# DATA-FREE BACKDOOR REMOVAL BASED ON CHANNEL LIPSCHITZNESS

**Runkai Zheng** [*]
School of Data Science
The Chinese University of Hong Kong, Shenzhen
Shenzhen
rkteddy@outlook.com

**Rongjun Tang** [*]
School of Science and Engineering
The Chinese University of Hong Kong, Shenzhen
Shenzhen
220019009@link.cuhk.edu.cn

**Jianze Li, Li Liu** [†]
Shenzhen Research Institute of Big Data
The Chinese University of Hong Kong, Shenzhen
Shenzhen
lijianze@gmail.com, liuli@cuhk.edu.cn

## ABSTRACT

Recent studies have shown that Deep Neural Networks (DNNs) are vulnerable to the backdoor attacks, which leads to malicious behaviors of DNNs when specific triggers are attached to the input images. It was further demonstrated that the infected DNNs possess a collection of channels, which are more sensitive to the backdoor triggers compared with normal channels. Pruning these channels was then shown to be effective in mitigating the backdoor behaviors. To locate those channels, it is natural to consider their Lipschitzness, which measures their sensitivity against worst-case perturbations on the inputs. In this work, we introduce a novel concept called Channel Lipschitz Constant (CLC), which is defined as the Lipschitz constant of the mapping from the input images to the output of each channel. Then we provide empirical evidences to show the strong correlation between an Upper bound of the CLC (UCLC) and the trigger-activated change on the channel activation. Since UCLC can be directly calculated from the weight matrices, we can detect the potential backdoor channels in a data-free manner, and do simple pruning on the infected DNN to repair the model. The proposed Channel Lipschitzness based Pruning (CLP) method is super fast, simple, data-free and robust to the choice of the pruning threshold. Extensive experiments are conducted to evaluate the efficiency and effectiveness of CLP, which achieves state-of-the-art results among the mainstream defense methods even without any data. Source codes are available at https://github.com/rkteddy/channel-Lipschitzness-based-pruning.

***K*eywords** Deep neural network, Backdoor defense, Lipschitz constant, Model pruning.

## 1 Introduction

In recent years, Deep Neural Networks (DNNs) have achieved significantly advanced performance in a wide range of fields, and have been further applied to industrial practices, including some security-critical fields, *e.g.*, audio-visual processing [1] and medical image processing [2]. Accordingly, the model security problems have gained much attention from the community.

One of the well-known model security problems of DNN is the adversarial attack [3], [4], [5], which has been extensively studied. The main idea is to add imperceptible perturbation into a well-classified sample to mislead the model prediction during the testing phase. Recently, backdoor attacks [6] remarkably show severe threats to model security due to its

[*]Equal Contribution
[†]Corresponding Author

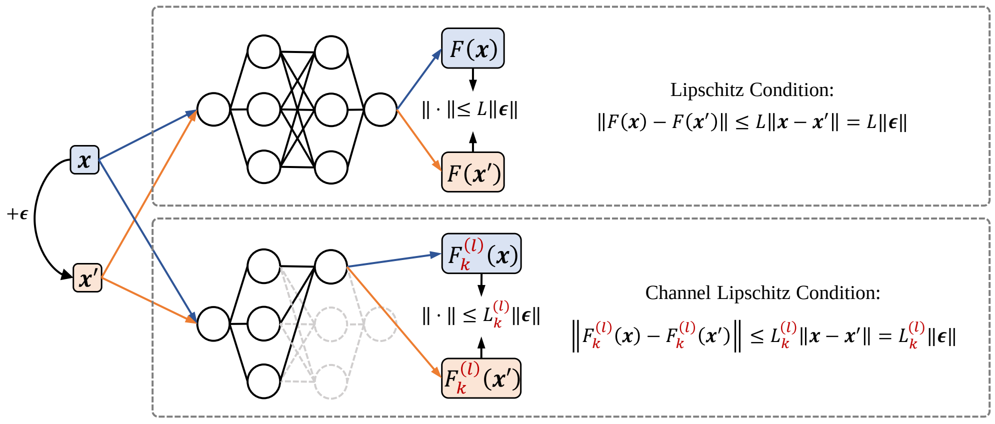
$F(x)$
$\|\cdot\| \leq L\|\epsilon\|$
$F(x')$
Lipschitz Condition:
$\|F(x) - F(x')\| \leq L\|x - x'\| = L\|\epsilon\|$
$x$
$+\epsilon$
$x'$
$F_k^{(l)}(x)$
$\|\cdot\| \leq L_k^{(l)}\|\epsilon\|$
$F_k^{(l)}(x')$
Channel Lipschitz Condition:
$\left\|F_k^{(l)}(x) - F_k^{(l)}(x')\right\| \leq L_k^{(l)}\|x - x'\| = L_k^{(l)}\|\epsilon\|$

Figure 1: An illustration of the differences between the commonly studied Lipschitz condition and the proposed channel Lipschitz condition. We highlight the differences in the formula with red color. In a nutshell, Lipschitz constant bounds the largest changing rate of the whole network function, while channel Lipschitz constant bounds that of a specific output channel.

well-designed attack mechanism, especially under high safety-required settings. Different from adversarial attacks, backdoor attacks modify the models by a carefully designed strategy during the model training process. For example, the attackers may inject a small proportion of malicious training samples with trigger patterns before training. In this way, DNN can be manipulated to have designated responses to inputs with specific triggers, while acting normally on benign samples [6].

Backdoor attacks may happen in different stages of the model adopting pipeline [7]. For example, when the victims try to train a DNN model on a suspicious third-party dataset, the implanted poisoned backdoor data perform the backdoor attack silently. The defender under such scenario thus has full access to the training process and the whole training data. Besides, backdoor attack happens with a greater frequency when the training process is uncontrollable by the victim, such as adopting third-party models or training on a third-party platform. Under such setting, the defender is only given a pre-trained infected model and usually a small set of benign data as an additional auxiliary to help to mitigate the backdoored model.

To better understand the mechanism of backdoor attacks, in this work, we revisit the common characteristics of backdoor attacks, *i.e.*, the ability of a small trigger to flip the outputs of benign inputs into the malicious target label. It is natural to relate such sensitivity to the high Lipschitz constant of the model. However, it may be a too strict condition for controlling the Lipschitz constant of the whole network function to achieve backdoor robustness. Since the backdoor vulnerability may come from the sensitivity of the channels to the backdoor trigger, we instead consider evaluating the Lipschitz constant (strictly speaking, an upper bound of the Lipschitz constant) among the channels to identify and prune the sensitive channels.

