# Absum: Simple Regularization Method for Reducing Structural Sensitivity of Convolutional Neural Networks

**Sekitoshi Kanai,**[1,2] **Yasutoshi Ida,**[1,3] **Yasuhiro Fujiwara,**[4] **Masanori Yamada,**[5] **Shuichi Adachi**[2]
[1]NTT Software Innovation Center, [2]Keio University, [3]Kyoto University
[4]NTT Communication Science Laboratories, [5]NTT Secure Platform Laboratories
sekitoshi.kanai.fu@hco.ntt.co.jp, yasutoshi.ida@ieee.org,
{yasuhiro.fujiwara.kh, masanori.yamada.cm}@hco.ntt.co.jp, adachi@appi.keio.ac.jp

## Abstract

We propose Absum, which is a regularization method for improving adversarial robustness of convolutional neural networks (CNNs). Although CNNs can accurately recognize images, recent studies have shown that the convolution operations in CNNs commonly have structural sensitivity to specific noise composed of Fourier basis functions. By exploiting this sensitivity, they proposed a simple black-box adversarial attack: Single Fourier attack. To reduce structural sensitivity, we can use regularization of convolution filter weights since the sensitivity of linear transform can be assessed by the norm of the weights. However, standard regularization methods can prevent minimization of the loss function because they impose a tight constraint for obtaining high robustness. To solve this problem, Absum imposes a loose constraint; it penalizes the absolute values of the summation of the parameters in the convolution layers. Absum can improve robustness against single Fourier attack while being as simple and efficient as standard regularization methods (e.g., weight decay and $L_1$ regularization). Our experiments demonstrate that Absum improves robustness against single Fourier attack more than standard regularization methods. Furthermore, we reveal that robust CNNs with Absum are more robust against transferred attacks due to decreasing the common sensitivity and against high-frequency noise than standard regularization methods. We also reveal that Absum can improve robustness against gradient-based attacks (projected gradient descent) when used with adversarial training.

## Introduction

Deep neural networks have achieved great success in many applications, e.g., image recognition (He et al. 2016) and machine translation (Vaswani et al. 2017). Specifically, CNNs and rectified linear units (ReLUs) have resulted in breakthroughs in image recognition (LeCun et al. 1989; Nair and Hinton 2010) and are de facto standards for image recognition and other applications (He et al. 2016; Radford, Metz, and Chintala 2016). Though CNNs can classify image data as accurately as humans, they are sensitive to small perturbations of inputs, i.e., injecting imperceptible perturbations can make deep models misclassify image data. Such attacks are called adversarial attacks and the perturbed inputs are called adversarial examples (Szegedy et al. 2013).

We can roughly divide adversarial attacks into two types; white-box attacks, which use the information of target models (Goodfellow, Shlens, and Szegedy 2014; Madry et al. 2018; Moosavi-Dezfooli, Fawzi, and Frossard 2016), and black-box attacks, which do not require the information of target models (Papernot, McDaniel, and Goodfellow 2016; Chen et al. 2017; Papernot et al. 2017). Black-box attacks, rather than white-box attacks, can threaten online deep-learning services since it is difficult to access the target models in online deep-learning applications (Papernot et al. 2017; Yuan et al. 2019).

Most black-box attacks are transferred attacks, which are generated as white-box attacks for substitute models instead of the target model (Papernot, McDaniel, and Goodfellow 2016). This implies that deep models have common sensitivity against specific perturbations. In fact, Tsuzuku and Sato (2019) have recently shown that CNNs have the structural sensitivity from the perspective that convolution can be regarded as the product of the circulant matrix and proposed single Fourier attack (SFA).[1] Fourier basis functions create singular vectors of circulant matrices, and SFA uses these singular vectors since the dominant singular vector can be the worst noise for a matrix-vector product. Although SFA is a very simple attack composed of a single-frequency component, it is universal adversarial perturbations for CNNs, i.e., it can decrease the classification accuracy of various CNN-based models without using the information about the model parameters and without depending on input images. To the best of our knowledge, a defense method against SFA has not been proposed. Therefore, such a method is necessary.

To defend CNNs against SFA, we first reveal that the spectral norm constraint (Sedghi, Gupta, and Long 2019) (hereinafter, we call it SNC) can reduce the structural sensitivity. While SNC was proposed to improve generalization performance, it can improve robustness in the Fourier domain since singular values of convolution layers correspond to the magnitude of the frequency response. However, SNC is not so practical since it requires high computational cost to compute the spectral norm (the largest singular value).



[1]Yin et al. (2019) concurrently proposed the same attack.

We then develop *Absum*; an efficient regularization method for reducing the structural sensitivity of CNNs. Instead of the spectral norm, we use the induced $\infty$-norm ($L_\infty$ operator norm) since it is the upper bound of the spectral norm for convolution. However, a constraint of the induced $\infty$-norm, which is equivalent to $L_1$ regularization, requires a tight constraint for robustness, which prevents minimization of the loss function. This is because the induced $\infty$-norm is a conservative measure; it handles the effects of negative inputs even though inputs always have positive values after ReLU activations. To improve robustness without preventing the loss minimization, Absum relaxes the induced $\infty$-norm by penalizing the absolute values of the summations of weights instead of elements on the basis that input vectors always have positive elements. Absum is as simple as standard regularization methods such as weight decay, but it can reduce sensitivity to SFA. We provide the proximal operator to minimize loss functions with Absum.

Image recognition experiments on MNIST, Fashion-MNIST (FMNIST), CIFAR10, CIFAR100, SVHN, and ImageNet demonstrate that Absum and SNC outperform $L_1$ and $L_2$ regularization methods in terms of improving robustness against SFA, and the computation time of Absum is about one-tenth that of SNC. In the additional empirical evaluation, we reveal that robust CNNs against SFA can be robust against transferred attacks by using white-box attacks (projected gradient descent: PGD (Kurakin, Goodfellow, and Bengio 2016; Madry et al. 2018)). This implies that sensitivity to SFA is one of the causes of the transferability of adversarial attacks. As a further investigation, we reveal that adversarial perturbations for CNNs trained with Absum and SNC have little high-frequency components, i.e., these CNNs are robust against high-frequency noise. Furthermore, our experiments show that Absum is effective against PGD when using adversarial training.

The following are main contributions of this paper:

- We show that SNC improves robustness against SFA. SNC was proposed to improve generalization performance, but effectiveness in robustness against SFA had not been evaluated.
- We propose Absum and its proximal operator. Absum improves robustness against SFA as well as SNC while its computational cost is lower than that of SNC.
- In the futher empirical evaluation, Absum and SNC can also improve robustness against other black-box attacks (transferred attacks and High-Frequency attacks (Wang et al. 2019)). In addition, Absum can improve robustness against PGD when used with adversarial training.

# Preliminaries

## CNNs, ReLUs and Circulant Matrix

In this section, we outline CNNs, ReLUs, and a circulant matrix for convolution operation. Let $\boldsymbol{X} \in \boldsymbol{R}^{n\times n}$ be an input map, $\boldsymbol{Y} \in \boldsymbol{R}^{n\times n}$ be an output map, and $\boldsymbol{K} \in \boldsymbol{R}^{n\times n}$ be a filter matrix such that $\boldsymbol{K} = [\boldsymbol{k}_1, \boldsymbol{k}_2, \dots, \boldsymbol{k}_n]^T$, where $\boldsymbol{k}_i = [k_{i,1}, k_{i,2} \dots, k_{i,n}]^T \in \boldsymbol{R}^n$. The output of the convolution operation $\boldsymbol{Y} = \boldsymbol{K} * \boldsymbol{X}$ becomes

$$Y_{l,m} = \sum_{p=1}^{n}\sum_{q=1}^{n} k_{p,q} X_{l+p-1,m+q-1}. \quad (1)$$

Note that when the filter size is $h \times h$ and $h < n$, we can embed it in the $n \times n$ matrix $\boldsymbol{K}$ by padding with zeros (Sedghi, Gupta, and Long 2019). After the convolution, we usually use ReLU activations as the following function:

$$ReLU(x) = \max(x, 0). \quad (2)$$

Typical model architectures use a combination of convolution and ReLU. For example, a standard block of ResNet (He et al. 2016) is composed as

$$h(\boldsymbol{X}) = ReLU(\boldsymbol{X} + BN(\boldsymbol{K}^{(2)} * ReLU(BN(\boldsymbol{K}^{(1)} * \boldsymbol{X})))), \quad (3)$$

where $BN$ is batch normalization (Ioffe and Szegedy 2015).

