RETHINKING THE TRIGGER OF BACKDOOR ATTACK

Anonymous authors
Paper under double-blind review

ABSTRACT

Backdoor attack intends to inject hidden backdoor into the deep neural networks (DNNs), such that the prediction of the infected model will be maliciously changed if the hidden backdoor is activated by the attacker-defined trigger, while it performs well on benign samples. Currently, most of existing backdoor attacks adopted the setting of *static* trigger, $i.e.$, triggers across the training and testing images follow the same appearance and are located in the same area. In this paper, we revisit this attack paradigm by analyzing the characteristics of the static trigger. We demonstrate that such an attack paradigm is vulnerable when the trigger in testing images is not consistent with the one used for training. We further explore how to utilize this property for backdoor defense, and discuss how to alleviate such vulnerability of existing attacks. We hope that this work could inspire more explorations on backdoor properties, to help the design of more advanced backdoor defense and attack methods.

1 INTRODUCTION

Deep neural networks (DNNs) have demonstrated their superior performance in a variety of applications. However, DNNs have been proved to be unstable that the small perturbation on the input may lead to a significant change in the output, which raises serious security concerns. For example, given one trained DNN model and one benign sample, the malicious perturbation could be optimized to encourage that the perturbed sample will be misclassified, while the perturbation is imperceptible to human eyes. It is dubbed *adversarial attack*, which happens in the inference stage (Goodfellow et al., 2015; Madry et al., 2018; Engstrom et al., 2019).

In contrast, some recent studies showed that some regular (*i.e.*, non-optimized) perturbations (*e.g.*, the local patch stamped on the right-bottom corner of the image) could also mislead DNNs, through influencing the model weights in the training process (Liu et al., 2020; Li et al., 2020; Gao et al., 2020). It is called as *backdoor attack*[1]. Specifically, some training samples are modified by adding the trigger (*e.g.*, the local patch). These modified samples with attacker-specified target labels, together with benign training samples, are fed into the DNN model for training. Consequently, the trained DNN model performs well on benign testing samples, similarly with the model trained using only benign samples; however, if the same trigger used in training is added onto a testing sample, then its prediction will be changed to the target label specified by the attacker. The backdoor attack could happen in the scenario that the training process is inaccessible or out of control by the user. Since the infected DNN model performs normally on benign samples, the user is difficult to realize the existence of the backdoor; even if the trigger is present, since it is usually just a regular local patch or even invisible, it is difficult for the user to identify the reason of the incorrect prediction. Hence, the insidious backdoor attack is a serious threat to the practical application of DNNs.

Many backdoor attacks have been proposed through designing different types of triggers (Gu et al., 2017; Chen et al., 2017; Liao et al., 2018; Li et al., 2019; Turner et al., 2019; Zhao et al., 2020b). It is interesting to find that most of existing works adopted the setting of *static* trigger, where the triggers across the training and testing images are located in the same area and have the same appearance, to the best of our knowledge. However, the user may modify the testing images before prediction, such that the trigger's location and appearance could be changed. It raises an intriguing question:

[1] Backdoor attack is also commonly called the 'Trojan attack', such as in (Liu et al., 2017a; Ding et al., 2019; Chen et al., 2019). In this paper, 'backdoor attack' refers specifically to attack methods that modify the training samples to create the backdoor, and we only focus on the image classification.

When the trigger in the attacked testing image is different from that used in training, can it still activate the hidden backdoor?

To answer this question, we explore the impacts of two basic characteristics of the backdoor trigger, including *location* and *appearance*. As shown in later experiments, we demonstrate that if the location or appearance of the trigger is slightly changed, then the attack performance may degrade sharply. It reveals that the backdoor attack with static trigger pattern may be non-robust to the change of trigger. The above observation inspires two further questions:

(1) *Can we utilize this non-robustness to defend existing backdoor attacks?* **(2)** *How to enhance the performance of existing backdoor attacks, such that they are robust to the change of trigger?*

