Interpreting and Improving Adversarial Robustness with Neuron Sensitivity

Chongzhi Zhang∗, Aishan Liu∗†, Xianglong Liu†, Yitao Xu, Hang Yu, Yuqing Ma, Tianlin Li
State Key Laboratory of Software Development Environment, Beihang University, Beijing, China
{chongzhizhang,xlliu,mayuqing}@nlsde.buaa.edu.cn, {hyu0829,liuaishan,xuyitao}@buaa.edu.cn, litl@act.buaa.edu.cn

Abstract
Deep neural networks (DNNs) are vulnerable to adversarial examples where inputs with imperceptible perturbations mislead DNNs to incorrect results. Despite the potential risk they bring, adversarial examples are also valuable for providing insights into the weakness and blind-spots of DNNs. Thus, the interpretability of a DNN in adversarial setting aims to explain the rationale behind its decision-making process and makes deeper understanding which results in better practical applications. To address this issue, we try to explain adversarial robustness for deep models from a new perspective of neuron sensitivity which is measured by neuron behavior variation intensity against benign and adversarial examples. In this paper, we first draw the close connection between adversarial robustness and neuron sensitivities, as sensitive neurons make the most non-trivial contributions to model predictions in adversarial setting. Based on that, we further propose to improve adversarial robustness by constraining the similarities of sensitive neurons between benign and adversarial examples which stabilizes the behaviors of sensitive neurons in adversarial setting. Moreover, we demonstrate that state-of-the-art adversarial training methods improve model robustness by reducing neuron sensitivities which in turn confirms the strong connections between adversarial robustness and neuron sensitivity as well as the effectiveness of using sensitive neurons to build robust models. Extensive experiments on various datasets demonstrate that our algorithm effectively achieve excellent results.

Introduction

Deep Neural Network (DNNs) have demonstrated remarkable performance in a wide spectrum of areas, including computer vision (Krizhevsky, Sutskever, and Hinton 2012), speech recognition (Mohamed, Dahl, and Hinton 2012) and natural language processing (Chen and Manning 2014). Despite the significant achievements, unfortunately, the emergence of adversarial examples (Szegedy et al. 2014; Goodfellow, Shlens, and Szegedy 2015), images containing small perturbations imperceptible to human but extremely misleading to DNNs, casts a cloud over the recent progress of interpretation. Also, it poses potential security threats by attacking or misleading the practical deep learning applications like auto driving and face recognition system, which may cause pecuniary loss or even people death with severe impairment (Kurakin, Goodfellow, and Bengio 2017a; Liu et al. 2019).

In the past years, plenty of defense methods have been proposed to improve model robustness to adversarial examples, avoiding the potential danger in real world applications. These methods can be roughly categorized into adversarial training (Goodfellow, Shlens, and Szegedy 2015; Kurakin, Goodfellow, and Bengio 2017b; Madry et al. 2018), input transformation (Dziugaite, Ghahramani, and Roy 2016; Guo et al. 2018; Xie et al. 2018), elaborately designed model architectures (Papernot et al. 2016; Liao et al. 2018) and adversarial example detection (Lu, Issaranon, and Forsyth 2017; Xu, Evans, and Qi 2018).

From another point of view, adversarial examples are also valuable and beneficial for understanding the behaviors of DNNs, which is an extremely difficult problem in the literature due to the myriad of linear and nonlinear operations in the “blackbox”. Understanding adversarial examples provides insights into weaknesses and blind-spots of DNNs in adversarial settings, which in turn offers us clues to interpret and build robust deep learning models. Several studies have been proposed to reveal the “tip of the iceberg” of model robustness towards adversarial noises. Dong et al. re-examined the internal representations of DNNs using adversarial examples and further improved their interpretabilities with an adversarial training scheme. Meanwhile, Xu et al. attempted to explore the weakness of models under adversarial conditions by analyzing the effects of different regions within a specific image. More recently, Ilyas et al. believed that robust features can be extracted with the help of adversarially robust deep models.

At a high-level perspective, model robustness to noises can be viewed as a kind of global insensitivity property (Tsipras et al. 2019). A deep model is able to learn insensitive representations towards adversarial examples, if it behaves stably without too much performance degeneration, when encountering such noises. Based on this motivation, this paper tries to uncover the mysteries of model robust-
ness in adversarial setting from the perspective of behavior sensitivity in the neuron-wise way.

We first demonstrate the concept of Neuron Sensitivity that considers the variation intensity of neuron behaviors for adversarial and benign examples, as a criterion to measure the stability of a DNN in adversarial settings. We further define Sensitive Neuron, a set of neurons most sensitive to adversarial examples, which we believe may conduct the most non-trivial contributions to sensitive model behaviors. Based on those findings, we propose a method named Sensitive Neuron Stabilizing which directly improves model robustness in adversarial setting by forcing benign and adversarial examples to share the similar representations in sensitive neurons. Moreover, with a series of empirical studies on neuron sensitivities, we show the fact that the state-of-the-art adversarial training methods improve model robustness mainly by embedding insensitivity of neurons which in turn confirms the effectiveness of using sensitive neurons to build robust models. Extensive experiments on CIFAR-10 and ImageNet empirically demonstrate that such a simple technique significantly outperforms state-of-the-art adversarial training strategies.

Neuron Sensitivity and Sensitive Neuron

In this paper, our goal is to interpret the model robustness against adversarial examples. Prior studies have shown that different neurons play different roles and possess different importance for model prediction even arranged in the same layer ([Zhou et al. 2015], [Dong et al. 2017], [Bau et al. 2017]). Inspired by this fact, this section will study the model adversarial robustness from the view of neuron behaviors.