Since SFA and Absum are based on a circulant matrix for convolution operation, we show that the convolution can be expressed as a product of a vector and doubly block circulant matrix. Let $\boldsymbol{x} = \mathrm{vec}(\boldsymbol{X})$ and $\boldsymbol{y} = \mathrm{vec}(\boldsymbol{Y})$ be vectors obtained by stacking the columns of $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively. Convolution $\boldsymbol{K} * \boldsymbol{X}$ can be written as

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}, \quad (4)$$

where $\boldsymbol{A} \in \boldsymbol{R}^{n^2 \times n^2}$ is the following matrix:

$$\boldsymbol{A} = \begin{bmatrix} \mathrm{c}(\boldsymbol{k}_1) & \mathrm{c}(\boldsymbol{k}_2) & \dots & \mathrm{c}(\boldsymbol{k}_n) \\ \mathrm{c}(\boldsymbol{k}_n) & \mathrm{c}(\boldsymbol{k}_1) & \dots & \mathrm{c}(\boldsymbol{k}_{n-1}) \\ \vdots & & \ddots & \vdots \\ \mathrm{c}(\boldsymbol{k}_2) & \mathrm{c}(\boldsymbol{k}_3) & \dots & \mathrm{c}(\boldsymbol{k}_1) \end{bmatrix}, \mathrm{c}(\boldsymbol{k}_i) = \begin{bmatrix} k_{i,1}, & k_{i,2}, & \dots, & k_{i,n} \\ k_{i,n}, & k_{i,1}, & \dots, & k_{i,n-1} \\ \vdots & & \ddots & \vdots \\ k_{i,2}, & k_{i,3}, & \dots, & k_{i,1} \end{bmatrix}. \quad (5)$$

The coefficients $k_{i,j}$ are cyclically shifted in $\mathrm{c}(\boldsymbol{k}_i) \in \boldsymbol{R}^{n\times n}$, and block matrices $\mathrm{c}(\boldsymbol{k}_i)$ are cyclically shifted in $\boldsymbol{A}$. Therefore, $\boldsymbol{A}$ is called a doubly block circulant matrix.

## Single Fourier Attack

As mentioned above, convolution can be written by a doubly block circulant matrix. Such matrices always have eigenvectors $\boldsymbol{Q} = \frac{1}{n}\boldsymbol{F} \otimes \boldsymbol{F}$, where elements of $\boldsymbol{F}$ are composed of the Fourier basis $F_{l,m} = \exp(-j\frac{2\pi}{n}lm)$, where $j = \sqrt{-1}$ (Jain 1989; Sedghi, Gupta, and Long 2019; Tsuzuku and Sato 2019), and singular vectors are also composed of $\boldsymbol{F}\otimes\boldsymbol{F}$ even if we stack convolution layers (Tsuzuku and Sato 2019; Karner, Schneid, and Ueberhuber 2003). From these characteristics, Tsuzuku and Sato (2019) proposed SFA. The perturbed input image $\hat{\boldsymbol{X}}$ by SFA is

$$\hat{\boldsymbol{X}} = \boldsymbol{X} + \varepsilon((1+j)(\boldsymbol{F})_l \otimes (\boldsymbol{F})_m + (1-j)(\boldsymbol{F})_{n-l} \otimes (\boldsymbol{F})_{n-m}), \quad (6)$$

where $(\boldsymbol{F})_l \in \boldsymbol{R}^n$ is the $l$-th column vector of $\boldsymbol{F}$, $\boldsymbol{X}$ is an input image, and $\varepsilon$ is magnitude of the attack. SFA is composed of $(\boldsymbol{F})_l \otimes (\boldsymbol{F})_m$ and its complex conjugate $(\boldsymbol{F})_{n-l} \otimes (\boldsymbol{F})_{n-m}$ to create a perturbation that has real values since inputs of CNNs are assumed to be real values. The $l$ and $m$ are hyperparameters such that $l = 0, 1, \dots, n-1, m = 0, 1, \dots, n-1$. Figure 1 shows examples of CIFAR10 perturbed by SFA. We can see that $(l, m)$ determines a space-frequency of the noise. Note that stacked convolution layers without activation functions (e.g., $\boldsymbol{A}^{(2)}\boldsymbol{A}^{(1)}\boldsymbol{x}$) also have singular vectors composed of Fourier basis functions. Even though we use nonlinear activation functions, many model architectures (e.g., WideResNet, DenseNet-BC, and GoogLeNet) are sensitive to SFA (Tsuzuku and Sato 2019).

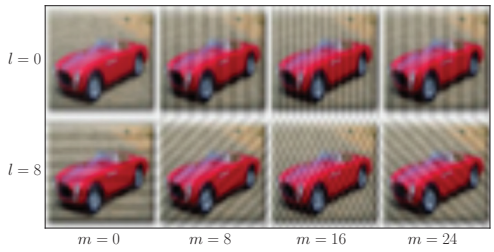



Figure 1: Examples perturbed by SFA.

## Vulnerability of CNNs in Frequency Domain

Sensitivity to SFA can be regarded as sensitivity to a single-frequency noise (Yin et al. 2019). To understand the vulnerability of CNNs, several studies focused on sensitivity of CNNs in the frequency domain (Yin et al. 2019; Wang et al. 2019; Das et al. 2018; Liu et al. 2019). These studies point out that sensitivity to high-frequency components in images is one of the causes of adversarial attacks since human visual systems are not sensitive to high-frequency components unlike CNNs. In fact, several studies show that CNNs are sensitive to high-frequency noise (Jo and Bengio 2017; Wang et al. 2019; Yin et al. 2019; Das et al. 2018). Jo and Bengio (2017) and Wang et al. (2019) show that CNNs misclassify images processed by low-pass filters and Wang et al. (2019) call this a High-Frequency attack, which is a simple black-box adversarial attack. There is a hypothesis that robust CNNs against high-frequency noise are also robust against adversarial attacks (Wang et al. 2019; Yin et al. 2019). Note that Wang et al. (2019) claimed that sensitivity in the high-frequency domain contributes to high performance on clean data; thus, there is a trade-off.

## Related Work

Adversarial attacks can be transferred to other models and transferred white-box attacks become adversarial black-box attacks (Papernot et al. 2017). These attacks can be defended against by adversarial training, which is a promising defense method (Papernot et al. 2017; Madry et al. 2018). However, the computational cost of adversarial training is larger than naive training. Note that Absum can be used with adversarial training. Several studies proposed black-box attacks using queries to ask the target model about predicted labels of given data, but these attacks might still be impractical since they require many queries (Chen et al. 2017; Brendel, Rauber, and Bethge 2018; Ilyas et al. 2018). On the other hand, SFA only uses the information that the target model is composed of CNNs and is more practical.

Our method simply penalizes parameters in a similar manner compared to standard regularization methods. As standard regularization methods, $L_2$ regularization (weight decay) is commonly used for improving generalization performance due to its simplicity. $L_1$ regularization is also used since it induces sparsity (Goodfellow, Bengio, and Courville 2016). In addition, spectral norm (induced 2-norm) regularization can also improve generalization performance (Yoshida and Miyato 2017; Sedghi, Gupta, and Long 2019).

# Defense Methods against SFA

In this section, we first show that SNC can improve robustness against SFA. Since SNC has a large time complexity, we next discuss whether standard regularizations can be alternatives. Finally, we discuss Absum and its proximal operator, which is an efficient defense method against SFA.

## Spectral Norm Constraint

SFA is based on the following properties of linear transform:

$$\sigma(\boldsymbol{A}) = \max_{||\boldsymbol{x}||_2=1} ||\boldsymbol{A}\boldsymbol{x}||_2,\ \boldsymbol{v} = \arg\max_{||\boldsymbol{x}||_2=1} ||\boldsymbol{A}\boldsymbol{x}||_2, \quad (7)$$

where $\sigma$ is the largest singular value (spectral norm or induced 2-norm), and $\boldsymbol{v}$ is the right singular vector corresponding to $\sigma$. Equation (7) shows that the singular vector can be the worst noise for linear transform, and SFA uses the singular vectors for convolutional layers. Since the spectral norm determines the impact of SFA, we can reduce sensitivity to SFA by constraining the spectral norm. The constraint of the spectral norm for CNNs (i.e., SNC) (Sedghi, Gupta, and Long 2019; Gouk et al. 2018) was proposed in the context of improving generalization performance. SNC clips $\sigma$ if it exceeds a preset threshold; thus, it can directly control sensitivity to a single-frequency perturbation. However, the constraints of the exact spectral norm[2] of $\boldsymbol{A}$ incurs large computation cost; the $O(n^2c^2(c + \log(n)))$ time for each convolution when input size is $n \times n$, and the numbers of input and output channels are $c$ even if we use the efficient spectral norm constraints (Sedghi, Gupta, and Long 2019). SNC can be infeasible when the size of inputs increases.

## Standard Regularizations fail to Defend

Instead of using the spectral norm, we can assess the effect of the perturbation for linear transform by using

$$\max_{||\boldsymbol{x}||_\infty=1} ||\boldsymbol{A}\boldsymbol{x}||_\infty. \quad (8)$$

Equation (8) is the induced $\infty$-norm $||\boldsymbol{A}||_\infty$, and we have $||\boldsymbol{A}||_2 \leq ||\boldsymbol{A}||_\infty$ for convolution (it is proved in appendix). This norm is calculated as:

$$||\boldsymbol{A}||_\infty = \max_l \sum_m |A_{l,m}|. \quad (9)$$

Substituting eq. (5) for eq. (9), we have

$$\max_l \sum_m |A_{l,m}| = \sum_m \sum_l |k_{l,m}|. \quad (10)$$

Thus, the penalty of the induced $\infty$-norm can be $L_1$ regularization (Gouk et al. 2018). Therefore, $L_1$ regularization can improve robustness. However, the induced $\infty$-norm is a conservative measure of robustness (Szegedy et al. 2013); the highly weighted $L_1$ regularization for robustness can prevent minimization of the loss function. Figure 2 shows the test accuracy of models, which is trained with $L_1$ regularization, on data perturbed by SFA against the regularization weight $\lambda$. In this figure, the robust accuracy against SFA increases along with the regularization weight, i.e., the robustness increases according to the regularization weight. However, the accuracy significantly decreases when the weight

[2]Spectral norm regularization of (Yoshida and Miyato 2017) is more efficient but uses heuristics and its spectral norm is often quite different from that of $\boldsymbol{A}$ (Sedghi, Gupta, and Long 2019; Gouk et al. 2018).