In this work, we propose a simple yet effective defense method towards attacks with static trigger in which the testing sample is spatially transformed (*e.g.*, flipping or scaling) before the prediction. The spatial transformation on the whole image is a feasible approach to change the trigger's location and appearance, which may fail to activate the hidden backdoor in the infected DNN model. Furthermore, we also propose to enhance the transformation robustness of the attack that all poisoned images will be randomly transformed before feeding into the training process. The proposed method is equivalent to adding a preprocessing step on the poisoned images. This attack enhancement could be naturally combined with any backdoor attack method. Consequently, the attack's robustness to the change of trigger is significantly enhanced, and the attack can evade the proposed transformation-based defense. Besides, we demonstrate the connection between the proposed attack enhancement and the physical attack, and it explains that the enhanced attack could still succeed in physical scenarios, while the standard backdoor attacks with the static trigger will fail. Moreover, we present the visualization by utilizing the *saliency map* (Simonyan et al., 2013) and the *critical data routing path* (Wang et al., 2018) of images, under the standard backdoor attack and the enhanced backdoor attack, to further understand their differences.

The main contributions of this work are three-fold: **(1)** We demonstrate that the location and appearance of the backdoor trigger have crucial impacts on activating the backdoor. **(2)** We verify that attacks with the static trigger pattern are transformation vulnerable, which inspires a simple yet effective defense method. **(3)** We propose an effective method to enhance the robustness of existing attacks against the change of trigger, and connect the proposed enhancement with the physical attack. We hope that our approach could inspire more explorations on backdoor properties, to help the design of more advanced backdoor attack and defense methods.

2 Related Work

2.1 Backdoor Attacks

Backdoor attack is an emerging research area, which raises serious concerns about training with third-party datasets or platforms. Similar to the data poisoning (Biggio et al., 2012; Alfeld et al., 2016; Liu et al., 2019), backdoor adversary also tampers the training process to achieve their goals. However, these methods have different purposes. Specifically, the target of data poisoning is to degrade the model's performance on benign inputs, whereas the backdoor attack is aiming to misclassify inputs as a target class when the input is manipulated by adding a backdoor trigger. Meanwhile, the infected model can still correctly recognize the label for any benign sample.

The backdoor attack was first proposed in (Gu et al., 2017; 2019). After that, (Chen et al., 2017) first discussed the invisible backdoor attack, where the trigger is visually imperceptible. Recently, (Turner et al., 2019) proposed a more stealthy attack approach, dubbed label-consistent attack, where the target label of poisoned samples is consistent with their ground-truth label. Several other backdoor attacks have also been proposed for different purposes (Liu et al., 2017a; Yao et al., 2019; Bagdasaryan et al., 2020). Except for image classification, backdoor attacks were also demonstrated to be effective towards other tasks (Zhang et al., 2020; Zhao et al., 2020b; Kurita et al., 2020). Although various backdoor attacks were proposed, most of them have a static trigger setting, and the research on their mechanisms and properties is left far behind.

2.2 Backdoor Defenses

To defend backdoor attacks, several empirical defense methods were proposed. These methods can be roughly divided into six main categories, including *preprocessing-based defense* (Liu et al., 2017b; Gia Doan et al., 2019; Villarreal-Vasquez & Bhargava, 2020), *model reconstruction based defense* (Liu et al., 2017b; 2018; Zhao et al., 2020a), *trigger synthesis based defense* (Wang et al., 2019; Qiao et al., 2019; Zhu et al., 2020), *model diagnosis based defense* (Kolouri et al., 2020; Huang et al., 2020; Wang et al., 2020), *poison suppression based defense* (Hong et al., 2020; Du et al., 2020), and *sample filtering based defense* (Tran et al., 2018; Gao et al., 2019; Javaheripi et al., 2020). Unfortunately, existing defenses either suffer from high complexity or relatively low clean accuracy. Not to mention that most of them have already been bypassed by subsequently adaptive attacks. How to better defend against backdoor attacks is still an important open question.