Given a dataset $D$ with data sample $x \in X$ and label $y \in Y$, the deep supervised learning model tries to learn a mapping or classification function $F: X \rightarrow Y$. The model $F$ consists of $L$ serial layers. For the $l$-th layer $F_l$, where $l = 1, \ldots, L$, it contains several neurons, which can also be regarded as a neuron set. We use superscript $m$ to denote the $m$-th neuron, and satisfying $F_l^m \in F_l$. The output of one neuron $F_l^m$ is equivalent to the $m$-th channel of the feature map produced by layer $l$. For model $F$, this paper chooses the popular deep convolutional neural networks (CNNs) for visual recognition task.

Neuron Sensitivity

The model robustness towards noises can be viewed as a global insensitive behavior showing small losses and consistent predictions under noise conditions. Recall the definition of model robustness in noise setting in ([Xu and Mannor 2012]), which are derived from the idea that if two instances are “similar” then their test errors are close, too:

$$\forall x_i, x_j \in D, \ s.t. \ ||x_i - x_j|| < \epsilon \Rightarrow \ ||\mathcal{L}_F(x_i) - \mathcal{L}_F(x_j)|| \leq \epsilon,$$

where $x_i$ and $x_j$ are samples selected randomly from the same dataset $D$ and $\mathcal{L}_F(\cdot)$ denotes the loss function. Meanwhile, $\|\cdot\|$ is a distance metric to quantify the distance between samples and $\epsilon$ denotes a small value.

This fact should hold for the benign sample $x \in D$ from the category $y$ and its adversarial example $x' \in D'$. However, in practice, the adversarial example misleads the non-robust classifier to predict wrong label which is defined as follows:

$$F(x') \neq y \ s.t. \ ||x - x'|| < \epsilon.$$

Intuitively, for a model that owns strong robustness, namely, insensitive to the adversarial examples, we expect that the benign sample $x$ and the corresponding adversarial example $x'$ share the similar representation in the hidden layers of the model, thus similar final predictions as well. Motivated by this intuition, to understand the adversarial robustness of deep model, one can concentrate on the deviation of the feature representation in hidden layers between benign samples and corresponding adversarial examples.

To achieve this goal, we introduce Neuron Sensitivity to interpret the model sensitivity from the view of neurons inside it. Specifically, given a benign example $x_i$, where $i = 1, \ldots, N$, from $D$ and its corresponding adversarial example $x'_i$ from $D'$, we can get the dual pair set $D = \{(x_i, x'_i)\}$, and then calculate the neuron sensitivity $\sigma$ as follows:

$$\sigma(F_l^m, D) = \sum_{i=1}^{N} ||F_l^m(x_i) - F_l^m(x'_i)||_1,$$

where $F_l^m(x_i)$ and $F_l^m(x'_i)$ respectively represent the output of neuron $F_l^m$ towards benign sample $x_i$ and corresponding adversarial example $x'_i$ during the forward process. Here, we use the $L_1$ norm as the distance metric empirically.

Sensitive Neuron

Once we have defined Neuron Sensitivity, we can further determine the most prominent neurons under this criterion and mark them as the Sensitive Neuron $\Omega_l$ as shown below:

$$\Omega_l = \text{top-k}(F_l, \sigma),$$

where top-k($\cdot$) represents top $k$ maximum instances of the input set based on certain metric, such as neuron sensitivity $\sigma$ in this case, which means $\Omega_l$ is a subset of $F_l$. This can be easily accomplished by traversing the neurons in each layer and selecting $k$ neurons with the maximal neuron sensitivity according to Equation [1] for N samples.
Obviously, sensitive neurons in each layer correspond to the vulnerable ones in a model towards adversarial examples, to which more concerns and attention should be given. Therefore, in the rest of this paper, we mainly focus on understanding the behaviors of sensitive neurons against adversarial examples, so that we can establish the connections between neuron sensitivity and model robustness, and further improve the model robustness with sensitive neurons. Figure 1 demonstrates the basic procedure of computing neuron sensitivity and selecting sensitive neurons.

Sensitive Neurons and Adversarial Robustness

Apart from viewing DNNs as a blackbox from a high-level viewpoint, it is natural for us to treat the model as a whitebox and make deeper insights into model adversarial weaknesses from the perspective of sensitive neurons. In this section, we first explore the strong connections between sensitive neurons and adversarial robustness. With several empirical analyses, we surprisingly find that sensitive neurons make the most non-trivial contributions towards model robustness in adversarial setting. In following part of this section, we will first illustrate the empirical settings, then give our analyses and discussions.

Empirical Settings

Datasets and models We adopt the widely used CIFAR-10 and ImageNet datasets. CIFAR-10 consists of 60K natural scene color images with 10 classes of size $32 \times 32 \times 3$ (Krizhevsky and Hinton 2009). We use VGG-16 (Simonyan and Zisserman 2015) and ResNet-18 (He et al. 2016) for CIFAR-10. ImageNet contains 14M images with more than 20k classes (Russakovsky et al. 2015). For simplicity, we only choose 200 classes from 1000 in ILSVRC-2012 with 100K and 10k images for training set and test set, respectively. The models we use for ImageNet are VGG-16 and AlexNet (Krizhevsky, Sutskever, and Hinton 2012).

Adversarial attack and defence methods We apply a diverse set of adversarial attack methods including: FGSM (Goodfellow, Shlens, and Szegedy 2015), Step-LL (Kurakin, Goodfellow, and Bengio 2017b), MI-FGSM (Dong et al. 2018), and PGD (Madry et al. 2018). As for adversarial defense models, we choose the state-of-the-art defense methods including naive adversarial training (NAT) (Kurakin, Goodfellow, and Bengio 2017b), PGD-based adversarial training (PAT) (Madry et al. 2018), ensemble adversarial training (EAT) (Tramèr et al. 2018) and adversarial logit pairing (ALP) (Kannan, Kurakin, and Goodfellow 2018). Among these methods, EAT achieved No.1 in round 1 in NeurIPS 2017 adversarial defense competition.