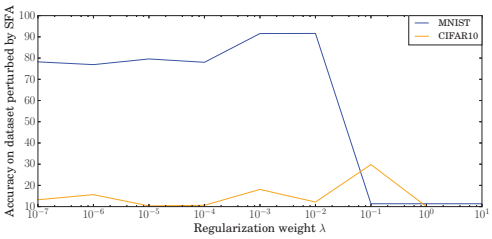



Figure 2: Accuracy of models trained with $L_1$ regularization on test dataset perturbed by SFA vs regularization weight. $l, m$ of SFA are tuned to minimize accuracy for each $\lambda$.

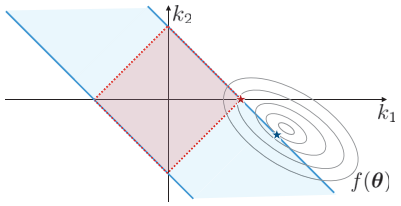



Figure 3: Comparison of search spaces of Absum: $|k_1 + k_2|$ (blue) and $L_1$ regularization: $|k1| + |k_2|$ (red) where $f(\boldsymbol{\theta})$ is loss function. We have $\{\boldsymbol{k}|\sum_i |k_i| \leq c\} \subseteq \{\boldsymbol{k}| \, |\sum_i k_i| \leq c\}$ for any constant $c \geq 0$ from triangle inequality.

exceeds a certain point. This is because training with high weighted $L_1$ regularization does not have sufficient search space to minimize the loss function. Note that weight decay can also penalize the spectral norm (in appendix) and imposes tight regularization, as discussed in the experiments section. Therefore, we need a weak regularization method such that models become both highly robust and accurate.

## Absum: Simple and Weak Regularization

To develop a weak regularization method, we reconsider the optimization problem of eq. (8). The maximum point (eq. (9)) is achieved by $x_m = \text{sign}(A_{l',m})$, where $l' = \arg\max_l \sum_m |A_{l,m}|$, i.e., $x_m = 1$ if $A_{l',m} > 0$ and $x_m = -1$ if $A_{l',m} < 0$. However, we should consider the sign of input in practice because we usually use ReLUs as activation functions. As described in eq. (3), ReLUs are used before convolution as $\boldsymbol{K} * ReLU(\cdot)$. Thus, $\boldsymbol{x}$ cannot have negative elements, i.e., $x_m$ cannot be $\text{sign}(A_{l',m})$ when $\text{sign}(A_{l',m}) = -1$. Therefore, the induced $\infty$-norm can overestimate sensitivity to the perturbation. From this insight, we consider the norm of $\boldsymbol{Ax}$ when $\boldsymbol{x} = \boldsymbol{1}$ instead of eq. (8)

$$||\boldsymbol{A1}||_\infty = \max_l |\sum_m A_{l,m}| = |\sum_m \sum_l k_{l,m}|. \quad (11)$$

For robustness, we use this value as the regularization term.[3] We call our method *Absum* since this value is the absolute value of the summation of the filter coefficients.

The objective function of training with Absum is

$$\min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{p=1}^{N} f(\boldsymbol{\theta}, \boldsymbol{X}_p, \boldsymbol{Y}_p) + \lambda \sum_{i=1}^{L} g(\boldsymbol{K}^{(i)}), \quad (12)$$

$$g(\boldsymbol{K}^{(i)}) = |\sum_{m=1}^{n} \sum_{l=1}^{n} k^{(i)}_{l,m}|,$$

where $f(\cdot)$ is a loss function, $\boldsymbol{X}_p$ and $\boldsymbol{Y}_p$ are the $p$-th training image and label, respectively, $\boldsymbol{\theta}$ is the parameter vector including $\boldsymbol{K}^{(i)}$ in the model, and $\lambda$ is a regularization weight. The $\boldsymbol{K}^{(i)}$ is the filter matrix of the $i$-th convolution, and $L$ is the number of convolution filters.[4] Figure 3 shows search spaces of Absum (blue) and $L_1$ regularization (red) when we have two parameters. The constraint of Absum is looser than $L_1$ regularization because a large element $k^{(i)}_{l,m} \gg 0$ is allowed if a small element $k^{(i)}_{l',m'} \ll 0$ satisfies $|k^{(i)}_{l,m}| = |k^{(i)}_{l',m'}|$. Even if $|\sum_l \sum_m k_{l,m}| = 0$, the search space of Absum is a $n^2 - 1$ dimensional space $\{\boldsymbol{K} | \boldsymbol{K} \in \boldsymbol{R}^{n \times n}, \sum_l \sum_m k_{l,m} = 0\}$ while that of $L_1$ regularization is a point $\boldsymbol{K} = \boldsymbol{O}$ if $\sum_l \sum_m |k_{l,m}| = 0$. Note that the search space of weight decay is also the point $\boldsymbol{K} = \boldsymbol{O}$ when $||\boldsymbol{K}||_F = 0$. Therefore, the loss function with Absum can be lower than that with $L_1$ regularization and weight decay if we use a large $\lambda$.

Note that when the filter size is $h \times h$ and $h < n$, we only need to compute $|\sum_{m=1}^{h} \sum_{l=1}^{h} k_{l,m}|$ since zeros padded in $\boldsymbol{K}$ do not affect eq. (12) (hereafter, we use $h$ instead of $n$).

## Proximal Operator for Absum

Since $g(\boldsymbol{K})$ is not differentiable at $\sum_l \sum_m k_{l,m} = 0$, the gradient method might not be effective for minimizing eq. (12). To minimize eq. (12), we use a proximal gradient method, which can minimize a differentiable loss function with a non-differentiable regularization term (Parikh, Boyd, and others 2014). We now introduce proximal operator for Absum. For clarity, let $\bar{\boldsymbol{k}}$ be $\bar{\boldsymbol{k}} = \text{vec}(\boldsymbol{K}) = [\boldsymbol{k}_0^T, \ldots, \boldsymbol{k}_{h-1}^T]^T \in \boldsymbol{R}^{h^2}$. The proximal operator for $\lambda g(\bar{\boldsymbol{k}})$ is

$$\text{prox}_{\lambda g}(\bar{\boldsymbol{k}}) = \begin{cases} \bar{\boldsymbol{k}} + \lambda \boldsymbol{1} & \text{if } \sum_l \sum_m k_{l,m} < -h^2 \lambda, \\ \bar{\boldsymbol{k}} - \frac{\sum_l \sum_m k_{l,m}}{h^2} \boldsymbol{1} & \text{if } -h^2 \lambda \leq \sum_l \sum_m k_{l,m} \leq h^2 \lambda, \\ \bar{\boldsymbol{k}} - \lambda \boldsymbol{1} & \text{if } \sum_l \sum_m k_{l,m} > h^2 \lambda. \end{cases} \quad (13)$$

The following lemmas show that eq. (13) is the proximal operator for Absum:

**Lemma 1.** *If $\bar{\boldsymbol{k}} = [\bar{k}_1, \ldots, \bar{k}_{\bar{n}}]^T \in \boldsymbol{R}^{\bar{n}}$, $g(\bar{\boldsymbol{k}}) = |\sum_i \bar{k}_i|$ is a convex function.*

**Lemma 2.** *If $\bar{\boldsymbol{k}} = [\bar{k}_1, \ldots, \bar{k}_{\bar{n}}]^T \in \boldsymbol{R}^{\bar{n}}$, $\boldsymbol{u} \in \boldsymbol{R}^{\bar{n}}$ and $g(\bar{\boldsymbol{k}}) = |\sum_i \bar{k}_i|$, we have*

$$\text{prox}_{\lambda g}(\bar{\boldsymbol{k}}) = \arg\min_{\boldsymbol{u}} \frac{1}{2} ||\boldsymbol{u} - \bar{\boldsymbol{k}}||_2^2 + \lambda |\sum_i u_i| \quad (14)$$

$$= \begin{cases} \bar{\boldsymbol{k}} + \lambda \boldsymbol{1} & \text{if } \sum_i \bar{k}_i < -\bar{n}\lambda, \\ \bar{\boldsymbol{k}} - \frac{\sum_i \bar{k}_i}{\bar{n}} \boldsymbol{1} & \text{if } -\bar{n}\lambda \leq \sum_i \bar{k}_i \leq \bar{n}\lambda, \\ \bar{\boldsymbol{k}} - \lambda \boldsymbol{1} & \text{if } \sum_i \bar{k}_i > \bar{n}\lambda. \end{cases} \quad (15)$$

The proofs of lemmas are provided in appendix. Lemma 1 shows that we can use the proximal gradient method, and Lemma 2 shows that the proximal operator of Absum can be obtained as the closed-form of eq. (15). By using the proximal operator after stochastic gradient descent (SGD), we update the $i$-th convolution filter:

$$\bar{\boldsymbol{k}}^{(i)} \leftarrow \text{prox}_{\eta\lambda g}(\bar{\boldsymbol{k}}^{(i)} - \eta \nabla_{\bar{\boldsymbol{k}}^{(i)}} \frac{1}{B} \sum_{b=1}^{B} f(\boldsymbol{\theta}, \boldsymbol{X}_b, \boldsymbol{Y}_b)) \quad (16)$$

[3] Absum is not an upper bound of the induced norms but empirically achieves good performance as shown in Experiments.