3 The Property of Existing Attacks with Static Trigger

3.1 Backdoor Attack with Static Trigger

We consider the scenario that the user cannot fully control the training process of the model $C(\cdot; w)$. Let y_{target} denotes the target label, $\mathcal{D}_{train} = \{(\boldsymbol{x}, y)\}$ with $\boldsymbol{x} \in \{0, 1, \dots, 255\}^{C\times W\times H}$ indicates the (benign) training set. The target of backdoor attack is to obtain an *infected model*, which performs well on benign tesing images whereas it may have been injected some insidious backdoors.

The typical process of the backdoor attack has two main steps: **(1)** generate the poisoned image $\boldsymbol{x}_{poisoned}$ with target label y_{target}; **(2)** Adopte both the benign and poisoned samples for training.

The Generation of Poisoned Images. As stated above, generating poisoned images is the first step of backdoor attack. Specifically, the poisoned image $\boldsymbol{x}_{poisoned}$ is generated through a *stamping process* S based on the trigger $\boldsymbol{x}_{trigger}$ and the benign image $\boldsymbol{x}$, *i.e.*,

$$\boldsymbol{x}_{poisoned} = S(\boldsymbol{x}; \boldsymbol{x}_{trigger}) = (\mathbf{1} - \boldsymbol{\alpha}) \otimes \boldsymbol{x} + \boldsymbol{\alpha} \otimes \boldsymbol{x}_{trigger}, \tag{1}$$

where $\boldsymbol{\alpha} \in [0, 1]^{C\times W\times H}$ is a trade-off hyper-parameter and $\otimes$ indicates the element-wise product.

Training Process. We denote $\mathcal{D}_{benign}$ as all benign samples used for backdoor training ($\mathcal{D}_{benign} \subset \mathcal{D}_{train}$), and denote the set of poisoned samples as $\mathcal{D}_{poisoned} = \{(\boldsymbol{x}_{poisoned}, y_{target})\}$. Both of them are utilized to train the model, as follows

$$\min_{w} \mathbb{E}_{(x,y)\in\mathcal{D}_{poisoned}\cup\mathcal{D}_{benign}} \mathcal{L}\left(C(\boldsymbol{x}; w), y\right), \tag{2}$$

where $\mathcal{L}(\cdot)$ indicates the loss function, such as the cross entropy.

3.2 The Effects of Different Characteristics

One backdoor trigger can be specified by two independent characteristics, including *location* and *appearance*. To study their individual effects to backdoor attack, we firstly present their accurate definitions in Definition 2. One illustrative example is also shown in Figure 1.

Definition 1 (Minimum Covering Box)**.** *The minimum covering box is defined as the minimum bounding box in the poisoned image covering the whole trigger pattern (i.e., all non-zero $\boldsymbol{\alpha}$ entries).*

Definition 2 (Two Characteristics of Backdoor Trigger)**.** *A trigger can be defined by two independent characteristics, including **location** and **appearance**. Specifically, **location** is defined by the position of the pixel at the bottom right corner of the minimum covering box, and **appearance** is indicated by the color value and the specific arrangement of pixels corresponding to non-zero $\boldsymbol{\alpha}$ entries in the minimum covering box.*

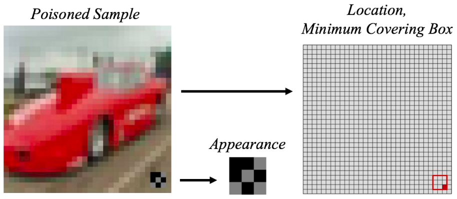
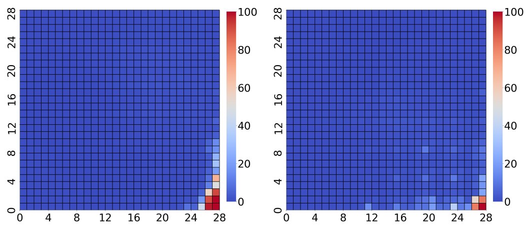


Figure 1: The illustration of characteristics of the backdoor trigger. The red box represents the boundary of the minimum covering box, and the red pixel indicates the trigger location.