Sensitive Neurons Contribute Most to Model Predictions in Adversarial Setting

We first analyze the behaviors and contributions of sensitive neurons to model robustness in adversarial setting. In order to compute the contribution of neurons towards prediction, we borrow the idea of calculating the contribution from representation of penultimate layer $F_{l-1}$ to logits from (Engstrom et al. 2019); for an input adversarial example $x'$ with predicted class $y$ we can select the important neurons $\Pi$ by:

$$\Pi^y(x') = \text{top-k}(F_{l-1}, \varphi),$$

where metric $\varphi(F_{l-1}, x', y)$ measures the direct contribution from the output of $F_{l-1}$ to the logit of target class $y$. And it can be described as below:

$$\varphi(F_{l-1}, x', y) = F_{l-1}(x') \cdot W_{my},$$

where $W_{my}$ is the linear layer mapping from $m$-th representations to $y$-th logit. Obviously, the important neurons from penultimate layer denote those units that make the biggest contributions to the model prediction (target logit) for sample $x'$.

As important neurons are designed for single adversarial image, we further extend the idea from one adversarial example image to one adversarial example set $D'$ for a specific target class $y$:

$$\Gamma^y = \text{top-k}(F_{l-1}, \rho),$$

where metric $\rho$ means the total score of weighted voting by important neurons $\Pi^y(x_1), \ldots, \Pi^y(x_N)$ computed with respect to each adversarial image in $D'$. Specifically, the important neurons for image $x_i$ with higher rank will have
larger voting weight. In our experiment, we choose top-20 important neurons for adversarial example set $D'$ and also select the top-20 sensitive neurons for comparison. For VGG-16 model, we investigate the neurons from pool5 layer.

Then, we utilize the Jaccard index as the metric to quantify the overlaps and similarities between important neurons and sensitive neurons for the same adversarial example set of a specific target class $y$ from penultimate layer:

$$J(\Omega_{L-1}^y, \Gamma^y) = \frac{|\Omega_{L-1}^y \cap \Gamma^y|}{|\Omega_{L-1}^y \cup \Gamma^y|},$$

where $\Omega_{L-1}^y$ and $\Gamma^y$ respectively represents the top-$k$ sensitive neuron set and important neuron set for targeted adversarial example set of label $y$, and the notion $| \cdot |$ denotes the size of set.

As shown in Figure [2](a), for target label ranging from 0 to 9, the overlap rates of two sets are extremely high, which indicates that sensitive neurons and important neurons are quite similar. In other words, sensitive neurons make the most non-trivial contributions to model predictions in adversarial setting and uncover the strongest weakness for deep models.

We further explore the behaviors of sensitive neurons in adversarial settings by showing what they detect during inference through visualization studies. Following the work in [Zhou et al. 2015], we compute and demonstrate the receptive fields of different neurons, e.g., sensitive neurons and vanilla ones, on ImageNet with VGG-16 using discrepancy maps. For each neuron, we first find top-5 images with the highest activation in the benign sample set and then visualize the discrepancy maps with highlighted regions where a large discrepancy can be observed using the corresponding adversarial examples generated by PGD untargeted attack. According to the visualization results in Figure [3], we can see that after adversarial attack, sensitive neurons tend to pay more attention to the noisy backgrounds and other meaningless regions, compared to their subtle detection regions on benign examples. However, as shown in the Figure [4], the receptive fields of vanilla neurons show almost no differences between benign and adversarial examples. With the above observation, we can double confirm the conclusion that sensitive neurons are more sensitive to adversarial noises and play critical roles to model’s final decisions in adversarial setting. Experimental results of VGG-16 on CIFAR-10 and AlexNet on ImageNet reported in the appendix convey the same conclusion.

**Sensitive Neurons Reveal Vulnerable Directions to Adversarial Weakness**

We further explore characteristics of sensitive neurons at layers of different depths. MI-FGSM method is used to generate targeted adversarial examples on CIFAR-10 with VGG-16. For each class in CIFAR-10, 9000 adversarial samples are generated leading to 9000 corresponding dual pairs, which will be used to select sensitive neurons set $\Omega_{2}^{y}, \ldots, \Omega_{10}^{y}$, where $l$ denotes the index of layer we use. We use two variants of Jaccard index to measure the similarity of sensitive neurons sets in same layer, and the definitions of them can be found in the appendix.

As shown in Figure [5](b), we draw a very important observation that the sensitive neurons in different target set vary a lot in the top layers, but have high similarities in the bottom layers, though they are attacked by adversarial examples with different target labels. The reason may lie in the hierarchical information processing structure of DNNs, since bottom layers focus on common low-level features, e.g., edges and textures, while top layers care more about high-level semantic features to specific classes (Zeiler and Fergus 2014). This interesting finding indicates that different targeted adversarial examples tend to share the same flaws of bottom layers, while utilize different fragile neurons in the...
top layers. Moreover, according to the results, the most sensitive neuron of the linear layer for each target set exactly corresponds to the targeted attack label. This uncovers the essence of targeted adversarial attacks that these elaborately designed noises perform adversarial attack by increasing the logits of the target label, which is consistent with the goal of targeted attack: increasing the probability of target label. In conclusion, sensitive neurons effectively reveal vulnerable directions to adversarial weakness, which are the severe flaws leveraged by adversarial attacks. We also provide more details and experimental results in the appendix.

Sensitive Neurons Convey Strong Semantic Information

As we have moved so far, we prefer to move further to give more insights about the roles of sensitive neurons. Prior studies have shown the ability of using suppression and ablation skills to study individual unit functions within a model (Zhou et al. 2014, Zhou et al. 2015). Thus, in this section, we try to suppress the outputs of top-10% sensitive neurons by multiplying them with a coefficient $\beta$ after activation (Vanilla model is obtained when $\beta=1.0$).

As shown in Figure 5 at the beginning, with the decreasing of $\beta$ from 1.0, the adversarial robustness increases, while the clean example accuracy drops drastically. The model robustness nearly reaches the best when $\beta$ is around 0.85. After that, when we continue to decrease the value of $\beta$, unfortunately, the model performance collapses in terms of both robustness and clean accuracy, when $\beta$ reaches about 0.7. From the figure, we can conclude that: (1) there indeed exists a trade-off between robustness and accuracy (Tsipras et al. 2019). (2) Sensitive neurons are responsible for both clean accuracy and robustness, indicating that they can extract strong semantic features for deep models, and will hurt model predictions a lot when suppressed.