[4] We penalize the filter matrix for each channel. If one convolution layer has $c_1$ output channels and $c_2$ input channels, the regularization term becomes $\lambda \sum_{l=1}^{c_1} \sum_{m=1}^{c_2} g(\boldsymbol{K}^{(l+(m-1)c_1)})$.

where $\eta$ is a learning rate, and $B$ is a minibatch size. We can compute the proximal operator in $O(h^2)$ time for each convolution when the filter size is $h \times h$ because we only need to compute the summation of parameters and elementwise operations. We can also compute weight decay and $L_1$ regularization in $O(h^2)$ since the number of parameters in each convolution is $h^2$. Therefore, the order of computational complexity of Absum is the same as those of weight decay and $L_1$ regularization. When we have $c$ input channels and $c$ output channels, the computational costs of Absum, weight decay, and $L_1$ regularization are $O(c^2h^2)$ and less than that of SNC $O(c^2n^2(c+\log(n)))$ where $n \geq h$.

Note that the loss function $f$ for training deep neural networks is usually non-convex while $g(\boldsymbol{K})$ is convex. Several studies investigate the proximal gradient method when $f$ is non-convex (Li and Lin 2015), and Wen et al. (2016) use the proximal gradient method for inducing sparse structures in deep learning. We observed that the algorithm of Absum can find a good parameter point during the experiments. In appendix, we compared Absum using the proximal gradient method with Absum using the automatic differentiation.

## Experiments

We discuss the evaluation of the effectiveness of SNC and Absum in improving robustness against SFA. Next, we show that Absum is more efficient than SNC especially when the size of input images and models are large. Finally, as the further investigation, we discuss the evaluation of the performance of Absum and SNC in terms of robustness against transferred attacks, vulnerability in frequency domain, and robustness against PGD when used with adversarial training. To evaluate effectiveness, we conducted experiments of image recognition on MNIST (LeCun et al. 1998), FMNIST (Xiao, Rasul, and Vollgraf 2017), CIFAR10, CIFAR100 (Krizhevsky and Hinton 2009), SVHN (Netzer et al. 2011), and ImageNet (ILSVRC2012) (Krizhevsky, Sutskever, and Hinton 2012). We compared Absum and SNC with standard regularizations (weight decay (WD) and $L_1$ regularization).

### Experimental Conditions

We provide details of the experimental conditions in appendix. In all experiments, we selected the best regularization weight from among $[10^1, 10^0, \ldots, 10^{-7}]$ for Absum and standard regularization methods, and the best spectral norm $\sigma$ from among $[0.01, 0.1, 0.5, 1.0, 10]$ for SNC. In SNC, we clipped $\sigma$ once in 100 iterations due to the large computational cost. For MNIST and FMNIST, we stacked two convolutional layers and two fully connected layers and used ReLUs as activation functions. For CIFAR10, CIFAR100, SVHN, and ImageNet, the model architecture was ResNet-18 (He et al. 2016). Additionally, we evaluated GoogLeNet (Szegedy et al. 2015) and DenseNet121 (Huang et al. 2017) on CIFAR10. We used SFA with $l, m \in \{0, 1, \ldots, 27\}$ and $\varepsilon = 80/255$ on MNIST and FMNIST, and $l, m \in \{0, 1, \ldots, 31\}$ and $\varepsilon = 10/255$ on CIFAR10, CIFAR100, SVHN, and ImageNet.

In addition, we used PGD to evaluate robustness against transferred attacks and white-box attacks since PGD is a sophisticated white-box attack. We evaluated robustness against PGD when we used adversarial training (Kurakin, Goodfellow, and Bengio 2016; Madry et al. 2018) with each method since Absum can be used with it. Model architectures were the same as in the experiments involving SFA. The hyperparameter settings for PGD were based on (Madry et al. 2018). The $L_\infty$ norm of the perturbation $\varepsilon$ was set to $\varepsilon = 0.3$ for MNIST and FMNIST and $\varepsilon = 8/255$ for CIFAR10, CIFAR100, and SVHN at training time. For PGD, we updated the perturbation for 40 iterations with a step size of 0.01 on MNIST and FMNIST at training and evaluation times, and on CIFAR10, CIFAR100, and SVHN, for 7 iterations with a step size of 2/255 at training time and 100 iterations with the same step size at evaluation time.

### Effectiveness and Efficiency

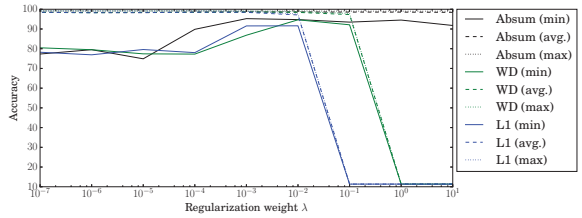
**Robustness against SFA** Table 1 lists the accuracies of each method on test data perturbed by SFA and selected $\lambda$ and $\sigma$. In this table, Avg. denote robust accuracies against SFA averaged over $(l, m)$, and Min. denotes minimum accuracies among hyperparameters $(l, m)$, i.e., robust accuracies against optimized SFA. CLN denotes accuracies on clean data. The $\lambda$ and $\sigma$ are selected so that Avg. would become the highest. In Tab. 1, Absum and SNC are more robust against SFA compared with WD and $L_1$. Although SNC is more robust than Absum on CIFAR10 and CIFAR100, clean accuracies of SNC are less than those of Absum and the computation time of SNC is larger than that of Absum as discussed below. Due to the computation cost, we could not evaluate SNC on ImageNet.

Figure 4 shows the test accuracies of the methods on MNIST and CIFAR10 perturbed by SFA against regularization weights. In this figure, min and max denote the minimum and maximum test accuracies among $(l, m)$, respectively, and avg. denotes test accuracies averaged over $(l, m)$. All methods tend to increase their minimum accuracy (results of SFA with optimized $(l, m)$) according to the regularization weight. However, $L_1$ and WD significantly decrease in accuracy when the regularization weight is higher than $10^{-1}$. On the other hand, Absum with the high regularization weight does not decrease in accuracy. Figure 5 shows the lowest training loss $\frac{1}{N}\sum f$ in training on CIFAR10 against $\lambda$. WD and $L_1$ with a large $\lambda$ prevent minimization of the training loss. On the other hand, Absum with a large $\lambda$ can decrease the training loss because the search space of $\boldsymbol{K}^{(i)} \in \boldsymbol{R}^{h \times h}$ has $h^2 - 1$ dimensional space even if $g(\boldsymbol{K}^{(i)}) = 0$. In conclusion, standard regularization methods might not be effective in improving robustness against SFA because the high regularization weight imposes too tight of constraints to minimize the loss function. On the other hand, Absum imposes looser constraints; thus, we can improve robustness while maintaining classification performance. The results of other datasets are almost the same as Fig. 4.

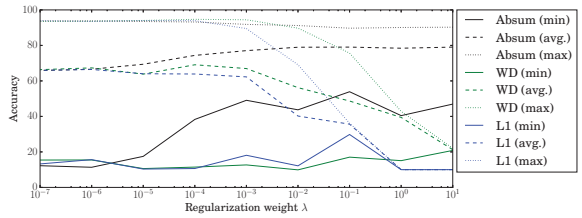
**Computational Cost** To confirm the efficiency of Absum, we evaluated the runtime for one epoch. We also evaluated the runtime of the forward and backward processes of ResNet-18 for one image when input size increases by using random synthetic three channels images whose sizes

Table 1: Accuracies on datasets perturbed by SFA.