Evaluation Criteria of Attacks. We adopt the attack performance to measure the effect, which is specified as the *attack success rate* (ASR). It is defined as the accuracy of attacked images predicted by the infected classifier $C(\cdot; \hat{w})$ with stamping process S, *i.e.*,

$$ASR_C(S) = \Pr_{(\boldsymbol{x},y)\in\mathcal{D}_{test}} \left[C\left(S(\boldsymbol{x}; \hat{w})\right) = y_{target} \mid y \neq y_{target}\right]. \tag{3}$$



(a) VGG-19 (b) ResNet-34

Figure 2: The heatmap of the attack success rate when the trigger is in different position at attacked images. The right corner is the position of the trigger in the poisoned images used for training.

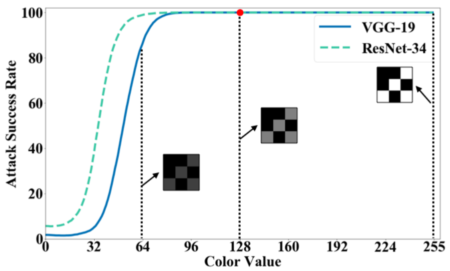


Figure 3: ASR and appearance of the trigger with different non-zero color value in attacked images. The red dot indicates the ASR of trigger with original color value (128 pixel).

For the sake of brevity, we will use $ASR(\cdot)$ instead, if specifying $C(\cdot; \hat{w})$ is not necessary.

Settings. In the following experiments in this section, we use BadNets (Gu et al., 2019) as an example to study the effects of location and appearance. We use VGG-19 (Simonyan & Zisserman, 2015) and ResNet-34 (He et al., 2016) as the model structure, and conduct experiments on CIFAR-10 dataset (Krizhevsky et al., 2009). The trigger is a 3×3 black-gray square, as shown in Figure 1. More details are shown in Appendix A.

The Effect of Location. While preserving the appearance of the trigger, we change its location in inference process to study its effect to the attack performance. As shown in Figure 2, when moving the location with a small distance ($2 \sim 3$ pixels, less than 10% of the image size), the ASR will drop sharply from 100% to below 50%. It tells that the attack performance is sensitive to the location of the backdoor trigger on the attacked image in the inference process.

The Effect of Appearance. While keeping the location of the trigger, we change its appearance in the inference stage to study the appearance's effect on the attack performance. The appearance could be modified by changing the shape or the pixel values of the trigger. For the sake of simplicity, here we only consider the change of pixel values. Specifically, there are only two values of the pixels within the trigger, $i.e.$, 0 and 128. We change the value 128 to different values from 0 to 255. The ASR scores corresponding to different pixel values are plotted in Figure 3. As shown in the figure, the ASR degrades sharply along with the decreasing of non-zero pixel values, while is not influenced when the non-zero pixel values are increased. According to this simple experiment, it is difficult to describe the exact relationship between the change of appearance and the attack performance, since the change modes of appearance are rather diverse. However, it at least tells that the attack is sensitive to the difference of appearance between the trigger on the attacked testing image and that used in training. More explorations about this phenomenon will be discussed in our future work.

4 Further Explorations of the Property

The studies presented in Section 3 demonstrate that the backdoor attack is sensitive to the difference between the training trigger and the testing trigger. It gives us two further questions: **(1)** Is it possible to utilize such a sensitivity to defend the current backdoor attacks with static trigger? **(2)** How to enhance the robustness of the backdoor attack to the change of trigger? We propose two simple yet effective approaches to answer this two questions in Section 4.1 and Section 4.2, respectively.

4.1 Backdoor Defense via Transformations

The answer to the first question is to change the location or appearance of the trigger in the inference process, such that the modified trigger may fail to activate the backdoor hidden in the model. However, since the user doesn't have the information about the trigger, it is impossible to exactly manipulate the trigger. Instead, we propose a transformation-based defense by changing the whole image with some transformations (*e.g.*, flipping or scaling), as shown in Definition 3.