Improving Robustness with Sensitive Neurons

We have demonstrated the close connections between neuron sensitivity and adversarial robustness, as well as the basic properties of sensitive neurons in deep models. With the above observations and conclusions, it is natural to improve model robustness by stabilizing those sensitive neurons. Therefore, in this section, we first try to explore the reasons why state-of-the-art adversarial training strategies achieve strong robustness from the view of sensitive neurons. Then, we come up with a strategy called Sensitive Neuron Stabilizing (SNS) to alleviate the hazard brought by adversarial examples and improve model robustness by reducing the sensitivity of sensitive neurons.

Adversarial Training Builds Robust Models by Reducing Neuron Sensitivities

With the increasing concerns of adversarial examples to model robustness, plenty adversarial defense methods have been proposed including: adversarial training, input transformation, etc. However, as discussed in (Athalye, Carlini, and Wagner 2018), most of these defense strategies just give a false sense of safety, which could be attacked eas-
ily through obfuscated gradient circumvent. Whereas, adversarial training based methods, which augment training data with adversarial examples, are relatively immune to these attacks and achieve the most robust models so far. Based on that, a number of work have been proposed to study and explain the essence of adversarial training to model robustness. [Goodfellow, Shlens, and Szegedy] first introduced the adversarial training strategy to defense adversarial attack by somewhat reducing the linearity for high-dimensional DNNs. [Shaham, Yamada, and Negahban] tried to explain the performance of adversarial training from the view of robust optimization theory, which improves model robustness by increasing local stability.

Different from them, in this part, attempts have been made to interpret adversarial training from the perspective of neuron sensitivity. One important take-away is: adversarial training improves model robustness by embedding representation insensitivities.

To demonstrate this point, at the beginning, we respectively trained a Vanilla model and a PGD-based adversarial training model (PAT) using VGG-16 on CIFAR-10. The whitebox PGD, FGSM, Step-LL and MI-FGSM are applied as the attack method. Then, for a specific layer, we can rank its neurons according to their sensitivities. As shown in Figure [6], all neurons are extremely insensitive to adversarial examples on PAT, compared to the Vanilla model on each layer. This observation clarifies the reason why adversarial training methods are insensitive or robust to adversarial noises. Meanwhile, we witness one interesting phenomenon that the differences of neuron sensitivity between Vanilla and PAT for the top 10% of the neurons are distinctly larger than that of others, which proves the rationality of sensitive neuron. In other words, sensitive neuron is a significant indicator to represent model behaviors in adversarial setting between benign and adversarial examples.

After neuron-wise analysis, further investigation has been made in an overall view by layers. As shown in Figure [7] (a), under PGD attack, PAT obtains the lower sensitivity value of sensitive neurons (top 10%) with big margins compared to Vanilla model, which explains why PAT enjoys much higher adversarial robustness. Moreover, we find a very interesting phenomenon in Figure [7] (b) that in fog noise setting the neuron sensitivity values of PAT are higher than that of Vanilla model in most layers. Actually, the accuracy of Vanilla and PAT on fog noise is 33.64% and 55.64% respectively, which further confirms that the neuron sensitivity serves as a distinct indicator of model robustness.

In this point of view, we believe that, with the help of stable neurons, adversarial noises are more easily absorbed and filtered during the forward propagation process, due to the insensitive hidden representations, which in turn promises consistent model behaviors and thus strong robustness.

Training Adversarially Robust Models via Sensitive Neurons Stabilizing

Motivated by our observations, a straightforward idea for improving model robustness is to force the sensitive neurons of benign and adversarial ones to behave similarly. In other words, we try to stabilize those sensitive neurons, and thus the whole model will be insensitive to adversarial examples. The Sensitive Neurons Stabilizing (SNS for short) can be easily accomplished by directly adding a loss term to measure the similarities of sensitive neurons behaviors when inputting the clean and adversarial examples:

$$\mathcal{L}_{sns}(x, x'; \theta) = \sum_{l \in S} \sum_{m} ||F_{l}^{m}(x) - F_{l}^{m}(x')||_1,$$  \hspace{1cm} (7)

where S denotes the indices of selected layers and $\Omega_l$ denotes sensitive neuron set of layer $l$.

Given clean examples $x$ and adversarial examples $x'$, our SNS method minimizes the following loss:

$$\mathcal{L}_{cls}(x, y; \theta) + \lambda \cdot \mathcal{L}_{sns}(x, x'; \theta),$$ \hspace{1cm} (8)

where $\mathcal{L}_{cls}$ denotes the popular cross entropy loss for the classification task and $\mathcal{L}_{sns}$ represents the similarity of sensitive neurons’ behaviors in some specific layers for the dual pair $(x, x')$. Here, $\lambda$ is a hyper-parameter balancing the two loss terms.

![Figure 8: Model performance on CIFAR-10 with VGG-16 when applying top-k convolutional layers to train models using the loss in Equation (8).](image)

As there are numerous layers in the architecture of deep models, a question emerges: *do we need to select all layers when performing SNS training?* Since different hidden layers behave variously from each other [Zhang, Bengio, and Singer 2019], it seems necessary to come up with a strategy for choosing the desired hidden layers. As discussed before that sensitive neurons that are more close to predictions reveal more adversarial weaknesses towards adversarial attack meanwhile contain more high-level semantic information contributing to the model final decisions, thus we conduct experiments of training models with top-k hidden layers to figure out the choices for layer selection. As illustrated in Figure [8] using layers from `conv8` to `conv13` reaches the most adversarially robust model. Thus, the rest of paper follows this guidance.