| | Avg. | | | | Min. | | | | CLN | | | | $\lambda$ and $\sigma$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Absum | WD | L1 | SNC | Absum | WD | L1 | SNC | Absum | WD | L1 | SNC | Absum | WD | L1 | SNC |
| MNIST | **98.64** | 98.59 | 98.48 | 98.55 | **94.76** | 86.84 | 78.01 | 91.79 | 99.14 | 99.10 | **99.18** | 99.10 | $10^{-2}$ | $10^{-3}$ | $10^{-4}$ | 10 |
| FMNIST | **83.11** | 83.09 | 82.49 | 82.60 | **60.12** | 47.57 | 58.38 | 55.36 | **88.46** | 86.99 | 87.05 | 87.50 | $10^{-3}$ | $10^{-2}$ | $10^{-3}$ | 10 |
| CIFAR10 (ResNet18) | 79.05 | 69.09 | 66.44 | **85.57** | 53.90 | 11.44 | 15.64 | **73.99** | 89.69 | **94.73** | 93.41 | 88.37 | $10^{-1}$ | $10^{-4}$ | $10^{-6}$ | 0.5 |
| CIFAR10 (GoogLeNet) | 75.77 | 73.47 | 73.53 | **86.42** | 47.27 | 35.64 | 38.12 | **78.26** | 92.26 | 92.83 | **93.34** | 89.54 | $10^{-3}$ | $10^{-3}$ | $10^{-4}$ | 0.1 |
| CIFAR10 (DenseNet121) | 70.48 | 64.07 | 65.30 | **83.71** | 15.09 | 18.53 | 16.80 | **70.26** | 92.72 | **94.31** | 94.00 | 85.26 | $10^{-3}$ | $10^{-5}$ | $10^{-7}$ | 0.1 |
| CIFAR100 | 48.69 | 42.97 | 38.99 | **60.42** | 16.32 | 5.23 | 9.84 | **45.05** | 68.72 | 67.05 | **71.68** | 62.76 | $10^{-3}$ | $10^{-6}$ | $10^{-7}$ | 1 |
| SVHN | **93.34** | 91.74 | 91.14 | 93.20 | **73.69** | 60.36 | 57.52 | 62.90 | 95.93 | **96.37** | 96.20 | 95.42 | $10^{-3}$ | $10^{-3}$ | $10^{-7}$ | 0.1 |
| ImageNet (Top 1) | **26.59** | 9.92 | 5.18 | N/A | **15.06** | 4.39 | 1.01 | N/A | 58.51 | **70.70** | 64.63 | N/A | $10^{-2}$ | $10^{-4}$ | $10^{-4}$ | N/A |
| ImageNet (Top 5) | **48.51** | 20.15 | 12.29 | N/A | **31.29** | 10.64 | 3.08 | N/A | 81.17 | **89.71** | 85.79 | N/A | $10^{-2}$ | $10^{-4}$ | $10^{-4}$ | N/A |




(a) MNIST




(b) CIFAR10

Figure 4: Accuracy on test datasets perturbed by SFA vs $\lambda$

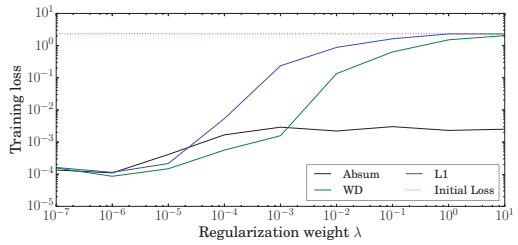
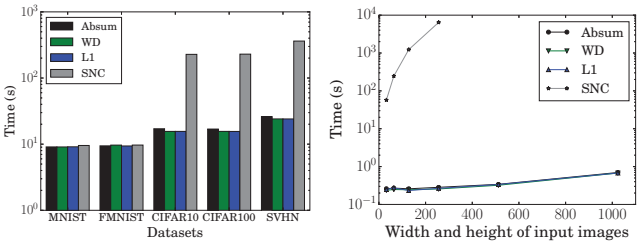



Figure 5: Training loss vs $\lambda$

were 32×32, 64×64, 128×128, 256×256, 512×512, and 1024×1024 with 10 random labels. The results are shown in Fig. 6. As shown in Fig. 6 (a), Absum is about ten times faster than SNC on 32 × 32 image datasets with ResNet18. The runtime of SNC is comparable to those of other methods on MNIST and FMNIST because we use only two convolution layers, and image sizes of these datasets are smaller than other datasets. In Fig. 6 (b), the runtime of Absum does not increase significantly compared with SNC and the increase in the runtime of Absum is similar to those of standard regularization methods. This is because the computational cost of Absum does not depend on the size of input images. Since SNC incurs large computational cost and depends on the input size, we could not evaluate the runtime when the image width is larger than 256.




(a) Runtime for one epoch (b) Runtime vs. input size

Figure 6: Computation time

Table 2: Robust accuracy against transferred PGD attacks. w/o Reg. denotes results of training without regularization.

| | Absum | WD | L1 | SNC | w/o Reg. |
|---|---|---|---|---|---|
| MNIST ($\varepsilon=0.2$) | **76.34** | 48.94 | 66.48 | 71.30 | 65.87 |
| FMNIST ($\varepsilon=0.2$) | **30.08** | 3.46 | 18.35 | 21.31 | 19.74 |
| CIFAR10 ($\varepsilon=4/255$) | 26.29 | 18.48 | 15.66 | **48.85** | 15.85 |
| CIFAR100 ($\varepsilon=4/255$) | 18.57 | 17.40 | 16.68 | **36.57** | 16.68 |
| SVHN ($\varepsilon=4/255$) | 49.11 | 40.49 | 52.79 | 46.36 | **54.39** |

## Extensive Empirical Investigation

**Robustness against Transferred Attacks** Sensitivity to SFA is caused by convolution operation and is universal for CNNs. This sensitivity might be a cause of transferability of adversarial attacks, and robust CNNs against SFA can be robust against transferred attacks. To confirm this hypothesis, we investigate sensitivity to transferred PGD. We generate adversarial examples by using the substitute models that were trained under the same setting as that presented in the previous section but with different random initializations. We used these substitute models rather than completely different models because they can be regarded as one of the worst-case instances for transferred attacks (Madry et al. 2018). The accuracies on these adversarial examples are listed in Tab. 2. Absum and SNC improve robustness compared to WD and $L_1$. Tables 1 and 2 imply that the method of improving robustness against SFA can also improve robustness against the transferred attacks. This is the first study that shows the relation between robustness against SFA and against transferred white-box attacks.

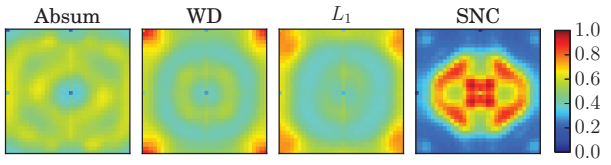



Figure 7: Power spectra of PGD perturbations on CIFAR10. Magnitudes in low-frequency and high-frequency domains are located near center and edge of each figure, respectively. They are normalized as in (0, 1) after logarithmic transform.

Table 3: Robust accuracy against High-Frequency attacks.

| | Absum | WD | L1 | SNC | w/o Reg. |
|---|---|---|---|---|---|
| MNIST | 99.00 | 98.98 | **99.10** | 98.97 | 99.01 |
| FMNIST | 84.15 | 83.91 | 82.56 | **84.30** | 84.03 |
| CIFAR10 | 64.51 | 52.82 | 47.01 | **82.11** | 47.46 |
| CIFAR100 | 41.44 | 36.15 | 31.53 | **61.22** | 31.80 |
| SVHN | **52.95** | 28.11 | 17.03 | 18.75 | 11.13 |

**Sensitivity in Frequency Domain** Several studies show that CNNs are sensitive to high-frequency noise unlike human visual systems since CNNs are biased towards high-frequency information (Wang et al. 2019; Yin et al. 2019). From the robustness against SFA, which is regarded as single-frequency noise, Absum and SNC can be expected not to bias CNNs towards high-frequency information. To confirm this hypothesis, we first investigate the power spectra of adversarial perturbations of models trained using each method. Next, we investigate robustness against High-Frequency attacks, which remove high-frequency components of image data. High-Frequency attacks have a hyper-parameter of radius that determines the cutoff frequency, and we set it as half the image width. In these experiments, $\lambda$ and $\sigma$ are the same as those in Tab. 1.

Figure 7 shows the power spectra of PGD perturbations on CIFAR10 and Tab. 3 lists the accuracies on the test data processed by High-Frequency attacks. In Fig. 7, we shift low frequency components to the center of the spectrum and power spectra are averaged over test data and RGB channels. This figure shows that vulnerabilities of WD and $L_1$ are biased in the high-frequency domain, while vulnerability of SNC is highly biased in the low-frequency domain. The power spectrum of Absum is not biased towards a specific frequency domain. Due to these characteristics, SNC and Absum are more robust against High-Frequency attacks than WD and $L_1$ (Tab. 3). Since human visual systems can perceive low-frequency noise better than high-frequency noise, attacks for Absum and SNC might be more perceptible than attacks for WD and $L_1$. Note that we observed that Absum is more robust against high-pass filtering than SNC. This result supports that Absum does not bias CNNs towards a specific frequency domain while SNC biases CNNs towards the low-frequency domain.

**Robustness against PGD with Adversarial Training** Table 4 lists the accuracies of models trained by adversarial training on data perturbed by PGD. When using adversarial training, Absum improves robustness against PGD, the highest among regularization methods, on almost all datasets. This implies that sensitivity to SFA is one of the causes of vulnerabilities of CNNs. The $\lambda$ of Absum tends to be higher than the $\lambda$ of WD and $L_1$; thus, Absum can also improve robustness against PGD without preventing minimization due to its looseness. Note that Absum does not improve robustness against PGD whithout adversarial training since the structural sensitivity of CNNs does not necessarily cause all vulnerabilities of CNN-based models. Even so, Absum is more effective than other standard regularizations since it can efficiently improve robustness against black-box attacks (SFA, transferred attacks, and High-Frequency attacks) and enhance adversarial training, as discussed above.