Definition 3 (transformation-based defense)**.** *The transformation-based defense is defined as introducing a transformation-based pre-processing module on the testing image before prediction,* $i.e.$*, instead of predicting* $\boldsymbol{x}$*, it predicts* $T(\boldsymbol{x})$*, where* $T(\cdot)$ *is a transformation.*

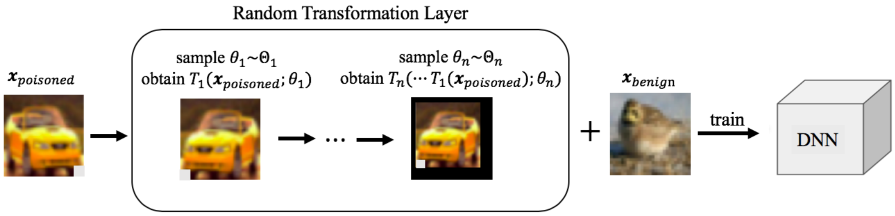

Random Transformation Layer
$x_{poisoned}$
sample $\theta_1 \sim \Theta_1$
obtain $T_1(x_{poisoned}; \theta_1)$
sample $\theta_n \sim \Theta_n$
obtain $T_n(\cdots T_1(x_{poisoned}); \theta_n)$
x_{benign}
train
DNN

Figure 4: The pipeline of the proposed transformation-based attack enhancement.

This simple defense method enjoys several advantages: **(1)** it is efficient since it only requires the transform the testing image; **(2)** it is attack-agnostic, therefore it can defend different attacks with static trigger simultaneously; **(3)** it is data-free and model-free, *i.e.*, compared with most existing defense methods, the defender does not need to have any clean samples or modify the model parameters. Accordingly, it would be the primary defense approach when adopting the third-party API of DNNs-based algorithms. These advantages will be further verified in Section 5.

In particular, we suggest to use spatial transformations for the defense, since it may probably change the location and appearance of the trigger simultaneously, while the location has a direct connection to the backdoor activation. A more comprehensive discussion about the defense with different transformations will be demonstrated in Appendix D.

4.2 Transformation-based Enhancement and Physical Backdoor Attack

In this section, we discuss how to enhance the transformation-robustness of existing attacks, and its connection with the physical attack.

Definition 4 (Transformation Robustness)**.** *The transformation robustness of attack with stamping process S under transformation $T(\cdot;\boldsymbol{\theta})$ (with parameter $\boldsymbol{\theta}$), the $R_T(S)$, is defined as the attack success rate after the transformation T, i.e.,*

$$R_T(S) = ASR(T(S)), \tag{4}$$

where

$$ASR(T(S)) = \Pr_{(\boldsymbol{x},y)\in\mathcal{D}} \left[C\left(T(S(\boldsymbol{x}))\right) = y_{target} \mid y \neq y_{target}\right].$$

Note that $R_T(S) \in [0,1]$. The larger value of $R_T(S)$ indicates the higher robustness towards transformation T. Besides, the transformation could also be a *compound transformation* $T(\cdot;\boldsymbol{\theta})$ of a sequence of basic transformation $\{T(\cdot;\theta_i)\}_{i=1}^{n}$, *i.e.*, $T(\cdot;\boldsymbol{\theta}) = T_n(T_{n-1}(\cdots T_1(\cdot;\theta_1);\theta_{n-1});\theta_n)$.

The key issues for improving transformation-robustness are how to determine the compound transformation and the corresponding parameter $\boldsymbol{\theta}$ used by defenders. In practice, the attacker is difficult to know the exact transformations. Even the adopted transformations are revealed to the attacker, the exact parameters in transformations cannot be known, as there may be randomness in practice (*i.e.*, different scaling factors in scaling transformation). To tackle this difficulty and to ensure the attack capability towards different possible transformation-based defenses, we specify T with the set of some common transformations. For each T_i, if there may be randomness in practice, then we define a value domain Θ_i for θ_i. Θ_i is parameterized by the maximal transformation size ϵ, *i.e.*,

$$\Theta_i = \{\theta | dist_i(\theta, I) \leq \epsilon_i\},$$

where $dist_i(\cdot,\cdot)$ is a given distance metric for T_i, and I indicates the identity transformation.

