. In addition, it achieves \( \sim 93\% \) reduction in memory footprint with significant speedup over standard adversarial training baselines. The results indicate that ANP-VS better suppresses the vulnerability of the network, which we can directly attribute to increased adversarial robustness.

<table>
<thead>
<tr>
<th>Model</th>
<th>Clean accuracy (( \uparrow ))</th>
<th>Adversarial accuracy (( \uparrow ))</th>
<th>Vulnerability (( \downarrow ))</th>
<th>Computational efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>White box attack</td>
<td>Black box attack</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>White box attack</td>
<td>Black box attack</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Memory (( \downarrow ))</td>
<td>xFLOPs (( \uparrow ))</td>
<td>Sparsity (( \uparrow ))</td>
</tr>
<tr>
<td>Standard</td>
<td>99.29±0.02</td>
<td>0.00±0.00</td>
<td>8.02±0.9</td>
<td></td>
</tr>
<tr>
<td>BP</td>
<td>99.34±0.05</td>
<td>0.00±0.00</td>
<td>12.99±0.5</td>
<td></td>
</tr>
<tr>
<td>AT</td>
<td>99.14±0.02</td>
<td>88.03±0.07</td>
<td>94.18±0.8</td>
<td></td>
</tr>
<tr>
<td>AT BNN</td>
<td>99.16±0.05</td>
<td>88.44±0.04</td>
<td>94.87±0.2</td>
<td></td>
</tr>
<tr>
<td>Pretrained AT</td>
<td>99.18±0.06</td>
<td>88.26±0.06</td>
<td>94.49±0.7</td>
<td></td>
</tr>
<tr>
<td>ADMM</td>
<td>99.01±0.02</td>
<td>88.47±0.04</td>
<td>94.61±0.7</td>
<td></td>
</tr>
<tr>
<td>TRADES</td>
<td>99.07±0.04</td>
<td>89.67±0.04</td>
<td>95.04±0.6</td>
<td></td>
</tr>
<tr>
<td>ANP-VS (ours)</td>
<td>99.05±0.08</td>
<td>91.31±0.09</td>
<td>95.43±0.8</td>
<td></td>
</tr>
<tr>
<td>Standard</td>
<td>92.76±0.1</td>
<td>13.79±0.8</td>
<td>41.65±0.9</td>
<td></td>
</tr>
<tr>
<td>BP</td>
<td>92.91±0.1</td>
<td>14.30±0.5</td>
<td>42.88±1.3</td>
<td></td>
</tr>
<tr>
<td>AT</td>
<td>87.50±0.5</td>
<td>49.85±0.9</td>
<td>63.70±0.6</td>
<td></td>
</tr>
<tr>
<td>AT BNN</td>
<td>86.69±0.5</td>
<td>51.87±0.9</td>
<td>64.92±0.9</td>
<td></td>
</tr>
<tr>
<td>Pretrained AT</td>
<td>87.50±0.4</td>
<td>52.25±0.7</td>
<td>66.10±0.8</td>
<td></td>
</tr>
<tr>
<td>ADMM</td>
<td>78.15±0.7</td>
<td>47.37±0.6</td>
<td>62.15±0.8</td>
<td></td>
</tr>
<tr>
<td>TRADES</td>
<td>80.33±0.5</td>
<td>52.08±0.7</td>
<td>64.80±0.5</td>
<td></td>
</tr>
<tr>
<td>ANP-VS (ours)</td>
<td>88.18±0.5</td>
<td>56.21±0.1</td>
<td>71.44±0.6</td>
<td></td>
</tr>
<tr>
<td>Standard</td>
<td>67.44±0.7</td>
<td>2.81±0.2</td>
<td>14.94±0.8</td>
<td></td>
</tr>
<tr>
<td>BP</td>
<td>69.40±0.7</td>
<td>3.12±0.1</td>
<td>16.39±2.2</td>
<td></td>
</tr>
<tr>
<td>AT</td>
<td>57.79±0.8</td>
<td>19.07±0.8</td>
<td>32.47±1.4</td>
<td></td>
</tr>
<tr>
<td>AT BNN</td>
<td>53.75±0.7</td>
<td>19.40±0.6</td>
<td>30.38±0.2</td>
<td></td>
</tr>
<tr>
<td>Pretrained AT</td>
<td>57.14±0.9</td>
<td>19.86±0.6</td>
<td>35.42±0.4</td>
<td></td>
</tr>
<tr>
<td>ADMM</td>
<td>52.52±0.5</td>
<td>19.65±0.5</td>
<td>31.30±0.3</td>
<td></td>
</tr>
<tr>
<td>TRADES</td>
<td>56.70±0.7</td>
<td>21.21±0.3</td>
<td>32.81±0.6</td>
<td></td>
</tr>
<tr>
<td>ANP-VS (ours)</td>
<td>59.15±1.2</td>
<td>22.35±0.6</td>
<td>37.01±1.1</td>
<td></td>
</tr>
</tbody>
</table>