Firstly, we try to test model robustness by using our proposed SNS on CIFAR-10 with VGG-16. As for the adversarial attack methods in this section, we choose Step-LL, MI-FGSM, FGSM and PGD. We first train model with the loss in Equation (8) using top-10% sensitive neurons, and from
Table 1 we can see that our model (SNS$^{cls}_{sen}$) outperforms NAT and EAT model for whitebox adversarial defense in all cases, which indicates the effectiveness of improving model robustness by constraining sensitive neurons.

Secondly, to examine the superiority of using sensitive neurons compared to all neurons for improving model robustness, we conduct a comparative experiment. As shown in Table 1 (SNS$^{cls}_{all}$), the model’s adversarial robustness decreases to some extent, which means that sensitive neurons are more critical for adversarial robustness, while constraining all neurons may somewhat lead to the drop of adversarial robustness. The reason might be that using vanilla neurons will introduce meaningless gradients, which is harmful for improving the model robustness.

Finally, in order to further improve model robustness, we introduce another version of sensitive neurons stabilizing loss, which adopts adversarial training loss instead of normal cross entropy loss:

$$L_{adv}(x,x';y;\theta) + \lambda L_{sns}(x,x';\theta), \tag{9}$$

where $L_{adv}$ denotes the adversarial training loss. The whole training process is demonstrated in Algorithm 1.

According to the result (SNS$^{cls}_{sen}$) in Table 1, our method outperforms all other comparative methods including PAT and ALP, indicating that SNS builds strong models towards adversarial examples. However, we notice that ALP and PAT decrease the model performance on clean examples drastically, meanwhile ALP also shows weak adversarial defense ability in most cases compared to PAT. These observations prove the importance of using sensitive neurons for improving model robustness compared to logits (ALP). Furthermore, stabilizing sensitive neurons also serves as a regularization term when used with adversarial training loss to alleviate clean example accuracy drops, since most adversarial training strategies build models with relatively low clean example accuracy. The corresponding experimental results on ImageNet with AlexNet are shown in Table 2.

Algorithm 1 Improve model robustness with Sensitive Neurons Stabilizing

Input: training set $\mathcal{D}$ with $N$ samples, Vanilla model $F_{\text{Vanilla}}$
Output: robust model $F_{\text{Robust}}$
Hyper-parameter: $\lambda$, batchsize $B$ and epoch $E$

1: Use PGD white-box attack to generate dual pair set $\mathcal{D} = \{(x_i, x'_i) \mid i = 1, \ldots, N\}$ and select Sensitive Neurons.
2: for $E$ training epochs do
3: for $[\frac{E}{B}]$ mini-batch numbers do
4: optimize the current model by $L_{adv}(x,x';y;\theta) + \lambda L_{sns}(x,x';\theta)$
5: end for
6: end for

Conclusion

In this paper, we attempt to interpret and improve adversarial robustness for deep models from a new perspective of neuron sensitivity which is measured by neuron behavior variation intensity against benign and adversarial examples. By thorough analyses, we first draw the close relationship between adversarial robustness and neuron sensitivities, as sensitive neurons make the most non-trivial contribution to model predictions in adversarial setting. Based on that conclusion, we further propose to improve model robustness against adversarial examples by sensitive neurons stabilizing which constrains the behaviors of sensitive neurons between benign and adversarial examples. Moreover, state-of-the-art adversarial training strategies also achieve strong robustness by reducing neuron sensitivities which in turn confirms the importance of sensitive neurons to model robustness in adversarial setting. Extensive experiments on various datasets demonstrate that our algorithm effectively achieves excellent results.

References

[Athalye, Carlini, and Wagner 2018] Athalye, A.; Carlini, N.; and Wagner, D. 2018. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In International Conference on Machine Learning.

[Bau et al. 2017] Bau, D.; Zhou, B.; Khosla, A.; Oliva, A.; and Torralba, A. 2017. Network dissection: Quantifying interpretability of deep visual representations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[Chen and Manning 2014] Chen, D., and Manning, C. 2014.
A fast and accurate dependency parser using neural networks. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP).

[Dong et al. 2017] Dong, Y.; Su, H.; Zhu, J.; and Bao, F. 2017. Towards interpretable deep neural networks by leveraging adversarial examples. CoRR abs/1708.05493.

[Dong et al. 2018] Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; and Li, J. 2018. Boosting adversarial attacks with momentum. In Proceedings of the IEEE conference on computer vision and pattern recognition.

[Dziugaite, Ghahramani, and Roy 2016] Dziugaite, G. K.; Ghahramani, Z.; and Roy, D. M. 2016. A study of the effect of JPG compression on adversarial images. CoRR abs/1608.00853.

[Engstrom et al. 2019] Engstrom, L.; Ilyas, A.; Santurkar, S.; Tsipras, D.; Tran, B.; and Madry, A. 2019. Learning perceptually-aligned representations via adversarial robustness. CoRR abs/1905.02175.

[Goodfellow, Shlens, and Szegedy 2015] Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and harnessing adversarial examples. In International Conference on Learning Representations.

[Goodfellow, Shlens, and Szegedy 2015] Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and harnessing adversarial examples. In International Conference on Learning Representations.

[Guo et al. 2018] Guo, C.; Rana, M.; Cissé, M.; and van der Maaten, L. 2018. Countering adversarial images using input transformations. In International Conference on Learning Representations.

[He et al. 2016] He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016.

[Ilyas et al. 2019] Ilyas, A.; Santurkar, S.; Tsipras, D.; Engstrom, L.; Tran, B.; and Madry, A. 2019. Adversarial examples are not bugs, they are features. CoRR abs/1905.02175.

[Kannan, Kurakin, and Goodfellow 2018] Kannan, H.; Kurakin, A.; and Goodfellow, I. J. 2018. Adversarial logit pairing. CoRR abs/1803.06373.

[Krizhevsky and Hinton 2009] Krizhevsky, A., and Hinton, G. 2009. Learning multiple layers of features from tiny images. Technical report, Citeseer.

[Krizhevsky, Sutskever, and Hinton 2012] Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems.