Consequently, the compound transformation used in the enhanced attack is specified as $\mathcal{T} = \{T(\cdot;\boldsymbol{\theta}) | \boldsymbol{\theta} \in \prod_{i=1}^{n} \Theta_i\}$. Then, the training objective of the enhanced attack is formulated as

$$\min_{w} \mathbb{E}_{\boldsymbol{\theta}} \left[\mathbb{E}_{(\boldsymbol{x},y)\in\mathcal{D}_{poisoned}^{(T(\cdot;\boldsymbol{\theta}))}\cup\mathcal{D}_{benign}} \left[\mathcal{L}\left(C(\boldsymbol{x};w),y\right)\right]\right]. \tag{5}$$

To solve the problem (5) exactly, attackers need to conduct the training process with all possible transformed variants, which is computation-consuming. Instead, we propose a sampling-based method for efficiency. Specifically, for each poisoned image, to handle the expectation over all possible configurations of $\boldsymbol{\theta}$, we sample one configuration, *i.e.*, $\boldsymbol{\theta} \sim \prod_{i=1}^{n} \Theta_i$, based on which we transform the original images. Then, we use the transformed poisoned images and benign images for training. The training process of the proposed enhanced attack is briefly illustrated in Figure 4.

Table 1: Comparison of different backdoor defenses on CIFAR-10 dataset. ‘Clean’ and ‘ASR’ indicates the accuracy (%) and attack success rate (%) on testing set, respectively. The boldface indicates the best results among all preprocessing based defenses.

Model Architectures →	VGG-19						ResNet-34					
Attack Methods →	BadNets		Blended Attack		Consistent Attack		BadNets		Blended Attack		Consistent Attack	
Defense Methods ↓	Clean	ASR	Clean	ASR	Clean	ASR	Clean	ASR	Clean	ASR	Clean	ASR
Standard	91.9	100	91.5	100	91.3	95.6	94.1	100	93.1	100	93.1	98.7
Fine-Pruning	91.3	0.7	83.6	0.2	72.6	0.1	92.1	0	91.9	0.3	92.0	18.9
Neural Cleanse	83.3	0.6	90.6	0.4	86.4	0.7	91.4	0.7	91.4	0.5	91.2	1.4
Auto-Encoder	86.4	2.1	86.0	1.7	85.4	**2.3**	87.5	2.7	87.2	1.9	88.4	**2.1**
Flip (Ours)	**91.0**	**1.1**	**91.1**	**0.9**	**90.5**	95.7	**93.6**	**0.8**	**92.8**	**0.8**	**92.3**	98.8
ShrinkPad-4 (Ours)	87.6	1.6	88.3	1.8	87.5	3.7	91.4	1.5	90.6	1.8	89.9	4.8

The connection between the proposed attack enhancement and physical attack. In real-world scenarios, the testing image may be acquired by some digitizing devices (*e.g.*, camera), rather than be directly provided in the digital space. In those scenarios, the trigger should be stamped on the object, which is then digitized by the camera to fool the model. It is dubbed *physical attack*. Some recent works (Schwarzschild et al., 2020; Wenger et al., 2020) demonstrated that existing backdoor attacks are vulnerable under the scenario of physical attack. It is due to that the relative distance and angle between the photo and the camera is varied in practice, therefore the location and appearance of the trigger in the digitized attacked image may be different from that of the trigger used for training. These spatial variations in physical scenarios can be approximated by some widely used transformations (*e.g.*, spatial transformations), which have been incorporated into proposed transformation-based enhancement. Thus, it is expected that the proposed transformation-based enhancement can serve as an effective physical attack, which will be futher verified in Section 5.4.

5 Experiment

5.1 Transformation-based Defense

In this section, we verify the effectiveness of the proposed defense with spatial transformations. We examine two simple spatial transformations, including left-right flipping (dubbed *Flip*), and padding after shrinking (dubbed *ShrinkPad*). Specifically, ShrinkPad consists of shrinking (based on bilinear interpolation) with a few pixels ($i.e.$, shrinking size), and random zero-padding around the shrunk image. The results of defense with non-spatial transformations will be shown in Appendix B.