Specifically, we view the mapping from the input images to each channel output as an independent function, as shown in Figure 1. The Lipschitz constant is considered for each channel, which measures the sensitivity of the channel to the input. As channels, which detect backdoor triggers, should be sensitive to specific perturbation (*i.e.*, the trigger) on inputs, **we argue that backdoor related channels should have a high Lipschitz constant compared to normal channels**. To demonstrate this, we track the forward propagation process of the same inputs with and without trigger to see how the trigger changes the activation on these channels. We provide empirical evidence to show the strong correlation between the Channel Lipschitz Constant (CLC) and the trigger-activated changes. To be more specific, we show that channels with large activation changes after attaching the trigger usually have high Lipschitz constant. Intuitively, pruning those channels may mitigate the changes brought by the backdoor trigger.

To this end, we propose a novel *Channel Lipschitzness based Pruning* (CLP), which prunes the channels with high Lipschitz constant to recover the model. Since the CLC can be directly derived from the weight matrices of the model, this method requires no access to any training data. Unlike previous methods that are designed for different specific threat models, **CLP's data-free property ensures a high degree of flexibility for its practical application, as long as the model parameters are accessible**. Besides, CLP is super fast and robust to the choice of its only hyperparameter, *i.e.*, the relative pruning threshold $u$. Finally, we show that CLP can effectively reduce the attack success rate (ASR) against different advanced attacks with only a negligible drop on the clean accuracy.

To summarize, our contributions are twofold:

1. We innovatively reveal the connections between backdoor behaviors and the channels with high Lipschitz constants. This conclusion generalizes various backdoor attacks, shedding light on the development of backdoor defense.

2. Inspired by the above observations, we propose the CLP, a data-free backdoor removal method. Even without data, it achieves state-of-the-art (SOTA) performance among existing defense methods which require a certain amount of benign training data.

## 2 Related work

### 2.1 Backdoor Attack

BadNets [6] is one of the earliest researches that perform backdoor attacks in DNNs. They injected several kinds of image pattern, referred to as the *triggers*, into some samples, and modified the labels of the these samples to the desired malicious label. Then the DNNs were trained with the poisoned dataset and will be implanted a backdoor. After that, more covert and advanced backdoor designs were proposed [8, 9, 10]. To prevent defenders from reversely generating possible triggers through the model, dynamic backdoor attacks such as the Input-Aware Backdoor (IAB) [11], Warping-based Backdoor (WaNet) [12] and Sample Specific Backdoor Attack (SSBA) [13] were proposed. They generate a unique trigger for every single input, which makes the defense even more difficult. These attacks can be classified as the poisoning based backdoor attacks.

Under some settings, it is also possible for the attackers to directly modify the architectures and parameters of a DNN, without poisoning any training data, known as the *non-poisoning based backdoor attacks*. For example, in Trojan attack [14], the triggers were optimized to increase the activation of some carefully chosen neurons, and then the network was retrained to finish the poisoning. In target bit trojan [15] and ProFlip [16], the authors choose vulnerable neurons and apply bit-flip techniques to perform backdoor attacks. Note that, such bit flip attack only occurs in the deployment phase of DNNs, and the ASR does not exceed other traditional ones. Therefore, this attack will not be discussed in this paper.

### 2.2 Backdoor Defense

#### 2.2.1 Training Stage Defenses.

The *training stage defenses* aim at suppressing the effect of implanted backdoor triggers or filtering out the poisoned data during the training stage, with a full access to the training process. According to the different distributions of poisoned data and clean data in feature space [17], several methods were proposed to filter out the poisoned data, including the robust statistics [18, 19], input perturbation techniques [20, 21] and semi-supervise training [17]. Besides, stronger data augmentation techniques [22] were proposed to suppress the effect of backdoor, such as the CutMix [23], CutOut [23] and MaxUp [24]. Besides, differential privacy [25] and randomized smoothing [26] provide some certificated methods to defend against backdoor attack.

#### 2.2.2 Model Post-processing Defenses.

The *model post-processing defenses* mainly focus on eliminating the backdoor threat in a suspicious DNN model. The first effective defense against backdoor attack with the combination of neuron pruning and fine-tuning was proposed in [27]. Inspired by the mechanism of fixed-trigger backdoor attack, Neural Cleanse (NC) [28] obtained the reversed engineering trigger and detoxify the model with such knowledge. Some other fine-tuning based defenses used knowledge distillation [29] in their pipeline, such as the backdoor knowledge distillation [30] and Neural Attention Distillation (NAD) [31]. Besides, the Mode Connectivity Repair [32] was also explored to eliminate the backdoor effect.

In addition to fine-tuning based defenses, the $L_{\text{inf}}$[‡] Neuron Pruning was proposed in [33] to filter out the neurons with high $L_{\text{inf}}$ in activations. Recently, the Adversarial Neuron Pruning (ANP) [34] detected and pruned sensitive neurons with adversarial perturbations, and achieve considerable defense performance. However, it needs careful tuning of hyperparameters and requires access to a certain number of benign data.

Unlike those model post-processing methods, the proposed CLP achieves superior defending performance without any benign data, and is robust to the choice of the **only** hyperparameter. Moreover, we hope that our work can enlighten

[‡]Refers to the infinite norm.

the study on the effectiveness of Lipschitz constant on backdoor learning, and provide a new perspective on backdoor attack and defense.

## 3 Preliminaries

### 3.1 Notations

In this paper, we consider a classification problem with $C$ classes. Suppose that $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N \subseteq \mathcal{X} \times \mathcal{Y}$ is the original training set, which contains $N$ i.i.d. sample images $x_i \in \mathbb{R}^{d_c \times d_h \times d_w}$ and the corresponding labels $y_i \in \{1, 2, ..., C\}$. Here, we denote by $d_c$, $d_h$ and $d_w$ the number of output and input channels, the height and the width of the image, respectively. It is clear that $d_c = 3$ for RGB images.

We consider a neural network $F(x;\theta)$ with $L$ layers:

$$F(x;\theta) = f^{(L)} \circ \phi \circ f^{(L-1)} \circ \cdots \circ \phi \circ f^{(1)},$$

where $f^{(l)}$ is the linear function (*e.g.,* convolution) in the $l^{th}$ layer with $1 \le l \le L$, $\phi$ is a non-linear activation function applied element wise. For simplicity, we denote $F(x;\theta)$ as $F(x)$ or $F$. Let $\mathcal{W}^{(l)} \in \mathbb{R}^{d_{c'}^{(l)} \times d_c^{(l)} \times d_h^{(l)} \times d_w^{(l)}}$ be the weight tensor of the $l^{th}$ convolutional layer, where $d_{c'}^{(l)}$, $d_c^{(l)}$, $d_h^{(l)}$ and $d_w^{(l)}$ are the number of output and input channels, the height and the width of the convolutional kernel, respectively. To do pruning, we apply a mask $\mathcal{M}^{(l)} \in \{0,1\}^{d_{c'}^{(l)} \times d_c^{(l)} \times d_h^{(l)} \times d_w^{(l)}}$ starting with $\mathcal{M}_k^{(l)} = \mathbf{1}_{d_c^{(l)} \times d_h^{(l)} \times d_w^{(l)}}$ in each layer, where $\mathbf{1}_{d_c^{(l)} \times d_h^{(l)} \times d_w^{(l)}}$ denotes an all-one tensor. Pruning neurons on the network refers to getting a collection of indices $\mathcal{I} = \{(l,k)_i\}_{i=1}^I$ and setting $\mathcal{M}_k^{(l)} = \mathbf{0}_{d_c^{(l)} \times d_h^{(l)} \times d_w^{(l)}}$ if $(l,k) \in \mathcal{I}$. The pruned network $F_{-\mathcal{I}}$ has the same architecture as $F$ with all the weight matrices of convolutional layers set to $\mathcal{W}^{(l)} \odot \mathcal{M}^{(l)}$, where $\odot$ denotes the Hadamard product [35].