[Kurakin, Goodfellow, and Bengio 2017a] Kurakin, A.; Goodfellow, I. J.; and Bengio, S. 2017a. Adversarial examples in the physical world. In International Conference on Learning Representations.

[Kurakin, Goodfellow, and Bengio 2017b] Kurakin, A.; Goodfellow, I. J.; and Bengio, S. 2017b. Adversarial machine learning at scale. In International Conference on Learning Representations.

[Liao et al. 2018] Liao, F.; Liang, M.; Dong, Y.; Pang, T.; Hu, X.; and Zhu, J. 2018. Defense against adversarial attacks using high-level representation guided denoiser. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[Liu et al. 2019] Liu, A.; Liu, X.; Fan, J.; Ma, Y.; Zhang, A.; Xie, H.; and Tao, D. 2019. Perceptual-sensitive gan for generating adversarial patches. In 33rd AAAI Conference on Artificial Intelligence.

[Lu, Issaranon, and Forsyth 2017] Lu, J.; Issaranon, T.; and Forsyth, D. A. 2017. Safetynet: Detecting and rejecting adversarial examples robustly. In IEEE International Conference on Computer Vision.

[Madry et al. 2018] Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2018. Towards deep learning models resistant to adversarial attacks. In International Conference on Learning Representations.

[Mohamed, Dahl, and Hinton 2012] Mohamed, A.; Dahl, G. E.; and Hinton, G. E. 2012. Acoustic modeling using deep belief networks. IEEE Trans. Audio, Speech & Language Processing 20(1).

[Paperot et al. 2016] Paperot, N.; McDaniel, P. D.; Wu, X.; Jha, S.; and Swami, A. 2016. Distillation as a defense to adversarial perturbations against deep neural networks. In IEEE Symposium on Security and Privacy.

[Russakovsky et al. 2015] Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M. S.; Berg, A. C.; and Li, F. 2015. Imagenet large scale visual recognition challenge. International Journal of Computer Vision 115(3).

[Shaham, Yamada, and Negahban 2018] Shaham, U.; Yamada, Y.; and Negahban, S. 2018. Understanding adversarial training: Increasing local stability of supervised models through robust optimization. Neurocomputing 307:195–204.

[Simonyan and Zisserman 2015] Simonyan, K., and Zisserman, A. 2015. Very deep convolutional networks for large-scale image recognition. In International Conference on Learning Representations.

[Szegedy et al. 2014] Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I. J.; and Fergus, R. 2014. Intriguing properties of neural networks. In International Conference on Learning Representations.

[Tramèr et al. 2018] Tramèr, F.; Kurakin, A.; Papernot, N.; Goodfellow, I. J.; Boneh, D.; and McDaniel, P. D. 2018. Ensemble adversarial training: Attacks and defenses. In International Conference on Learning Representations.

[Tsipras et al. 2019] Tsipras, D.; Santurkar, S.; Engstrom, L.; Tran, B.; and Madry, A. 2019. Robustness may be at odds with accuracy. In International Conference on Learning Representations.

[Xie et al. 2018] Xie, C.; Wang, J.; Zhang, Z.; Ren, Z.; and Yuille, A. L. 2018. Mitigating adversarial effects through randomization. In International Conference on Learning Representations.

[Xu et al. 2012] Xu, H.; and Mannor, S. 2012. Robustness and generalization. Machine learning 86(3).

[Xu et al. 2019] Xu, K.; Liu, S.; Zhang, G.; Sun, M.; Zhao, P.; Fan, Q.; Gan, C.; and Lin, X. 2019. Interpreting adversarial examples by activation promotion and suppression. CoRR abs/1904.02057.
Appendix

Empirical Analysis Details

In this section, we will give some details of our methods and measurements used in the empirical analysis section.

Similarity Measurements for Sensitive Neuron Sets

As mentioned in main body part, we adopt MI-FGSM method to generate targeted adversarial example set $D'$ for a specific target class $y$ on CIFAR-10 with VGG-16, where $y \in \{1, \ldots, Y\}$, it denotes target label. Then, we respectively select the sensitive neurons $\Omega^y_l$ for $D'$, where $l \in \{1, \ldots, L\}$. To measure the similarity of these sets in one specific layer $l$, we utilized two metrics.

The first one is average pair Jaccard index which is computed as follow:

$$J_{l} = \frac{\sum_{i \neq j} J(\Omega^y_i, \Omega^y_j)}{\sum_{i,j} J(\Omega^y_i, \Omega^y_j)}.$$  

It is used to represent the average case of the overlap between two different sets in layer $l$.

Meanwhile, we simultaneously measure the common similarity of all sensitive neuron sets $\Omega^y_l$ in layer $l$, where $y \in \{1, \ldots, Y\}$. Specifically, we adopt the formula below:

$$J(\Omega^1_l, \ldots, \Omega^Y_l) = \frac{|\bigcap_{y=1}^Y \Omega^y_l|}{\bigcup_{y=1}^Y \Omega^y_l|},$$

which represents the common overlap of these sets, and we name it as total Jaccard index.

Weighted Voting Method

In order to select the Important Neurons set $\Gamma^y$ to one adversarial example set $D'$ for a specific target class $y$, we apply a weighted voting method $\rho$. Specifically, an Important Neurons $\Pi^y(x_i)$ contains $k_3$ neurons which are ranked in descending order by sensitivity. To emphasize their different importances, we give them different votes from $k_1$ to 1. After the voting process, votes for each neuron will be counted and the top-$k_2$ of them will be selected. In our experiment, we choose $k_1 = k_2 = 20$.

More Experiment Results

In this section, we will give more experiment results.