Settings. We use three representative backdoor attacks, including BadNets (Gu et al., 2017), attack with blended strategy (Chen et al., 2017) (dubbed Blended Attack), and label consistent backdoor attack (Turner et al., 2019) (dubbed Consistent Attack) to evaluate the performance of backdoor defenses. For BadNets and Blended Attack, the trigger is a 3×3 black-white square, which is similar to the one used in Section 3.2. For defense comparison, we select four important baseline, including fine-pruning (Liu et al., 2018), neural cleanse (Wang et al., 2019), auto-encoder based defense (dubbed Auto-Encoder) (Liu et al., 2017b), and standard training (dubbed Standard).

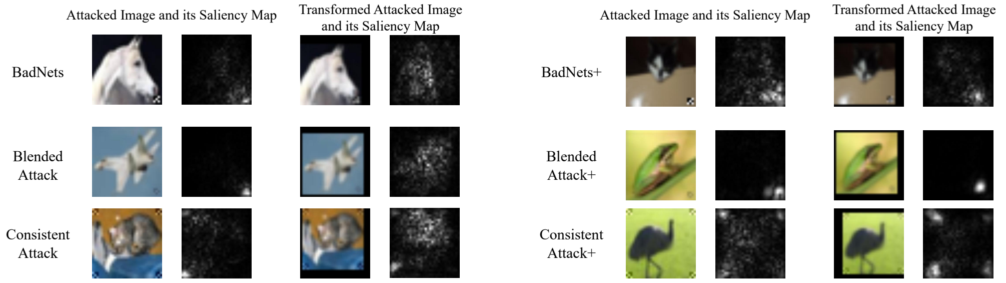
Results. As shown in Table 1, the proposed defense is effective. Specifically, ShrinkPad with 4 pixels shrinking size could decrease the ASR by more than 90% in all cases. Flip also shows satisfied defense performance towards BadNets and Blended attacks. But it doesn't work on defending against Consistent Attack, due to the symmetrical trigger used in Consistent Attack. Compared with the state-of-the-art preprocessing based method ($i.e.$, Auto-Encoder), the proposed method has higher clean accuracy and lower ASR in general. Besides, its performance is even on par with Fine-Pruning and Neural Cleanse, which require stronger defensive capabilities (*i.e.*, modify the model parameters and access to benign samples). Moreover, the proposed method is more efficient compared with other baseline methods, and is even more effective when the backdoor trigger is the universal adversarial perturbation (Moosavi-Dezfooli et al., 2017). It will be shown in Appendix E.

5.2 Attack Enhancement

Settings. In the enhanced backdoor attack, we adopt random Flip followed by random ShrinkPad in the random transformation layer. There is only one hyper-parameter in the enhanced attack, $i.e.$, the maximal shrinking size, which is set to 4 pixels in our experiments. Other settings are the same as those used in Section 5.1. More setting details and an ablation study about the effect of the hyper-parameter are shown in Appendix C and Appendix F, respectively.

Table 2: The comparison between standard backdoor attacks and enhanced backdoor attacks from the aspect of attack success rate against different transformation-based defenses.

Model Architectures →	VGG-19				ResNet-34			
Attacks ↓, Defenses →	Standard	Flip	ShrinkPad-2	ShrinkPad-4	Standard	Flip	ShrinkPad-2	ShrinkPad-4
BadNets	**100.0**	1.1	22.7	1.6	**100.0**	0.8	14.9	1.5
BadNets+	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
Blended Attack	**100.0**	0.9	40.8	1.8	**100.0**	0.8	18.2	1.8
Blended Attack+	99.9	**99.9**	**100.0**	**98.7**	**100.0**	**100.0**	**100.0**	**99.5**
Consistent Attack	**95.6**	**95.7**	67.1	3.7	**98.7**	**98.8**	24.2	4.8
Consistent Attack+	86.0	86.3	**97.2**	**90.9**	96.4	97.3	**97.4**	**98.7**


Attacked Image and its Saliency Map
Transformed Attacked Image and its Saliency Map
BadNets
Blended Attack
Consistent Attack
Attacked Image and its Saliency Map
Transformed Attacked Image and its Saliency Map
BadNets+
Blended Attack+
Consistent Attack+

(a) Standard Backdoor Attack

(b) Enhanced Backdoor Attack

Figure 5: The saliency maps of images under standard and enhanced backdoor attacks.