Table 4: Accuracies (%) on test datasets perturbed by PGD.

| MNIST | Adversarial training | | | | | |
|---|---|---|---|---|---|---|
| $\varepsilon$ | 0.05 | 0.10 | 0.15 | 0.20 | 0.25 | 0.30 |
| Absum $\lambda = 10^{-3}$ | **96.01** | **94.92** | **93.75** | **92.73** | **91.59** | **90.78** |
| WD $\lambda = 10^{-4}$ | 92.97 | 91.34 | 89.69 | 88.02 | 87.05 | 85.96 |
| L1 $\lambda = 10^{-4}$ | 93.12 | 91.86 | 90.60 | 89.28 | 88.25 | 87.06 |
| SNC $\sigma = 10$ | 91.92 | 89.43 | 86.77 | 83.89 | 80.24 | 76.92 |
| w/o Reg. | 91.57 | 89.85 | 88.43 | 86.87 | 85.76 | 84.86 |
| **FMNIST** | **Adversarial training** | | | | | |
| Absum $\lambda = 10^{-3}$ | **66.94** | **65.92** | **65.77** | **65.52** | **65.24** | **64.95** |
| WD $\lambda = 10^{-5}$ | 65.38 | 63.64 | 62.91 | 62.60 | 62.11 | 61.96 |
| L1 $\lambda = 10^{-6}$ | 66.13 | 64.16 | 62.95 | 62.23 | 61.64 | 61.66 |
| SNC $\sigma = 10$ | 51.58 | 49.33 | 47.31 | 45.85 | 44.86 | 44.04 |
| w/o Reg. | 63.36 | 61.66 | 61.15 | 60.97 | 60.46 | 60.26 |
| **CIFAR10** | **Adversarial training** | | | | | |
| $\varepsilon$ | 4/255 | 8/255 | 12/255 | 16/255 | 20/255 | |
| Absum $\lambda = 10^{-5}$ | 69.42 | 49.39 | **30.22** | **15.03** | **6.54** | |
| WD $\lambda = 10^{-5}$ | **69.48** | 49.38 | 29.37 | 14.45 | 6.06 | |
| L1 $\lambda = 10^{-5}$ | 68.99 | **49.45** | 29.51 | 14.68 | 6.31 | |
| SNC $\sigma = 10$ | 68.47 | 48.74 | 29.07 | 14.32 | 6.04 | |
| w/o Reg. | 68.46 | 48.77 | 29.20 | 14.50 | 6.08 | |
| **CIFAR100** | **Adversarial training** | | | | | |
| Absum $\lambda = 10^{-4}$ | **42.19** | **27.25** | 15.89 | **8.47** | 4.14 | |
| WD $\lambda = 10^{-7}$ | 41.14 | 27.05 | **15.90** | 8.26 | **4.28** | |
| L1 $\lambda = 10^{-4}$ | 40.75 | 26.14 | 14.45 | 7.61 | 3.67 | |
| SNC $\sigma = 10$ | 40.90 | 26.61 | 15.53 | 8.32 | 4.13 | |
| w/o Reg. | 40.70 | 26.24 | 14.85 | 7.94 | 3.86 | |
| **SVHN** | **Adversarial training** | | | | | |
| Absum $\lambda = 10^{-5}$ | **77.78** | **52.74** | **27.39** | 11.97 | 5.50 | |
| WD $\lambda = 10^{-7}$ | 76.66 | 50.40 | 25.05 | 10.86 | 5.04 | |
| L1 $\lambda = 10^{-6}$ | 76.50 | 51.49 | 27.10 | **12.12** | **5.63** | |
| SNC $\sigma = 1.0$ | 77.23 | 50.80 | 25.24 | 11.04 | 5.03 | |
| w/o Reg. | N/A | N/A | N/A | N/A | N/A | |

## Conclusion

We proposed Absum; an efficient defense method against SFA that can reduce the structural sensitivity of CNNs with ReLUs while its computational cost remains comparable to standard regularizations. By reducing the structural sensitivity, Absum can improve robustness against not only SFA, but also transferred PGD, and High-Frequency attacks. Due to its simplicity, Absum can be used with other methods, and Absum can enhance adversarial training of PGD.

# Appendix

## Proofs of Lemmas

In this section, we provide the proofs of the lemmas in the paper.

**Lemma 1.** *If $\bar{\boldsymbol{k}} = [\bar{k}_1, \ldots, \bar{k}_{\bar{n}}]^T \in \boldsymbol{R}^{\bar{n}}$, $g(\bar{\boldsymbol{k}}) = |\sum_i^{\bar{n}} \bar{k}_i|$ is a convex function.*

*Proof.* If $g(\cdot)$ is a convex function, we have $g(t\boldsymbol{x} + (1-t)\boldsymbol{y}) \leq tg(\boldsymbol{x}) + (1-t)g(\boldsymbol{y})$, where $t \in [0,1]$ and $\forall \boldsymbol{x}, \boldsymbol{y} \in \boldsymbol{R}^{\bar{n}}$. Thus, we investigate $J = tg(\boldsymbol{x}) + (1-t)g(\boldsymbol{y}) - g(t\boldsymbol{x} + (1-t)\boldsymbol{y})$, and if $J \geq 0$, we prove the lemma. We have

$$\begin{aligned} J =& t|\textstyle\sum_i x_i| + (1-t)|\sum_i y_i| - |\sum_i (tx_i + (1-t)y_i)|, \\ =& |t\textstyle\sum_i x_i| + |(1-t)\sum_i y_i| - |t\sum_i x_i + (1-t)\sum_i y_i|, \end{aligned} \quad (17)$$

since $t \geq 0$ and $1-t \geq 0$. Let $\alpha = t\sum_i x_i$ and $\beta = (1-t)\sum_i y_i$; thus, we have $J = |\alpha| + |\beta| - |\alpha + \beta|$. From the triangle inequality, we have $J \geq 0$; thus, this completes the proof. □

**Lemma 2.** *If $\bar{\boldsymbol{k}} = [\bar{k}_1, \ldots, \bar{k}_{\bar{n}}]^T \in \boldsymbol{R}^{\bar{n}}$, $\boldsymbol{u} \in \boldsymbol{R}^{\bar{n}}$ and $g(\bar{\boldsymbol{k}}) = |\sum_i^{\bar{n}} \bar{k}_i|$, we have*

$$\begin{aligned} \mathrm{prox}_{\lambda g}(\bar{\boldsymbol{k}}) =& \arg\min_{\boldsymbol{u}} \tfrac{1}{2}||\boldsymbol{u} - \bar{\boldsymbol{k}}||_2^2 + \lambda|\textstyle\sum_i^{\bar{n}} u_i| \\ =& \begin{cases} \bar{\boldsymbol{k}} + \lambda\boldsymbol{1} & \text{if} \quad \sum_i^{\bar{n}} \bar{k}_i < -\bar{n}\lambda, \\ \bar{\boldsymbol{k}} - \frac{\sum_i^{\bar{n}} k_i}{\bar{n}}\boldsymbol{1} & \text{if} \quad -\bar{n}\lambda \leq \sum_i^{\bar{n}} \bar{k}_i \leq \bar{n}\lambda, \\ \bar{\boldsymbol{k}} - \lambda\boldsymbol{1} & \text{if} \quad \sum_i^{\bar{n}} \bar{k}_i > \bar{n}\lambda. \end{cases} \end{aligned}$$