The backdoor poisoning attack involves changing the input images and the corresponding labels[§] on a subset of the original training set $\mathcal{D}_p \subseteq \mathcal{D}$. We denote the poisoning function to the input images by $\delta(x)$. The ratio $\rho = \frac{|\mathcal{D}_p|}{|\mathcal{D}|}$ is defined as the *poisoning rate*.

### 3.2 $L$-Lipschitz Function

A function $g : \mathbb{R}^{n_1} \to \mathbb{R}^{n_2}$ is *L-Lipschitz continuous* [36] in $\mathcal{X} \subseteq \mathbb{R}^{n_1}$ under $p$-norm, if there exists a non-negative constant $L \ge 0$ such that

$$\|g(\boldsymbol{x}) - g(\boldsymbol{x}')\|_p \le L\|\boldsymbol{x} - \boldsymbol{x}'\|_p, \quad \forall\, \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}. \tag{1}$$

The smallest $L$ guaranteeing equation (1) is called the *Lipschitz constant* of $g$, denoted by $\|g\|_{\text{Lip}}$. For simplicity, we choose $p = 2$ in this paper. Viewing $\boldsymbol{x}'$ as a perturbation of $\boldsymbol{x}$, the Lipschitz constant $\|g\|_{\text{Lip}}$ can be regarded as the maximum ratio between the resulting perturbation in the output space and the source perturbation in the input space. Thus, it is commonly used for measuring the sensitivity of a function to the input perturbation.

### 3.3 Lipschitz Constant in Neural Networks

According to the Cauchy-Schwartz inequality, we are now able to control the Lipschitz constant of the whole network as follows:

$$\begin{aligned}\|F\|_{\text{Lip}} &= \|f^{(L)} \circ \phi \circ f^{(L-1)} \circ \cdots \circ \phi \circ f^{(1)}\|_{\text{Lip}} \\ &\le \|f^{(L)}\|_{\text{Lip}} \cdot \|\phi\|_{\text{Lip}} \cdot \|f^{(L-1)}\|_{\text{Lip}} \cdots \|\phi\|_{\text{Lip}} \cdot \|f^{(1)}\|_{\text{Lip}}.\end{aligned} \tag{2}$$

Most of the commonly used activation functions are L-Lipschitz (*e.g.,* ReLU, LeakyReLU, Sigmoid, Tanh, ELU, SeLU). In particular, we have $L = 1$ for the ReLU function, which is used in this paper. Note that $f^{(l)}(\boldsymbol{x}^{(l)}) = \boldsymbol{W}^{(l)}\boldsymbol{x}^{(l)} + \boldsymbol{b}^{(l)}$. It follows that

$$\|F\|_{\text{Lip}} \le \prod_{l=1}^{L} \|f^{(l)}\|_{\text{Lip}} = \prod_{l=1}^{L} \max_{\|\boldsymbol{z}\|_2 \ne 0} \frac{\|\boldsymbol{W}^{(l)}\boldsymbol{z}\|_2}{\|\boldsymbol{z}\|_2} = \prod_{l=1}^{L} \sigma(\boldsymbol{W}^{(l)}), \tag{3}$$

where $\sigma(\boldsymbol{W}^{(l)}) = \max_{\|\boldsymbol{z}\|_2 \ne 0} \frac{\|\boldsymbol{W}^{(l)}\boldsymbol{z}\|_2}{\|\boldsymbol{z}\|_2}$ is the spectral norm.

[§]the labels remain unchanged in clean label attacks [8].

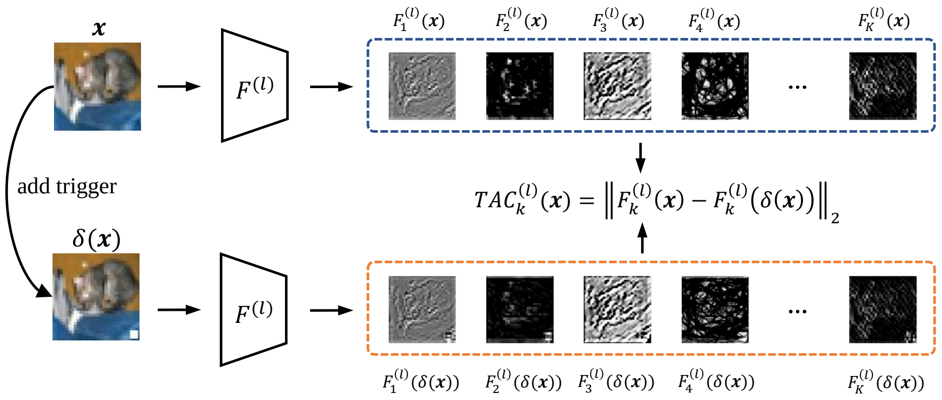

Figure 2: A simple diagram for calculating TAC in the $l^{th}$ layer. As illustrated, the word TAC stands for the activation differences of the feature maps before and after attaching the trigger to the images, and $k$ is the number of feature maps.