Sensitive Neurons Reveal Vulnerable Directions to Adversarial Weakness

In the empirical analysis part, we have demonstrated the finding that sensitive neurons in different target set vary a lot in the top layers, but have high similarities in the bottom layers using VGG-16 on CIFAR-10. To further confirm it, we adopt experiment on ImageNet with AlexNet. As there are excessive classes in ImageNet, we randomly select 10 of them and apply MI-FGSM targeted attack to these target class $y$ to generate adversarial example dataset $D'$ respectively and then obtain the sensitive neuron sets $\Omega^y_l$ of each classes, where $l \in \{1, \ldots, L\}$. Total Jaccard index and average pair Jaccard index of sensitive neuron set are computed and the results are shown in Figure 9 which double confirms the conclusion.

Moreover, we list the detailed results of sensitive neurons on 10 different targeted adversarial example sets by layers within VGG-16 on CIFAR-10. Table 3 and Table 4 list the result of several specific layers: conv1, conv4, conv7, conv11, conv13 and fc. All the results are ranked in descending order by sensitivity.
Adversarial Training Improves Model Robustness by Reducing Neuron Sensitivities

In this section, we demonstrate more results about the contributions of sensitive neurons to adversarial training strategies. As shown in Figure [10][11] the mean sensitivity values of sensitive neurons on CIFAR-10 with VGG-16 and ImageNet with AlexNet using PAT are lower than that of Vanilla model. Then, the result of ResNet-18 on CIFAR-10 is shown in Figure [12]. After that, we also show the sensitivities of all neurons on different layers of AlexNet on ImageNet in Figure [13]. Above results confirm the evidence that adversarial training improves model robustness by embedding neuron insensitivity.

Sensitive Neurons contribute Most to Adversarial Misclassification

In this section, we further demonstrate the result of visualizing receptive fields of sensitive neurons and vanilla ones in two more models with same process mentioned before. The comparison of visualization result of sensitive neurons and vanilla ones on CIFAR-10 with VGG-16 is shown in

Figure [14] and [15] respectively. Meanwhile, Figure [16] and [17] demonstrate the result on ImageNet with AlexNet. From these visualization results, same conclusions can be drawn as stated before that sensitive neurons contribute most to adversarial misclassification.
Adversarial Attack Algorithms

In this section, the implementation details of adversarial attack algorithms we used are introduced.

**FGSM.** For FGSM attack, we set the parameters as $\epsilon = 8/255$.

**Step-LL.** For Step-LL attack, we set perturbation to $\epsilon = 8/255$.

**MI-FGSM.** For MI-FGSM attack, we set the parameters as $\epsilon = 8/255$, step size $\alpha = \epsilon/10$ and iteration $k = 10$.

**PGD.** For PGD attack, we set perturbation to $\epsilon = 8/255$, step size $\alpha = \epsilon/10$ and iteration $k = 10$.

Adversarial Defensive Methods

In this section, the implementation details of SNS and other defensive methods are given.

**SNS.** PGD attack is used for generating adversarial example set from training set, based on which sensitive neurons are selected. Then we train models with different hyper-parameters on different datasets. On CIFAR-10 with VGG-16, we use the sensitive neurons from $\text{conv}8$ to $\text{conv}13$ with $\lambda = 5.0$ in the training set. On ImageNet with AlexNet, we use the sensitive neurons from $\text{conv}3$ to $\text{conv}5$ with $\lambda = 6.0$ in the training set.

**NAT.** Step-LL attack on current training model is used for adversarial examples generating with $\epsilon$ obtained by absolute value of normal distribution $N(0, 8)$ and truncated to $[0, 16]$. Meanwhile, the coefficient ratio adopted for clean and adversarial examples during the training phase are 1 to 1.

**EAT.** Experiment settings for EAT are the same for NAT except for the target models for adversarial example generating. On CIFAR-10, the target models are Inception-v2, ResNet-18 and the current training model.

**PAT.** PGD attack on current training model is used for adversarial examples generating with $\epsilon = 8, k \in \{10\}$, $\alpha = 0.8$. The clean and adversarial examples are mixed in the ratio of 1 to 1 during the training phase.

**ALP.** The loss terms of ALP consist of two main parts: adversarial training loss whose setting is consistent with PAT and logit pairing term which is implemented with $L_2$ loss. The ratio of these two terms during training phase are 2 to 1, namely the coefficient $\lambda$ of logit pairing term is 0.5.

---

Figure 13: Neuron sensitivity of all neurons on different layers ($\text{conv}2$, $\text{conv}3$, $\text{conv}4$ and $\text{conv}5$ from subfigure (a) to (d)) with Vanilla and PAT using AlexNet on ImageNet.
Figure 14: Receptive fields of sensitive neurons on benign (top line) and adversarial examples (bottom line). (a), (b) and (c) represent sensitive neuron 6, 53 and 74 in the pool2 layer of VGG-16 on CIFAR-10, respectively.

Figure 15: Receptive fields of vanilla neurons on benign (top line) and adversarial examples (bottom line). (a), (b) and (c) represent vanilla neuron 5, 73 and 45 in the pool2 layer of VGG-16 on CIFAR-10, respectively.

Figure 16: Receptive fields of sensitive neurons on benign (top line) and adversarial examples (bottom line). (a), (b) and (c) represent sensitive neuron 112, 89 and 136 in the pool3 layer of AlexNet on ImageNet, respectively.