*Proof.* For clarity, let $J = \frac{1}{2}||\boldsymbol{u} - \bar{\boldsymbol{k}}||_2^2 + \lambda|\sum_i^{\bar{n}} u_i|$. We have three cases; (a) $\sum_i^{\bar{n}} u_i > 0$, (b) $\sum_i^{\bar{n}} u_i < 0$, and (c) $\sum_i^{\bar{n}} u_i = 0$. In (a), we have $|\sum_i^{\bar{n}} u_i| = \sum_i^{\bar{n}} u_i$, and $\frac{\partial J}{\partial u_i} = u_i - \bar{k}_i + \lambda = 0$ at the optimal point. Therefore, $u_i = \bar{k}_i - \lambda$, and the solution becomes $\boldsymbol{u} = \bar{\boldsymbol{k}} - \lambda\boldsymbol{1}$. The condition is $\sum_i^{\bar{n}} u_i = \sum_i^{\bar{n}} \bar{k}_i - \bar{n}\lambda > 0$, i.e., $\sum_i^{\bar{n}} \bar{k}_i > \bar{n}\lambda$. In (b), we have $|\sum_i^{\bar{n}} u_i| = -\sum_i^{\bar{n}} u_i$, and we can optimize $J = \frac{1}{2}||\boldsymbol{u} - \bar{\boldsymbol{k}}||_2^2 - \lambda\sum_i^{\bar{n}} u_i$ in the same manner as (a). As a result, $\boldsymbol{u} = \bar{\boldsymbol{k}} + \lambda\boldsymbol{1}$ if $\sum_i^{\bar{n}} \bar{k}_i < -\bar{n}\lambda$. In (c), $|\sum_i^{\bar{n}} u_i|$ is non-differentiable, but we can use subgradient $\boldsymbol{v}$ such as $|\sum_i^{\bar{n}} z_i| \geq |\sum_i^{\bar{n}} u_i| + \boldsymbol{v}^T(\boldsymbol{z} - \boldsymbol{u})$. Let $B$ be $B = \{\boldsymbol{u} \pm r\boldsymbol{e}_i | i = 1, \ldots, n\}$ where small $r > 0$ and $\boldsymbol{e}_k$ be the standard basis; thus, we have $M = \max_{\boldsymbol{z} \in \boldsymbol{B}} |\sum_i z_i| = |\sum_i u_i \pm r| = r$ when $|\sum_i u_i| = 0$. As a result, $\boldsymbol{v}$ is bounded as $||\boldsymbol{v}||_\infty \leq \frac{M - |\sum_i u_i|}{r} = 1$. We then have $\frac{\partial J}{\partial u_i} = u_i - \bar{k}_i + \lambda v_i = 0$; thus, $\boldsymbol{u} = \bar{\boldsymbol{k}} - \lambda\boldsymbol{v}$. Since $||\boldsymbol{v}||_\infty \leq 1$, we have $-\bar{n}\lambda \leq \lambda\sum v_i \leq \bar{n}\lambda$. Thus, the condition becomes $-\bar{n}\lambda \leq \sum_i \bar{k}_i \leq \bar{n}\lambda$ since $\sum_i u_i = \sum_i \bar{k}_i + \lambda\sum_i v_i = 0$. By substituting $\boldsymbol{u} = \bar{\boldsymbol{k}} - \lambda\boldsymbol{v}$ into $J$, we have $J = \frac{1}{2}||\lambda\boldsymbol{v}||_2^2$ subject to $\sum_i v_i = \frac{\sum_i k_i}{\lambda}$ and $||\boldsymbol{v}||_\infty \leq 1$. Thus, the minimum point is $v_1 = v_2 = \ldots = v_{\bar{n}} = \frac{\sum_i k_i}{\bar{n}\lambda}$, i.e., $\boldsymbol{v} = \frac{\sum_i k_i}{\bar{n}\lambda}\boldsymbol{1}$. Therefore, $\boldsymbol{u} = \bar{\boldsymbol{k}} - \frac{\sum_i k_i}{\bar{n}}\boldsymbol{1}$ is the minimum point when $-\bar{n}\lambda \leq \sum_i \bar{k}_i \leq \bar{n}\lambda$. This completes the proof. □

## Inequality of Induced Norms for Convolution

The $u + n(v-1)$-th singular value of a doubly circulant matrix $\boldsymbol{A}$ can be written as $\sigma_{u,v} = |\sum_{l,m} k_{l,m} \exp(j\frac{2\pi}{n}(ul + vm))|$ (not arranged in descending order), and we have $||\boldsymbol{A}||_2 = \max_{u,v} \sigma_{u,v} \leq \sum_{l,m} |k_{l,m}| \leq ||\boldsymbol{A}||_\infty$. Therefore, the spectral norm of $\boldsymbol{A}$ is bounded above by the induced $\infty$-norm as $||\boldsymbol{A}||_2 \leq ||\boldsymbol{A}||_\infty$.

Next, we explain that $L_2$ regularization (weight decay: WD) can constrain the induced norm of a convolutional layer. The $L_2$ regularization term of the convolution filter $\boldsymbol{K} \in \boldsymbol{R}^{n \times n}$ is $\sum_l^n \sum_m^n k_{l,m}^2$. On the other hand, the square of the Frobenius norm of $\boldsymbol{A}$ becomes $||\boldsymbol{A}||_F^2 = \sum_l^{n^2} \sum_m^{n^2} A_{l,m}^2 = n^2 \sum_l^n \sum_m^n k_{l,m}^2$. Thus, if we use the $L_2$ regularization, we constrain the Frobenius norm of $\boldsymbol{A}$. In addition, let $\boldsymbol{M}$ be $m \times m$ matrices. We have $||\boldsymbol{M}||_2 \leq ||\boldsymbol{M}||_F$, $\frac{||\boldsymbol{M}||_\infty}{\sqrt{m}} \leq ||\boldsymbol{M}||_2 \leq \sqrt{m}||\boldsymbol{M}||_\infty$ where $||\cdot||_2$ is the induced 2-norm, which is the largest singular value. From the above inequalities, we have $\frac{||\boldsymbol{A}||_\infty}{n} \leq ||\boldsymbol{A}||_2 \leq ||\boldsymbol{A}||_F$, and thus, if we decrease the Frobenius norm of $\boldsymbol{A}$, the induced 2-norm and $\infty$-norms are also decreased.

## Experimental Conditions

We had roughly two experimental conditions according to the dataset. Our experiments ran once for each hyperparameter. We assumed that all images were divided by 255 and pixels had the values between 0 and 1. In addition, MNIST, CIFAR10 and CIFAR100 were standardized as (mean, standard deviation)=(0,1) before the images were applied to the models as preprocessing. In the evaluation of robustness, we standardized input images by using the means and standard deviations of clean data after adversarial perturbation. The computation graph of the standardization was preserved in gradient-based attacks; thus, perturbations of PGD were optimized while considering this preprocess.

**MNIST and Fashion-MNIST** For MNIST and Fashion-MNIST (FMNIST), we stacked two convolutional layers and two fully connected layers, the first convolutional layer had the 10 output channels and the second convolutional layer had 20 output channels. The kernel sizes of the convolutional layers were 5, their strides were 1, and we did not use zero-padding in these layers. After each convolutional layer, we applied max pooling (the stride was 2) and ReLU activation. The output of the second convolutional layer was applied to the first fully connected layer (the size was $320 \times 50$), and we used the ReLU activation after the first fully connected layer. The size of the second fully connected layer was $50 \times 10$, and we used softmax as the output function. After the second convolution layer and before the second fully connected layer, we applied 50 % dropout. We trained the model for 100 epochs by using Momentum SGD (the learning rate of 0.01 and momentum of 0.5). We set the minibatch size to 64. For fair comparison, all regularization methods were applied to only convolution filter parameters.

**CIFAR10, CIFAR100, SVHN, and ImageNet** For SVHN, we used cropped digits, which were cropped as $32 \times 32$. The model architecture was ResNet-18 for CIFAR10, CIFAR100, and SVHN (He et al. 2016).[5] As the preprocessing for training, given images were randomly

[5]Our training settings are based on the open source of https://github.com/kuangliu/pytorch-cifar.

cropped as $32 \times 32$ after padding a sequence of four on each border of the images. Horizontal flip was randomly applied to images with a probability of 0.5. We trained the model for 350 epochs with Momentum SGD (momentum 0.9). The initial learning rate was set to 0.1, and after 150 and 250 epochs, we divided the learning rate by 10. We set the mini-batch size to 128. For ImageNet, we used hyperparameters in the example code of PyTorch except for regularization weight, and we trained models for 240 epochs. For fair comparison, all regularization methods were applied to only convolution filter parameters.

Note that about 20 % of SVHN have the class label of '1'. Due to the class imbalance, models output class '1' regardless of input images in some hyperparameter settings. In this case, the robust accuracies are always about 20%; thus, these models sometimes outperform properly trained models in terms of robust accuracy. However, these results are not meaningful, and we do not list them in the tables. For the other datasets, we also do not list the results of the models that output one class regardless of input images.

**High-Frequency Attack** To evaluate robustness in the frequency domain, we used High-Frequency attacks. High-Frequency attacks can be regarded as low-pass filteres, which remove high-frequency components. In High-Frequency attacks (Wang et al. 2019), we first apply discrete Fourier transform (DFT) $\mathcal{F}$ to data $\boldsymbol{X}$ as $\boldsymbol{Z} = \mathcal{F}(\boldsymbol{X})$. Next, we decompose the low- and high-frequency components as

$$Z^l_{i,j} = \begin{cases} Z_{i,j} & \text{if } d((i,j),(c_i,c_j)) \leq r \\ 0 & \text{otherwise} \end{cases}, \tag{18}$$

$$Z^h_{i,j} = \begin{cases} 0 & \text{if } d((i,j),(c_i,c_j)) \leq r \\ Z_{i,j} & \text{otherwise} \end{cases}, \tag{19}$$

where $Z^l_{i,j}$ and $Z^h_{i,j}$ are elements of low- and high-frequency components in the frequency domain, respectively, $(c_i, c_j)$ is a centroid of the image, $d(\cdot,\cdot)$ is the Euclidean distance, and $r$ is a radius that determines the cutoff frequency. Finally, we apply the inverse DFT to $\boldsymbol{Z}^l$ as $\boldsymbol{X}^l = \mathcal{F}^{-1}(\boldsymbol{Z}^l)$, and $\boldsymbol{X}^l$ is an input image attacked by High-Frequency attacks. While $r$ is gradually reduced and accuracies are iteratively evaluated for each $r$ in (Wang et al. 2019), we used fixed $r$ as half of the image width since we just focus on comparing Absum with other methods.