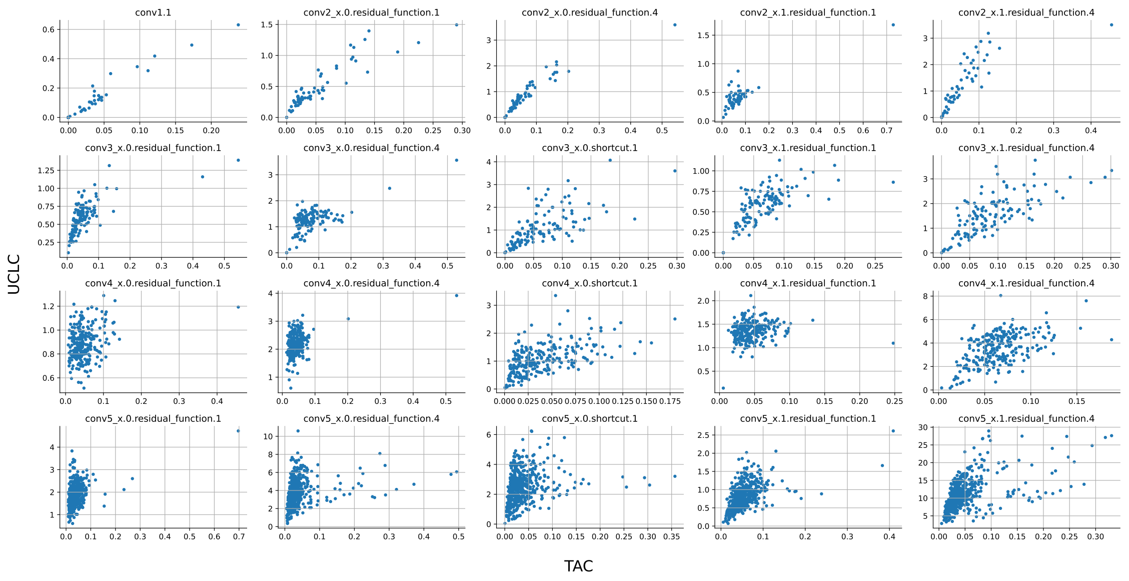

Figure 3: A scatter plot to demonstrate the relationship between UCLC and TAC. As shown in the figure, we observe a strong correlation between the two indices.

## 4 Methodology

### 4.1 Channel Lipschitz Constant

We denote the function from the input to the $k^{th}$ channel of $l^{th}$ layer by:

$$F_k^{(l)} = f_k^{(l)} \circ \phi \circ f^{(l-1)} \cdots \circ \phi \circ f^{(1)},$$

where $f_k^{(l)}$ is the $k^{th}$ output channel function of the $l^{th}$ layer. In particular, if

$$f^{(l)}(\boldsymbol{x}) = \boldsymbol{W}^{(l)}\boldsymbol{x} + \boldsymbol{b}^{(l)}, \quad \boldsymbol{W}^{(l)} \in \mathbb{R}^{d_{\text{out}}^{(l)} \times d_{\text{in}}^{(l)}},$$

then $f_k^{(l)}(\boldsymbol{x}) = \boldsymbol{w}_k^{(l)}\boldsymbol{x} + b_k^{(l)}$, where $\boldsymbol{w}_k^{(l)} \in \mathbb{R}^{1 \times d_{in}^{(l)}}$ is the $k^{th}$ row of $\boldsymbol{W}^{(l)}$. It follows that the *channel Lipschitz constant* (CLC) satisfies

$$\|F_k^{(l)}\|_{\text{Lip}} \leq \|\boldsymbol{w}_k^{(l)}\|_{\text{Lip}} \prod_{i=1}^{l-1} \sigma(\boldsymbol{W}^{(i)}).$$

We refer to the right side of the above inequality as an *Upper bound* of the Channel Lipschitz Constant (UCLC).

Particularly, convolution is a special linear transformation, with the weight matrix $\boldsymbol{W}^{(l)}$ being a doubly block-Toeplitz (DBT) form of the weight tensor $\mathcal{W}^{(l)}$. To calculate the exact spectral norm of a convolution operation, one should first convert the weight tensor into a DBT weight matrix. However, calculating the DBT matrix is time-consuming and memory-expensive, especially when the number of channels is large. A much simpler alternative way for this is to reshape the weight tensor into a matrix form and use the spectral norm of the reshaped matrix as an approximation to the original spectral norm. This approximation has been adopted by previous researches [37]. In our work, for simplicity, we calculate the spectral norm using the reshaped kernel matrix $W_k^{(l)} \in \mathbb{R}^{d_c^{(l)} \times (d_h^{(l)} d_w^{(l)})}$, which also shows acceptable results in our experiments.

## 4.2 Trigger-activated Change

In order to study the extent to which these channels are related to the backdoor behaviors, we define the *Trigger-activated Change* (TAC). Specifically, we first train a ResNet-18 with BadNets in CIFAR-10, and track the forward propagation process of the same inputs with and without trigger. TAC is defined as the average differences of the $k^{th}$ channel of the $l^{th}$ layer for test samples $\boldsymbol{x}$ in dataset $\mathcal{D}$:

$$TAC_k^{(l)}(\mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{\boldsymbol{x} \in \mathcal{D}} \|F_k^{(l)}(\boldsymbol{x}) - F_k^{(l)}(\delta(\boldsymbol{x}))\|_2,$$

where $\delta(\cdot)$ is the poisoning function. A more detailed illustration of this quantity is shown in Figure 2. TAC is the change of the activation before and after the input image is attached with a trigger. Its magnitude reflects the effect the trigger has on a given channel. Higher TAC indicate higher sensitivity of the channel to the trigger. Note that TAC is proposed to study the backdoor behavior, but it can not used for defensing. This is because the calculation of TAC requires the access to the trigger pattern, which defenders cannot get in general.

## 4.3 Correlation between CLC and TAC

A scatter plot chart of TAC and UCLC for each layer under BadNets (All to One) [6] is shown in Figure 3, from which we can observe a high correlation between them. In particular, there are some outlying channels with extremely high TAC in some layers, indicating that they are sensitive to the trigger. Hence, it is reasonable to consider them as the potential backdoor channels. As expected, most of these channels have abnormally high UCLC. Remind that TAC is inaccessible for a defender, but UCLC can be directly calculated from the weight matrices of the given model. Hence, we use UCLC as an alternated index to detect potential backdoor channels. We will show that pruning those high-UCLC channels will significantly reduce the backdoor ASR in Section 5.

## 4.4 Special Case in CNN with Batch Normalization

The *Batch Normalization* (BN) [38] is adopted to stabilize the training process and make optimization landscape much smoother in modern neural networks. It normalizes the batch inputs of each channel and adjusts the mean and variance through trainable parameters. BN is usually placed after the convolution and before the non-linear function. Note that BN is also a linear transformation, and can be merged with convolution into a matrix-matrix multiplication. In this paper, we view a Conv-BN block as one linear layer, and an UCLC is calculated based on the composed weight matrix.

## 4.5 Channel Lipschitzness based Pruning

Based on the above observations, it is natural to think of removing the high-UCLC channels to recover the model. On this basis, we propose the channel Lipschitzness based pruning (CLP), which calculates UCLC for channels in each layer, and prunes the channels with UCLC larger than a pre-defined threshold within the layer. Note that in the same layer, all channels share the same cumulative product term, which is the Lipschitz upper bound of the previous layers. Hence, a much simplified way to compare CLC within a particular layer is to directly compare $\sigma(\boldsymbol{W}_k^{(l)})$. The overall algorithm procedure is shown in Algorithm Line 12.