²[https://github.com/divyam3897/ANP_VS](https://github.com/divyam3897/ANP_VS)
Table 2. Adversarial accuracy of CIFAR-10 and CIFAR-100 for VGG-16 architecture under $\ell_{\infty}$-PGD white-box attack for various $\varepsilon$ values and PGD iterations with the perturbation per step of 0.007. The best results are highlighted in bold.

<table>
<thead>
<tr>
<th>Epsilon ($\varepsilon$)</th>
<th># Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>0.03</td>
</tr>
<tr>
<td>Standard</td>
<td>40.54±0.9</td>
</tr>
<tr>
<td>BP</td>
<td>41.67±0.6</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>AT</td>
</tr>
<tr>
<td></td>
<td>AT BNN</td>
</tr>
<tr>
<td></td>
<td>Pretrained AT</td>
</tr>
<tr>
<td></td>
<td>ADMM</td>
</tr>
<tr>
<td></td>
<td>TRADES</td>
</tr>
<tr>
<td>ANP-VS (ours)</td>
<td>72.33±0.4</td>
</tr>
<tr>
<td>CIFAR-100</td>
<td>Standard</td>
</tr>
<tr>
<td></td>
<td>BP</td>
</tr>
<tr>
<td></td>
<td>AT</td>
</tr>
<tr>
<td></td>
<td>AT BNN</td>
</tr>
<tr>
<td></td>
<td>Pretrained AT</td>
</tr>
<tr>
<td></td>
<td>ADMM</td>
</tr>
<tr>
<td></td>
<td>TRADES</td>
</tr>
<tr>
<td>ANP-VS (ours)</td>
<td>35.70±0.8</td>
</tr>
</tbody>
</table>

Evaluation on CIFAR-10 and CIFAR-100. In order to show that ANP-VS is capable of scaling to more complex datasets, we evaluate our method on CIFAR-10 and CIFAR-100 dataset with VGG-16 architecture. We use $\varepsilon = 0.03$, 10 PGD steps for training and 40 steps with random restart for evaluation. The results are summarized in Table 1. We make the following observations from the results: (1) ANP-VS achieves $\sim 7\%$ improvement in adversarial accuracy, $\sim 58\%$ reduction in vulnerability of the network against white-box and black-box attack over all the baselines for both the datasets. (2) Our proposed method is not only effective on adversarial robustness but also outperforms all the compared adversarial baselines on the standard generalization performance, and strongly support our hypothesis of obtaining robust features. (3) Compared to the standard adversarial training baselines, our method shows a reduction $\sim 85\%$ in the memory-footprint with $2\times$ fewer FLOPS, demonstrating the effectiveness of our method.

One might also be concerned regarding the number of PGD steps and different $\varepsilon$ values as for certain defenses; the robustness decreased as the number of PGD steps were increased (Engstrom et al., 2018). Table 2 shows the results for different $\ell_{\infty}$ epsilon values and PGD steps up to 1000. Remarkably, in both scenarios, our proposed defense outperforms all the compared baselines, illustrating the practicality of our method. These results indicate that even if the attacker uses greater resources to attack, the effect on our proposed method is negligible. This allows us to conclude that ANP-VS can induce reliable and effective adversarial robustness with much less computation cost.

5.3. Ablation studies

Analysis of individual components. To further gain insights into the performance of ANP-VS, we perform ablation studies to dissect the effectiveness of various components (ANP and VS) for robustness in Table 3. First, we study the impact of VS loss on AT, whether VS produces better robustness, and curtails the vulnerability of the network. One can observe that AT-VS improves the adversarial accuracy ($\sim 3\%$) and leads to $\sim 50\%$ reduction in vulnerability over AT for MNIST and CIFAR-10 dataset. In addition, note that the clean accuracy for AT-VS is approximately the same as AT across all the datasets, which indicates that VS may have a limited effect on standard accuracy, but has a substantial impact on robustness.