**Computational Cost** We used one NVIDIA Tesla V100 GPU and 32 Intel(R) Xeon(R) Silver 4110 CPUs, and our implementation used Python 3.6.8, pytorch 0.4.1, CUDA 9.0, and numpy 1.11.3. Note that we used numpy to compute the FFT and singular value decomposition, which is difficult to parallelize, in SNC. The model architectures and training process were the same as those of the experiments involving SFA. We used $\lambda = 10^{-4}$ and $\sigma = 1.0$.

**Robustness against PGD** We evaluated Absum with adversarial training (Goodfellow, Shlens, and Szegedy 2014) because Absum and other regularization methods can be used with adversarial training. In these experiments, we used advertorch (Ding, Wang, and Jin 2019) to generate adversarial examples of PGD.

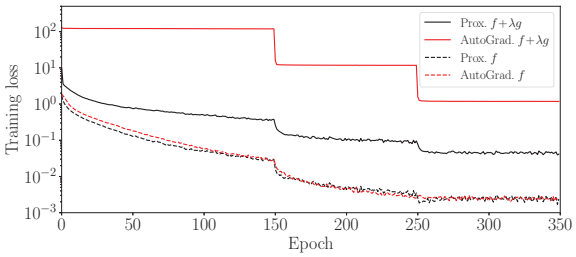



Figure 8: Convergence of the proximal gradient method.

Model architectures and training conditions were almost the same as those discussed in MNIST and Fashion-MNIST and CIFAR10, CIFAR100, SVHN, and ImageNet Sections. The number of epochs for MNIST and FMNIST was set to 100 in adversarial training since adversarial training required more epochs than naive training in this experiment. On the other hand, we observed overfitting in the adversarial training on CIFAR10, CIFAR100, and SVHN. Therefore, we trained the model for 150 epochs with Momentum SGD (momentum 0.9). The initial learning rate was set to 0.1, and after 50 and 100 epochs, we divided the learning rate by 10. We also applied weight decay of $10^{-4}$ to all parameters on CIFAR10 and CIFAR100 in the adversarial training of PGD since overfitting easily occurred in adversarial training on these datasets.

In PGD, the $L_\infty$ norm of the perturbation $\varepsilon$ was set to $\varepsilon = [0.05, 0.1, 0.15, 0.2, 0.25, 0.4]$ for MNIST and FMNIST, and $\varepsilon = [4/255, 8/255, 12/255, 16/255, 20/255]$ for CIFAR10 at evaluation time. For PGD, we updated the perturbation for 40 iterations with a step size of 0.01 on MNIST and FMNIST at training and evaluation times. On CIFAR10, CIFAR100, and SVHN, we updated the perturbation for 7 iterations with a step size of 2/255 at training time and 100 iterations at evaluation time. The starting points of PGD were randomly initialized from a uniform distribution of [-2/255, 2/255]. For adversarial training, we used training data perturbed by PGD with $\varepsilon = 0.3$ on MNIST and $\varepsilon = 8/255$ on CIFAR10, CIFAR100, and SVHN. In adversarial training, we only used adversarial examples of training data. The above conditions are based on (Madry et al. 2018).

## Effectiveness of proximal operator

To evaluate the proximal operator for Absum, we compared the convergence of the proximal gradient method with that of automatic differentiation based optimization; i.e. we calculate the gradient of $\frac{1}{N}\sum f + \lambda \sum g$ by automatic differentiation of PyTorch and minimize it by SGD. Note that in the automatic differentiation, the derivative at $|\sum_l \sum_m k_{l,m}| = 0$ is 0. Figure 8 shows $\frac{1}{N}\sum f + \lambda \sum g$ and $\frac{1}{N}\sum f$ on CIFAR10 against epochs when $\lambda = 0.01$. Prox. is the result of the proximal gradient method and AutoGrad. is the result of the automatic differentiation based optimization. In this figure, AutoGrad does not effectively minimize the regularization term $\lambda g$ while it can minimize the loss function $f$. On the other hand, the proximal gradient method minimizes the loss function and regularization term.

## References


Brendel, W.; Rauber, J.; and Bethge, M. 2018. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *Proc. ICLR*.

Chen, P.-Y.; Zhang, H.; Sharma, Y.; Yi, J.; and Hsieh, C.-J. 2017. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 15–26. ACM.

Das, N.; Shanbhogue, M.; Chen, S.-T.; Hohman, F.; Li, S.; Chen, L.; Kounavis, M. E.; and Chau, D. H. 2018. Shield: Fast, practical defense and vaccination for deep learning using jpeg compression. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 196–204. ACM.

Ding, G. W.; Wang, L.; and Jin, X. 2019. AdverTorch v0.1: An adversarial robustness toolbox based on pytorch. *arXiv preprint arXiv:1902.07623*.

Goodfellow, I.; Bengio, Y.; and Courville, A. 2016. *Deep learning*. MIT press.

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

Gouk, H.; Frank, E.; Pfahringer, B.; and Cree, M. 2018. Regularisation of neural networks by enforcing lipschitz continuity. *arXiv preprint arXiv:1804.04368*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proc. CVPR*, 770–778.

Huang, G.; Liu, Z.; Van Der Maaten, L.; and Weinberger, K. Q. 2017. Densely connected convolutional networks. In *Proc. CVPR*, 4700–4708.

Ilyas, A.; Engstrom, L.; Athalye, A.; and Lin, J. 2018. Black-box adversarial attacks with limited queries and information. In *Proc. ICML*, 2137–2146.

Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. ICML*, 448–456.

Jain, A. K. 1989. *Fundamentals of Digital Image Processing*. Prentice-Hall.

Jo, J., and Bengio, Y. 2017. Measuring the tendency of cnns to learn surface statistical regularities. *arXiv preprint arXiv:1711.11561*.

Karner, H.; Schneid, J.; and Ueberhuber, C. W. 2003. Spectral decomposition of real circulant matrices. *Linear Algebra and Its Applications* 367:301–311.

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

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Proc. NeurIPS*, 1097–1105.

Kurakin, A.; Goodfellow, I.; and Bengio, S. 2016. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*.

LeCun, Y.; Boser, B.; Denker, J. S.; Henderson, D.; Howard, R. E.; Hubbard, W.; and Jackel, L. D. 1989. Backpropagation applied to handwritten zip code recognition. *Neural computation* 1(4):541–551.

LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P.; et al. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11):2278–2324.

Li, H., and Lin, Z. 2015. Accelerated proximal gradient methods for nonconvex programming. In *Proc. NeurIPS*, 379–387.

Liu, Z.; Liu, Q.; Liu, T.; Xu, N.; Lin, X.; Wang, Y.; and Wen, W. 2019. Feature distillation: Dnn-oriented jpeg compression against adversarial examples. In *Proc. CVPR*, 860–868.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2018. Towards deep learning models resistant to adversarial attacks. In *Proc. ICLR*.

Moosavi-Dezfooli, S.-M.; Fawzi, A.; and Frossard, P. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *Proc. CVPR*, 2574–2582.

Nair, V., and Hinton, G. E. 2010. Rectified linear units improve restricted boltzmann machines. In *Proc. ICML*, 807–814. Omnipress.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; and Ng, A. Y. 2011. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*.

Papernot, N.; McDaniel, P.; Goodfellow, I.; Jha, S.; Celik, Z. B.; and Swami, A. 2017. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, 506–519. ACM.

Papernot, N.; McDaniel, P.; and Goodfellow, I. 2016. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*.

Parikh, N.; Boyd, S.; et al. 2014. Proximal algorithms. *Foundations and Trends® in Optimization* 1(3):127–239.

Radford, A.; Metz, L.; and Chintala, S. 2016. Unsupervised representation learning with deep convolutional generative adversarial networks. In *Proc. ICLR*.

Sedghi, H.; Gupta, V.; and Long, P. M. 2019. The singular values of convolutional layers. In *Proc. ICLR*.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *Proc. CVPR*, 1–9.

Tsuzuku, Y., and Sato, I. 2019. On the structural sensitivity of deep convolutional networks to the directions of fourier basis functions. In *Proc. CVPR*, 51–60.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Proc. NeurIPS*, 5998–6008.

Wang, H.; Wu, X.; Yin, P.; and Xing, E. P. 2019. High frequency component helps explain the generalization of convolutional neural networks. *arXiv preprint arXiv:1905.13545*.

Wen, W.; Wu, C.; Wang, Y.; Chen, Y.; and Li, H. 2016. Learning structured sparsity in deep neural networks. In *Proc. NeurIPS*, 2074–2082.

Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

Yin, D.; Gontijo Lopes, R.; Shlens, J.; Cubuk, E. D.; and Gilmer, J. 2019. A fourier perspective on model robustness in computer vision. In *Proc. NeurIPS*, 13255–13265.

Yoshida, Y., and Miyato, T. 2017. Spectral norm regularization for improving the generalizability of deep learning. *arXiv preprint arXiv:1705.10941*.

Yuan, X.; He, P.; Zhu, Q.; and Li, X. 2019. Adversarial examples: Attacks and defenses for deep learning. *IEEE transactions on neural networks and learning systems*.