Determining an appropriate threshold is crucial to the performance of this method. In this work, we simply set the threshold for the $l^{th}$ layer as $\mu^{(l)} + u * s^{(l)}$, where $\mu^{(l)} = \frac{1}{K}\sum_{i=1}^{K} \sigma(\boldsymbol{W}_k^{(l)})$ and $s^{(l)} = \sqrt{\frac{1}{K}\sum_{i=1}^{K}(\sigma(\boldsymbol{W}_k^{(l)}) - \mu^{(l)})^2}$ are the mean and the standard deviation of the $l^{th}$ layer indices set $\{\sigma(\boldsymbol{W}_k^{(l)}) : k = 1, 2, \ldots K\}$, and $u$ is the only hyperparameter for the CLP. The above threshold is also known as the common outlier dividing line for a standard Gaussian distribution. We find this simple setting works well in our experiments.

**Algorithm 1:** Channel Lipschitzness based Pruning

```
Input: L layer neural network function $F^{(L)}$ with a set of convolution weight tensor $\{\mathcal{W}^{(l)} : l = 1, 2, \ldots, L\}$, threshold hyperparameter $u$
Output: A pruned network
1  $\mathcal{I} := \emptyset$
2  for l = 1 to L do
3      for k = 1 to K do
4          $\boldsymbol{W}_k^{(l)} := \mathrm{ReshapeToMatrix}(\mathcal{W}_k^{(l)})$
5          $\sigma_k^{(l)} := \sigma(\boldsymbol{W}_k^{(l)})$
6      end
7      $\mu^{(l)} := \frac{1}{K}\sum_{i=1}^{K}\sigma_k^{(l)}$
8      $s^{(l)} := \sqrt{\frac{1}{K}\sum_{i=1}^{K}(\sigma_k^{(l)} - \mu^{(l)})^2}$
9      $\mathcal{I}^{(l)} := \{(l, k) : \sigma_k^{(l)} > \mu^{(l)} + u * s^{(l)}\}$
10     $\mathcal{I} = \mathcal{I} \cup \mathcal{I}^{(l)}$
11 end
12 return $F_{-\mathcal{I}}^{(L)}$
```

## 5 Experiments

### 5.1 Experimental Settings

#### 5.1.1 Attack Settings.

We test our proposed CLP on a variety of famous attack methods, *i.e.*, BadNets [6], Clean Label Attack [8], Trojan Attack [14], Blended Backdoor Attack [10], WaNet [12], IAB [11] and Sample Specific Backdoor Attack [13]. The attacks are performed on CIFAR-10 [39] and Tiny ImageNet [40] using ResNet-18 [41]. For BadNets, we test both All-to-All (BadA2A) attack and All-to-One (BadA2O) attack, which means that the attack target labels $y_t$ are set to all labels by $y_t = (y + 1)\%C$ (% denotes modulo operation) or one particular label $y_t = C_t$. Due to the image size requirement of the SSBA, its corresponding experiments are only conducted on Tiny ImageNet. We use $\sim 95\%$ of the training data to train the backdoored model. The rest $5\%$ are split into $4\%$ of validation data, and $1\%$ of benign training data to perform other defenses. The trigger size is $3 \times 3$ for CIFAR-10 and $5 \times 5$ for Tiny ImageNet. The poison label is set to the $0^{th}$ class, and the poisoning rate is set to $10\%$ by default. We use SGD [42] as the base optimizer, and train the backdoored model with learning rate 0.1, momentum 0.9 and batch size 128 for 150 epochs on CIFAR-10, batch size 64 for 100 epochs on Tiny ImageNet. We use cosine scheduler to adjust the learning rate. All the experiments are conducted on Pytorch [43] framework.

#### 5.1.2 Defense Settings.

We compare our approaches with the commonly used model repairing methods, *i.e.*, FT, FP [27], NAD [31] and the SOTA neuron pruning strategy ANP [34]. All defense methods are allowed to access $1\%$ of the benign training data. Note that **no data** are used in CLP. The fine-tuning based methods default the training process with batch size 128 and learning rate 0.005 for 20 epochs. We adjust the hyperparameters including pruning ratio in fine-pruning [27], attentional weight in NAD [31], and pruning threshold in ANP [34] to obtain their best performance instructed by their original papers. The CLP hyperparameter $u$ is default to be 3 on CIFAR-10 and 5 on Tiny ImageNet. Further study on the effect of the hyperparameter is conducted in Section 5.3.

#### 5.1.3 Evaluation Metric.

The evaluation of the model includes the performance on benign data, Accuracy on Clean data (ACC) and the performance on the backdoored data, which we call the attack success rate (ASR). Note that the ASR is the ratio for poisoned samples that are **misclassified** as the target label, and it is calculated using the backdoored samples whose ground-truth labels do not belong to the target attack class. In a nutshell, a successful defense should achieve low ASR without much degradation on the ACC.

### 5.2 Experimental Results

In this section, we verify the effectiveness of CLP and compare its performance with other 4 existing model repairing methods as shown in Table 1 and Table 2. Table 1 shows the experimental results on CIFAR-10, and the proposed CLP remarkably achieves almost the highest robustness against several advanced backdoor attacks. To be specific, the proposed CLP successfully cut the average ASR down to $2.81\%$ with only a slight drop on the ACC ($0.67\%$ on

Table 1: Performance evaluation of the proposed CLP without data and 4 other defense methods with 500 benign data against seven mainstream attacks on CIFAR-10 with ResNet-18. Higher ACC and Lower ASR are preferable, and the best results are boldfaced. ↓ denotes the drop rate on average.

| Trigger Type | Attacks | Backdoored | | FT | | FP[27] | | NAD[31] | | ANP[34] | | CLP(ours) | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | ACC | ASR | ACC | ASR | ACC | ASR | ACC | ASR | ACC | ASR | ACC | ASR |
| Static | BadA2O [6] | 93.86 | 100.00 | 92.22 | 2.16 | 92.18 | 2.97 | 91.67 | 5.40 | 91.67 | 5.40 | **93.46** | **1.38** |
| | BadA2A | 94.60 | 93.89 | 92.03 | 60.76 | 91.75 | 66.82 | 92.86 | 1.33 | 90.29 | 86.22 | **93.69** | **1.06** |
| | Trojan [14] | 94.06 | 100.00 | 92.58 | 99.99 | 90.78 | 86.43 | 92.13 | 5.76 | 93.44 | 8.11 | **93.54** | **2.06** |
| | CLA [8] | 93.14 | 100.00 | 91.86 | 0.39 | 91.02 | 93.21 | **92.46** | **0.44** | 91.13 | 11.76 | 91.89 | 2.84 |
| | Blended [44] | 94.17 | 99.62 | 93.90 | 70.27 | 90.92 | 3.24 | 92.72 | **1.61** | 93.66 | 5.03 | **94.07** | 1.90 |
| Dynamic | IAB [11] | 93.87 | 97.91 | 91.78 | **9.52** | 87.04 | 21.33 | 93.52 | 10.61 | **93.52** | 10.61 | 92.78 | 9.88 |
| | WaNet [12] | 94.50 | 99.67 | 92.93 | 9.37 | 92.07 | 1.03 | 94.12 | 0.51 | **94.12** | **0.51** | 94.06 | 0.56 |
| | Average | 94.03 | 98.72 | 92.47 | 36.07 | 90.82 | 39.29 | 92.78 | 4.30 | 92.54 | 18.23 | **93.36** | **2.81** |
| | Drop | ↓ 0.00 | ↓ 0.00 | ↓ 1.56 | ↓ 62.66 | ↓ 3.21 | ↓ 59.43 | ↓ 1.25 | ↓ 94.42 | ↓ 1.49 | ↓ 80.49 | ↓ **0.67** | ↓ **95.91** |