Second, we investigate the impact of ANP on AT, whether pruning the vulnerable latent-features is more effective than merely suppressing the vulnerability of those features. Table 3 shows that ANP not only improves the robustness ($\sim 10\%$) but also the standard generalization ($\sim 3\%$) over AT. Furthermore, we can observe that ANP on AT significantly outperforms VS on AT with CIFAR-10 dataset by $\sim 4\%$ on the adversarial accuracy while significantly reducing the computational cost. This observation supports our conjecture that ANP incorporates the vulnerability of latent-features, and pruning the vulnerable latent-features improves robustness. We know from these results that using VS or ANP can already outperform AT, while incorporating them together in ANP-VS additionally further boosts the performance, as testified.
Robustness and sparsity. Our proposed method could be vital when we want to obtain a lightweight yet robust network. To show that we can achieve both goals at once, we experiment with different scaling coefficient for the KL term in Equation (9) to obtain architectures with varying degrees of sparsity, whose details can be found in the supplementary material.

Figure 3 clearly shows that ANP-VS outperforms AT up to a sparsity level of $\sim 80\%$ for CIFAR-10 and CIFAR-100 after which there is a decrease in the robust and standard generalization. The results are not surprising, as it is an overall outcome of the model capacity reduction and the removal of the robust features. We further analyze a baseline PAT where we first perform Bayesian pruning, freeze the dropout mask, and then perform AT (see Figure 3). We can observe that PAT marginally improves the robustness over AT but loses on clean accuracy. Note that ANP-VS significantly outperforms PAT, supporting our hypothesis that just naive approach of AT over pruning can hurt performance.

5.4. Further analysis on defense performance

Vulnerability analysis. We conduct additional experiments to visualize the vulnerability of the latent-feature space to investigate the robust and vulnerable features. Figure 4 (Top) compares the vulnerability of various models. One can observe that the latent-features of the standard model are the most vulnerable, and the vulnerability decreases with the AT and is further suppressed by half with ANP-VS. Further, note that the latent features of our proposed method capture more local information of the objects and align much better with human perception.

In addition, Figure 4 (Bottom) shows the histogram of the vulnerability of latent-features for input-layer defined in Equation (1) for various methods. We consistently see that standard Bayesian pruning zeros out some of the distortions in the latent-features, and AT reduces the distortion level of all the latent-features. On the other hand, ANP-VS does both, with the largest number of latent-features with zero distortion and low distortion level in general. In this vein, we can say that ANP-VS demonstrates the effectiveness of our method as a defense mechanism by obtaining robust features utilizing pruning and latent vulnerability suppression.

Loss landscape visualization. It has been observed that adversarial training flattens the adversarial loss landscape and obfuscated gradients can often lead to bumpier loss landscapes (Engstrom et al., 2018). To investigate this problem of obfuscated gradients, we visualize the adversarial loss landscape of the baseline models and ANP-VS in Figure 5. We vary the input along a linear space defined by the sign of gradient where x and y-axes represent the perturbation added in each direction, and the z-axis represents the loss.

We can observe that the loss is highly curved in the vicinity of the data point x for the standard networks, which reflects that the gradient poorly models the global landscape. On the other hand, we observe that both sparsity and adversarial training make the loss surface smooth, with our model obtaining the smoothest surface. It demonstrates that our proposed method is not susceptible to the problem of gradients obfuscation. In addition, it indicates that suppressing the network vulnerability leads to robust optimization, uncovering some unexpected benefits of our proposed method over adversarial training.

Table 3. Ablation studies for Vulnerability Suppression loss (VS) and Adversarial Neural Pruning (ANP) to investigate the impact of VS and ANP against $\ell_\infty$ white-box PGD attack. ↑ (↓) indicates that the higher (lower) number is the better. The best results are highlighted in bold.