Figure 17: Receptive fields of vanilla neurons on benign (top line) and adversarial examples (bottom line). (a), (b) and (c) represent vanilla neuron 116, 32 and 207 in the pool3 layer of AlexNet on ImageNet, respectively.
Table 3: Sensitive neurons on 10 targeted adversarial example sets in conv1, conv4, conv7 and conv11 layer of VGG-16 on CIFAR-10.

| Target Label | conv1       | conv4       | conv7       | conv11      |
|--------------|-------------|-------------|-------------|-------------|
| 0            | 52, 10, 14, | 52, 10, 14, | 52, 10, 14, | 52, 10, 14, |
|              | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  |
| 1            | 10, 52, 14, | 10, 52, 14, | 10, 52, 14, | 10, 52, 14, |
|              | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  |
| 2            | 10, 52, 14, | 10, 52, 14, | 10, 52, 14, | 10, 52, 14, |
|              | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  |
| 3            | 10, 52, 14, | 10, 52, 14, | 10, 52, 14, | 10, 52, 14, |
|              | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  |
| 4            | 10, 52, 14, | 10, 52, 14, | 10, 52, 14, | 10, 52, 14, |
|              | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  |
| 5            | 52, 10, 14, | 52, 10, 14, | 52, 10, 14, | 52, 10, 14, |
|              | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  |
| 6            | 52, 10, 14, | 52, 10, 14, | 52, 10, 14, | 52, 10, 14, |
|              | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  |
| 7            | 52, 10, 14, | 52, 10, 14, | 52, 10, 14, | 52, 10, 14, |
|              | 55, 15, 25  | 55, 15, 25  | 55, 15, 25  | 55, 15, 25  |
| 8            | 52, 10, 14, | 52, 10, 14, | 52, 10, 14, | 52, 10, 14, |
|              | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  |
| 9            | 10, 52, 14, | 10, 52, 14, | 10, 52, 14, | 10, 52, 14, |
|              | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  | 55, 25, 15  |
| Target Label | conv13                                                                 | fc                        |
|--------------|------------------------------------------------------------------------|---------------------------|
| 0            | 387, 220, 414, 212, 362, 8, 2, 511, 431, 440, 324, 33, 291, 54, 461, 9, 478, 67, 280, 60, 374, 143, 480, 188, 462, 364, 219, 162, 381, 226, 114, 370, 46, 16, 389, 256, 115, 140, 287, 179, 124, 176, 506, 152, 371, 353, 167, 84, 459, 209 | 0, 1, 6, 8, 9, 5, 7, 3, 4, 2 |
| 1            | 84, 462, 256, 431, 105, 283, 341, 410, 79, 108, 465, 371, 403, 138, 147, 45, 125, 124, 123, 184, 353, 175, 136, 445, 372, 328, 225, 385, 257, 221, 85, 327, 485, 440, 274, 167, 407, 305, 340, 347, 483, 325, 212, 383, 392, 486, 460, 240, 334, 269 | 1, 8, 4, 9, 7, 6, 2, 5, 3, 0 |
| 2            | 389, 334, 462, 490, 420, 442, 466, 65, 344, 257, 468, 145, 72, 407, 265, 162, 318, 288, 267, 146, 347, 511, 122, 351, 179, 88, 365, 161, 186, 460, 371, 383, 409, 359, 82, 2, 25, 435, 54, 185, 445, 78, 17, 489, 294, 51, 361, 222, 493, 190, 68 | 2, 8, 1, 9, 0, 3, 7, 6, 4, 5 |
| 3            | 462, 140, 64, 36, 161, 485, 209, 392, 327, 102, 320, 162, 55, 368, 191, 423, 409, 84, 309, 391, 330, 435, 7, 307, 467, 486, 426, 175, 492, 493, 2, 484, 427, 79, 371, 34, 201, 389, 344, 411, 263, 498, 5, 273, 288, 85, 185, 359, 10, 361 | 3, 1, 8, 9, 4, 0, 7, 6, 5, 2 |
| 4            | 179, 431, 372, 315, 340, 385, 419, 149, 364, 221, 409, 263, 462, 487, 485, 496, 353, 219, 129, 448, 490, 185, 162, 509, 302, 79, 274, 327, 138, 156, 481, 278, 44, 277, 203, 139, 145, 404, 362, 15, 394, 399, 428, 208, 479, 95, 239, 288, 77 | 4, 1, 9, 8, 7, 2, 3, 6, 5, 0 |
| 5            | 384, 84, 466, 291, 335, 371, 33, 353, 394, 191, 490, 354, 209, 309, 123, 359, 67, 203, 101, 186, 72, 269, 187, 460, 179, 286, 134, 483, 221, 279, 34, 362, 372, 333, 481, 387, 283, 325, 392, 61, 440, 287, 272, 85, 25, 462, 97, 267, 364, 504 | 5, 1, 6, 4, 9, 2, 8, 0, 7, 3 |
| 6            | 340, 283, 372, 188, 460, 15, 108, 288, 155, 487, 107, 262, 496, 334, 182, 77, 500, 332, 353, 162, 282, 149, 378, 485, 185, 17, 201, 389, 291, 498, 393, 407, 431, 511, 399, 482, 428, 417, 74, 104, 440, 428, 468, 272, 307, 381, 219, 36, 257, 256, 38 | 6, 1, 9, 7, 0, 8, 4, 2, 3, 5 |
| 7            | 283, 155, 274, 333, 407, 361, 451, 374, 291, 421, 143, 201, 33, 493, 340, 378, 307, 431, 139, 263, 326, 107, 371, 267, 348, 186, 269, 262, 158, 36, 5, 248, 118, 72, 51, 20, 65, 223, 179, 100, 59, 209, 447, 286, 299, 394, 434, 77, 206, 92, 204 | 7, 8, 1, 6, 9, 0, 4, 2, 5, 3 |
| 8            | 107, 145, 131, 26, 424, 374, 354, 261, 101, 419, 361, 499, 154, 151, 189, 362, 385, 340, 373, 67, 49, 2, 411, 170, 347, 59, 335, 5, 70, 283, 110, 497, 229, 414, 343, 14, 380, 119, 190, 232, 417, 383, 351, 398, 384, 55, 294, 392, 370, 271, 446 | 8, 4, 5, 1, 7, 9, 2, 6, 3, 0 |
| 9            | 281, 353, 164, 43, 127, 295, 509, 335, 340, 372, 137, 463, 203, 33, 175, 325, 179, 147, 138, 459, 454, 371, 91, 28, 490, 126, 272, 166, 208, 493, 381, 220, 141, 468, 196, 59, 411, 249, 274, 61, 431, 460, 266, 55, 256, 174, 369, 419, 417, 511 | 9, 4, 1, 6, 2, 5, 8, 7, 3, 0 |