Table 2: Performance evaluation of the proposed CLP without data and 4 other defense methods with 1,000 benign data against seven mainstream attacks on Tiny ImageNet with ResNet-18. Higher ACC and Lower ASR are preferable, and the best results are boldfaced. ↓ denotes the drop rate on average.

| Trigger Type | Attacks | Backdoored | | FT | | FP[27] | | NAD[31] | | ANP[34] | | CLP(ours) | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | ACC | ASR | ACC | ASR | ACC | ASR | ACC | ASR | ACC | ASR | ACC | ASR |
| Static | BadA2O [6] | 62.99 | 99.89 | 56.97 | 99.26 | 57.43 | 57.42 | 61.63 | 0.85 | **63.05** | 3.93 | 62.94 | **0.61** |
| | Trojan [14] | 64.09 | 99.99 | 62.85 | 3.45 | 60.43 | 99.59 | 61.12 | 95.05 | 62.68 | 10.43 | **63.86** | **0.77** |
| | CLA [8] | 64.94 | 84.74 | 53.59 | 27.32 | 61.18 | 82.72 | 59.75 | 30.65 | 60.98 | 15.69 | **64.71** | **0.41** |
| | Blended [44] | 63.30 | 99.70 | 59.98 | 1.02 | 59.84 | 62.17 | 61.94 | 11.55 | 62.49 | **0.61** | **63.12** | 0.74 |
| Dynamic | IAB [11] | **61.40** | 98.28 | 58.35 | 89.89 | 57.03 | **0.21** | 58.17 | 68.43 | 61.39 | 4.67 | 59.09 | 8.70 |
| | WaNet [12] | 60.76 | 99.92 | 57.96 | 97.45 | 53.86 | 23.70 | 56.42 | 87.79 | 54.82 | 86.98 | **59.52** | **1.57** |
| | SSBA [13] | 66.51 | 99.78 | 62.66 | 73.11 | 62.89 | 4.68 | 60.13 | 24.68 | 60.98 | 1.01 | **63.49** | **0.42** |
| | Average | 63.43 | 97.47 | 58.91 | 55.93 | 58.95 | 47.21 | 59.88 | 62.13 | 60.91 | 17.62 | **62.39** | **1.89** |
| | Drop | ↓ 0.00 | ↓ 0.00 | ↓ 4.52 | ↓ 41.54 | ↓ 4.48 | ↓ 50.26 | ↓ 6.76 | ↓ 45.57 | ↓ 2.52 | ↓ 79.85 | ↓ **1.04** | ↓ **95.58** |

average). Note that CLP reaches such incredible result with no data requirement, and the SOTA defenses ANP and NAD give a similar performance on the ASR with a larger trade-off on the ACC under the access to benign data.

The standard fine-tuning provides promising defense results against several attacks, especially BadNets (A2O), but fails to generalize to more complex attacks such as Trojan, BadNets (A2A) and blended attack. NAD repairs the backdoored model based on knowledge distillation with supporting information from the attention map of a fine-tuned model. Though it achieves relatively good defense performance, it requires carefully tuning of the hyperparameters.

As for the other pruning based methods, fine-pruning adds an extra neuron pruning step according to the neuron activation to benign images before fine-tuning the model, and achieves the best defense performance against CLA. However, it also fails to maintain high robustness against some more covert attacks. ANP utilizes an adversarial strategy to find and prune the neurons that are sensitive to perturbations, which are considered to be highly backdoor related. While both ANP and CLP leverage the concept of sensitivity on channels, ANP measures the sensitivity of the model output to the perturbation on the channels as the pruning index, which requires additional data and careful hyperparameter tuning for different attacks. Unlike ANP, our CLP prunes channels based on the sensitivity of the channels to the inputs, which comes closer to the essence of backdoor attack and can be directly induced by the properties of the model weights without any data. We find both strategy works well on CIFAR-10 against various attacks, but on average, CLP performs better.

Table 2 shows more experimental results on Tiny ImageNet. All the compared defense methods suffer from severe degradation on both the ACC and the ASR when confront a larger-scale dataset. On the contrast, our CLP still maintains its robustness against those different attacks, including SOTA sample specific attacks IAB and SSBA, which further suggests the strong correlation between channel Lipschitzness and the backdoor behaviors.

## 5.3 Ablation Studies

### 5.3.1 Performance with Different Choices of Hyperparameter $u$.

As mentioned in Section 4.5, the CLP hyperparameter $u$ controls the trade-off between test accuracy on clean data and robust against backdoor attack. Figure 4 shows the ACC and the ASR of the backdoored model after applying CLP with different hyperparameter $u$. From the fact that ASR drops rapidly to near $0\%$ while ACC drops much later as $u$ decreases, we argue that the backdoor related channels usually possess higher UCLC than normal channels, which will be pruned precisely by the CLP. Generally speaking, we can regard the interval between the two points when ASR drops to a very low level and ACC starts to drop as an index of the robustness with hyperparameter. For example, it is much easier to choose the hyperparameter $u$ to defend against Blended attack [44] because choosing $u \in [3, 5]$ will not affect that much on the performance. CLA has the smallest gap, and it requires a more carefully chosen hyperparameter.

A possible reason is that the UCLC in the CLA attacked model is not that high compared with other attacked models. Nevertheless, setting $u = 3$ still has an acceptable performance on CLA. Note that when $u$ decreases near to 0, nearly all the channels in the model are pruned, so the prediction of the model can be illogical. That's why the ASR curves in some attacks rapidly increase to $100\%$ when $u$ decreases to 0.

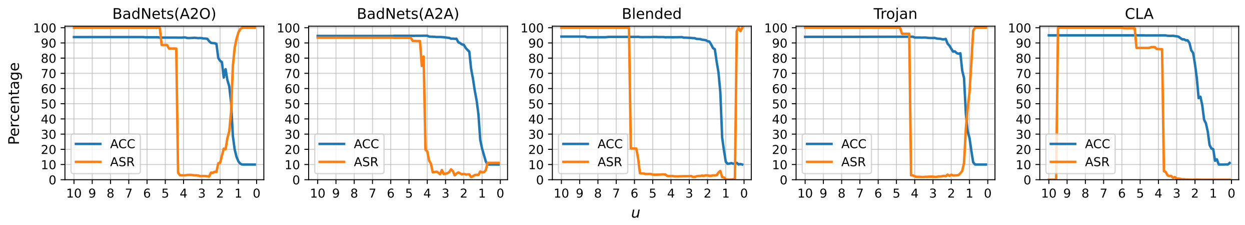

Figure 4: Performance of the CLP with different hyperparameter against different attacks on CIFAR-10 with ResNet-18.