<table>
<thead>
<tr>
<th>Model</th>
<th>Clean Acc. (↑)</th>
<th>Adv. acc. (↑)</th>
<th>Vulnerability (↓)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT</td>
<td>99.14 ± 0.02</td>
<td>80.03 ± 0.7</td>
<td>0.045 ± 0.001</td>
</tr>
<tr>
<td>AT-VS</td>
<td>99.24 ± 0.05</td>
<td>90.36 ± 0.8</td>
<td>0.022 ± 0.001</td>
</tr>
<tr>
<td>ANP</td>
<td>98.64 ± 0.07</td>
<td>90.12 ± 0.5</td>
<td>0.020 ± 0.002</td>
</tr>
<tr>
<td>ANP-VS</td>
<td>99.05 ± 0.08</td>
<td>91.31 ± 0.9</td>
<td>0.011 ± 0.002</td>
</tr>
<tr>
<td>MNIST</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AT</td>
<td>87.50 ± 0.5</td>
<td>49.85 ± 0.9</td>
<td>0.050 ± 0.002</td>
</tr>
<tr>
<td>AT-VS</td>
<td>87.44 ± 0.5</td>
<td>51.52 ± 0.4</td>
<td>0.024 ± 0.002</td>
</tr>
<tr>
<td>ANP</td>
<td>88.36 ± 0.4</td>
<td>55.63 ± 0.9</td>
<td>0.022 ± 0.001</td>
</tr>
<tr>
<td>ANP-VS</td>
<td>88.18 ± 0.5</td>
<td>56.21 ± 1.0</td>
<td>0.016 ± 0.000</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AT</td>
<td>57.79 ± 0.8</td>
<td>19.07 ± 0.8</td>
<td>0.079 ± 0.003</td>
</tr>
<tr>
<td>AT-VS</td>
<td>57.74 ± 0.6</td>
<td>20.06 ± 0.9</td>
<td>0.061 ± 0.005</td>
</tr>
<tr>
<td>ANP</td>
<td>58.47 ± 1.02</td>
<td>22.20 ± 1.1</td>
<td>0.037 ± 0.003</td>
</tr>
<tr>
<td>ANP-VS</td>
<td>59.15 ± 1.2</td>
<td>22.35 ± 0.6</td>
<td>0.035 ± 0.001</td>
</tr>
</tbody>
</table>
Figure 4. Top: Visualization of the vulnerability of the latent-features with respect to the input pixels for various set of datasets. Bottom: Histogram of vulnerability of the features for the input layer for CIFAR-10 with the number of zeros shown in orange color.

Figure 5. Comparison of loss landscapes for various methods. We observe that our proposed method results in a much smoother and flattened loss surface compared to adversarial training. The z axis represents the loss projected along two random directions.

6. Conclusion

We hypothesized that the adversarial vulnerability of deep neural networks comes from the distortion in the latent feature space since if they are suppressed at any layers of the deep network, they will not affect the prediction. Based on this hypothesis, we formally defined the vulnerability of a latent feature and proposed Adversarial Neural Pruning (ANP) as a defense mechanism to achieve adversarial robustness as well as a means of achieving a memory- and computation-efficient deep neural networks. Specifically, we proposed a Bayesian formulation that trains a Bayesian pruning (dropout) mask for adversarial robustness of the network. Then we introduced Vulnerability Suppression (VS) loss, that minimizes network vulnerability. To this end, we proposed Adversarial Neural Pruning with Vulnerability Suppression (ANP-VS), which prunes the vulnerable features by learning pruning masks for them, to minimize the adversarial loss and feature-level vulnerability. We experimentally validate ANP-VS on three datasets against recent baselines, and the results show that it significantly improves the robustness of the deep network, achieving state-of-the-art results on all the datasets. Further qualitative analysis shows that our method obtains more interpretable latent features compared to standard counterparts, effectively suppresses feature-level distortions, and obtains smoother loss surface. We hope that our work leads to more follow-up works on adversarial learning that investigates the distortion in the latent feature space, which we believe is a more direct cause of adversarial vulnerability.
Acknowledgements

We thank the anonymous reviewers for their insightful comments and suggestions. We are also grateful to the authors of Lee et al. (2018) (Juho Lee, Saehoon Kim, and Hae Beom Lee)—for their insights and efforts replicating, extending and discussing our experimental results. This work was supported by Google AI Focused Research Award, Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2020-0-00153), Penetration Security Testing of ML Model Vulnerabilities and Defense), Engineering Research Center Program through the National Research Foundation of Korea (NRF) funded by the Korean Government (NRF-2018R1A5A1059921), Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-00075), and Artificial Intelligence Graduate School Program (KAIST). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

References


Han, S., Pool, J., Tran, J., and Dally, W. J. Learning both weights and connections for efficient neural networks. In NeurIPS, 2015.


Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. Adversarial examples are not bugs, they are features. In NeurIPS, 2019.