### 5.3.2 Performance on Different Architectures.

To verify the generalization performance of CLP across different architectures, we perform BadNets attack on CIFAR-10 with the commonly used CNN architectures ResNet [41], VGG [45] and SENet [46] of different depths. Then we plot the ACC and the ASR curves *w.r.t* to the hyperparameter $u$. This is shown in Figure 5. Overall, CLP achieves very good results on all the tested CNN architectures. Nevertheless, the optimal $u$ for different architectures are different. For example, the optimal choice of $u$ in VGG-16 is about 9. However, such choice of $u$ will not work on other architectures. In addition, we find that both VGG architectures and SENet architectures show better robustness to the choice of hyperparameter than ResNet architectures. In general, choosing $u$ between 3 and 4 generalizes well on different architectures.

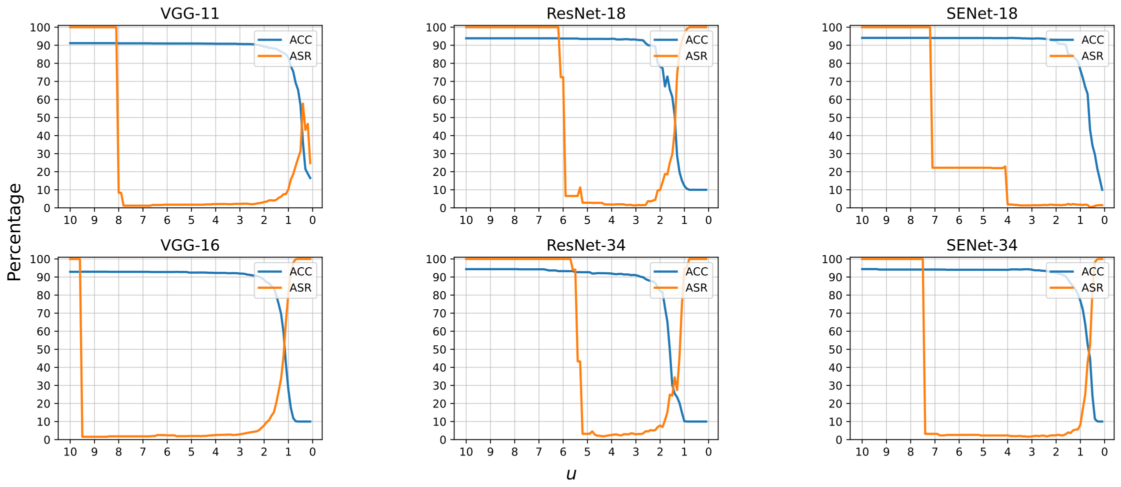

Figure 5: Performance of the CLP with different hyperparameter $u$ in variant CNN architectures against BadNets on CIFAR-10.

Table 3: Performance of the CLP against typical backdoor attacks with different poisoning rate on CIFAR-10 with ResNet-18.

| Poisoning rate | Model | BadNets (A2O) | | BadNets (A2A) | | Trojan [14] | | CLA[8] | | Blended[44] | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | ACC | ASR | ACC | ASR | ACC | ASR | ACC | ASR | ACC | ASR |
| 1% | Backdoored | 93.86 | 100.00 | 94.60 | 93.47 | 94.06 | 100.00 | 93.14 | 100.00 | 94.17 | 100.00 |
| | CLP Pruned | 93.46 | 1.38 | 93.69 | 1.06 | 93.54 | 0.92 | 91.89 | 2.84 | 94.07 | 1.90 |
| 5% | Backdoored | 94.29 | 100.00 | 94.21 | 92.57 | 94.48 | 100.00 | 94.46 | 99.86 | 94.33 | 100.00 |
| | CLP Pruned | 92.33 | 6.42 | 93.93 | 0.76 | 93.84 | 5.56 | 90.71 | 5.96 | 93.37 | 2.34 |
| 10% | Backdoored | 95.03 | 100.00 | 94.75 | 90.77 | 94.79 | 100.00 | 94.99 | 98.83 | 94.60 | 83.63 |
| | CLP Pruned | 94.21 | 2.76 | 94.03 | 0.74 | 93.17 | 3.32 | 93.67 | 10.04 | 93.23 | 0.83 |

#### 5.3.3 Performance under Different Poisoning Rates.

The different choice of poisoning rate also affects the defense performance. To study the robustness of the proposed CLP, we try different poisoning rate of different backdoor attacks with the hyperparameter unchanged ($u = 3$ on CIFAR-10 by default). As shown in Table 3, CLP effectively reduces the ASR and maintains high ACC under different settings. We note that decreasing the poisoning rate of CLA leads to increasing ASR after applying CLP. CLA with poisoning rate set to $1\%$ gives the worse defense results, and the ASR is about 10%. However, such ASR doesn't give a considerable threat to our models. Overall, we find poisoning rate doesn't affect much on the performance of CLP.

#### 5.3.4 Running Time Comparison.

We record the running time of the above-mentioned defense methods on 500 CIFAR-10 images with ResNet-18, and show the results in Table 4. The proposed CLP is the only one, which do not require data propagation, so CPU is enough to apply CLP, and we only record the CPU time (i7-5930K CPU @ 3.50GHz). The other methods are evaluated on RTX 2080Ti GPU. The proposed CLP is almost five times faster than the fastest methods among them and only requires 2.87 seconds.

Table 4: The overall running time of different defense methods on 500 CIFAR-10 images with ResNet-18. All the methods except for CLP are training on GPU. * denotes that the results of CLP is in CPU time.

| Defense Methods | FT | FP | NAD | ANP | CLP* |
|---|---|---|---|---|---|
| Runing Time (sec.) | 12.35s | 14.59s | 25.68s | 22.08s | **2.87s** |

## 6 Conclusions

In this paper, we reveal the connection between the Lipschitzness and the backdoor behaviors of the channels in an infected DNN. On this basis, we calculate an upper bound of the channel Lipschitz constant (UCLC) for each channel, and prune the channels with abnormally high UCLC to recover the model, which we refer to as the Channel Lipschitzness based Pruning (CLP). Due to the fact that UCLC can be directly induced by the model weights, our CLP does not require any data and runs super fast. To the best of our knowledge, CLP is the first productive data-free backdoor defense method. Extensive experiments show that the proposed CLP can efficiently and effectively remove the injected backdoor while maintaining high ACC against various SOTA attacks. Finally, ablation studies show that CLP is robust to the **only** hyperparameter $u$, and generalizes well to different CNN architectures. Our work further shows the effectiveness of channel pruning in defense of backdoor attacks. More importantly, it sheds light on the relationship between the sensitive nature of backdoor and the channel Lipschitz constant, which may help to explain the backdoor behaviors in neural networks.

## 7 Acknowledgement

This work was supported in part by the National Natural Science Foundation of China (No. 62101351), the GuangDong Basic and Applied Basic Research Foundation (No.2020A1515110376), and the Shenzhen Outstanding Scientific and Technological Innovation Talents PhD Startup Project (No. RCBS20210609104447108).

## References

[1] Li Liu, Gang Feng, Denis Beautemps, and Xiao-Ping Zhang. Re-synchronization using the hand preceding model for multi-modal fusion in automatic continuous cued speech recognition. *IEEE Transactions on Multimedia*, 23:292–305, 2020.

[2] Yixiong Chen, Chunhui Zhang, Li Liu, Cheng Feng, Changfeng Dong, Yongfang Luo, and Xiang Wan. Uscl: Pretraining deep ultrasound image diagnosis model through video contrastive representation learning. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 627–637. Springer, 2021.

[3] F Tramèr, D Boneh, A Kurakin, I Goodfellow, N Papernot, and P McDaniel. Ensemble adversarial training: Attacks and defenses. In *6th International Conference on Learning Representations, ICLR 2018-Conference Track Proceedings*, 2018.

[4] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[5] Battista Biggio, Blaine Nelson, and Pavel Laskov. Support vector machines under adversarial label noise. In *Asian conference on machine learning*, pages 97–112. PMLR, 2011.

[6] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7:47230–47244, 2019.

[7] Yiming Li, Baoyuan Wu, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Backdoor learning: A survey. *arXiv preprint arXiv:2007.08745*, 2020.

[8] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Label-consistent backdoor attacks. *arXiv preprint arXiv:1912.02771*, 2019.

[9] Mingfu Xue, Can He, Jian Wang, and Weiqiang Liu. One-to-n & n-to-one: Two advanced backdoor attacks against deep learning models. *IEEE Transactions on Dependable and Secure Computing*, 2020.

[10] Yunfei Liu, Xingjun Ma, James Bailey, and Feng Lu. Reflection backdoor: A natural backdoor attack on deep neural networks. In *European Conference on Computer Vision*, pages 182–199. Springer, 2020.

[11] Tuan Anh Nguyen and Anh Tran. Input-aware dynamic backdoor attack. *Advances in Neural Information Processing Systems*, 33:3454–3464, 2020.

[12] Anh Nguyen and Anh Tran. Wanet–imperceptible warping-based backdoor attack. *arXiv preprint arXiv:2102.10369*, 2021.

[13] Yuezun Li, Yiming Li, Baoyuan Wu, Longkang Li, Ran He, and Siwei Lyu. Invisible backdoor attack with sample-specific triggers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16463–16472, 2021.

[14] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018*. The Internet Society, 2018.

[15] Adnan Siraj Rakin, Zhezhi He, and Deliang Fan. Tbt: Targeted neural network attack with bit trojan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13198–13207, 2020.

[16] Huili Chen, Cheng Fu, Jishen Zhao, and Farinaz Koushanfar. Proflip: Targeted trojan attack with progressive bit flips. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7718–7727, 2021.

[17] Kunzhe Huang, Yiming Li, Baoyuan Wu, Zhan Qin, and Kui Ren. Backdoor defense via decoupling the training process. In *International Conference on Learning Representations*, 2021.

[18] Jonathan Hayase, Weihao Kong, Raghav Somani, and Sewoong Oh. Spectre: defending against backdoor attacks using robust statistics. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 4129–4139. PMLR, 18–24 Jul 2021.

[19] Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 8011–8021, 2018.

[20] Yansong Gao, Change Xu, Derui Wang, Shiping Chen, Damith C Ranasinghe, and Surya Nepal. Strip: A defence against trojan attacks on deep neural networks. In *Proceedings of the 35th Annual Computer Security Applications Conference*, pages 113–125, 2019.

[21] Bao Gia Doan, Ehsan Abbasnejad, and Damith C Ranasinghe. Februus: Input purification defense against trojan attacks on deep neural network systems. In *Annual Computer Security Applications Conference*, pages 897–912, 2020.

[22] Eitan Borgnia, Valeriia Cherepanova, Liam Fowl, Amin Ghiasi, Jonas Geiping, Micah Goldblum, Tom Goldstein, and Arjun Gupta. Strong data augmentation sanitizes poisoning and backdoor attacks without an accuracy tradeoff. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3855–3859. IEEE, 2021.

[23] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[24] Chengyue Gong, Tongzheng Ren, Mao Ye, and Qiang Liu. Maxup: A simple way to improve generalization of neural network training. *arXiv preprint arXiv:2002.09024*, 2020.

[25] Min Du, Ruoxi Jia, and Dawn Song. Robust anomaly detection and backdoor attack detection via differential privacy. In *International Conference on Learning Representations*, 2019.

[26] Elan Rosenfeld, Ezra Winston, Pradeep Ravikumar, and Zico Kolter. Certified robustness to label-flipping attacks via randomized smoothing. In *International Conference on Machine Learning*, pages 8230–8241. PMLR, 2020.

[27] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pages 273–294. Springer, 2018.

[28] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 707–723. IEEE, 2019.

[29] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[30] Kota Yoshida and Takeshi Fujino. Disabling backdoor and identifying poison data by using knowledge distillation in backdoor attacks on deep neural networks. In *Proceedings of the 13th ACM Workshop on Artificial Intelligence and Security*, pages 117–127, 2020.

[31] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. Neural attention distillation: Erasing backdoor triggers from deep neural networks. In *International Conference on Learning Representations*, 2020.

[32] Pu Zhao, Pin-Yu Chen, Payel Das, Karthikeyan Natesan Ramamurthy, and Xue Lin. Bridging mode connectivity in loss landscapes and adversarial robustness. In *International Conference on Learning Representations*, 2019.

[33] Kaidi Xu, Sijia Liu, Pin-Yu Chen, Pu Zhao, and Xue Lin. Defending against backdoor attack on deep neural networks. *arXiv preprint arXiv:2002.12162*, 2020.

[34] Dongxian Wu and Yisen Wang. Adversarial neuron pruning purifies backdoored deep models. *Advances in Neural Information Processing Systems*, 34, 2021.

[35] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

[36] Larry Armijo. Minimization of functions having lipschitz continuous first partial derivatives. *Pacific Journal of mathematics*, 16(1):1–3, 1966.

[37] Yuichi Yoshida and Takeru Miyato. Spectral norm regularization for improving the generalizability of deep learning. *arXiv preprint arXiv:1705.10941*, 2017.

[38] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[39] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[40] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

[41] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[42] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[43] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[44] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

[45] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[46] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.