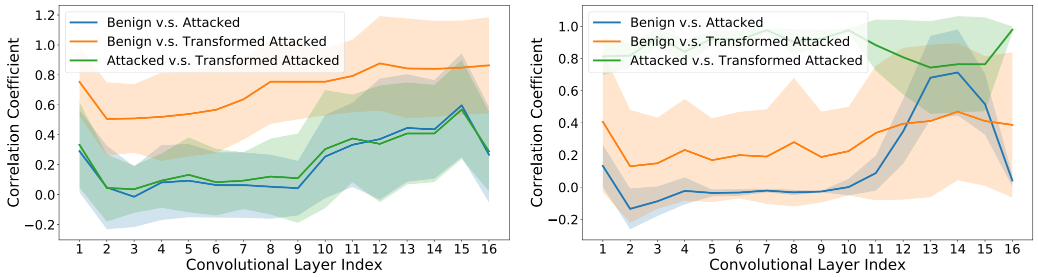
Results. As demonstrated in Table 2, enhanced backdoor attacks can still achieve a high ASR even under the defenses with spatial transformations. Specifically, the ASR of enhanced backdoor attacks is better than the one of their corresponding standard attack under defenses in almost all cases. Especially under ShrinkPad with shrinking 4 pixels, the ASR improvement of enhanced attacks is more than 85% (mostly over 95%). The only exception is the Consistent Attack+ under Flip defense. It is partially due to the fact the trigger of Consistent Attack is symmetrical, as mentioned in Section 5.1. Besides, compared to BadNets+ and Blended Attack+, Consistent Attack+ poisoned fewer images (see the attack settings), which is not favorable to the random trigger.

5.3 The Differences between Backdoor Attack and Enhanced Attack

In this section, we further explore the intrinsic difference between the standard backdoor attack and the enhanced backdoor attack (*i.e.*, the attack with the enhancement). Specifically, we adopt the *saliance map* (Simonyan et al., 2013) to understand their overall behaviors by identifying critical pixels of different images. Besides, we adopt the *critical data routing paths* (CDRPs) (Wang et al., 2018) between different samples to discuss the layer-wise behaviors of different attacks. CDRPs are the paths contribute most to the prediction. More setting details are shown in Appendix G.

As shown in Figure 5, the saliency area of regularly ($i.e.$, non-transformed) attacked images mainly lies in the area of the backdoor trigger on both standard attacks and enhanced attacks, while the outline area of the object is not significantly activated. This phenomenon explains why these attacked samples can successfully mislead infected networks. Moreover, the saliency maps of transformed attacked images and those of regularly attacked images have significantly different patterns under standard attacks. For example, the saliency map of transformed attacked images mainly activates at object structure rather than at the backdoor trigger. In contrast, the saliency maps of attacked and transformed attacked images share certain similarities under enhanced attacks. The saliency maps of both types of attacked images concentrate on the area of the backdoor trigger. Those visualization results somewhat explain the different behaviors of standard attacks and enhanced attacks.

Figure 6 (a) shows that the CDRPs of transformed attacked samples are similar with those of benign samples under standard backdoor attacks, corresponding to high correlation coefficients as demonstrated by the orange curve. In contrast, the CDRPs of another two pairs are different, corresponding to lower correlation coefficients shown in the blue and green curves. This phenomenon explains why only attacked samples are classified as the target label, while both benign and transformed attacked samples are still classified as their ground-truth labels. Figure 6 (b) shows that, under the enhanced backdoor attack, the CDRPs of attacked and transformed attacked samples are similar while their CDRPs are different from those of benign examples. This phenomenon is consistent with the result



(a) Standard Backdoor Attack

(b) Enhanced Backdoor Attack

Figure 6: Layerwise correlation coefficients of critical data routing paths between (benign sample, attacked sample), (benign sample, transformed attacked sample), and (attacked sample, transformed attacked sample). The background color indicates the standard deviation over 100 samples.

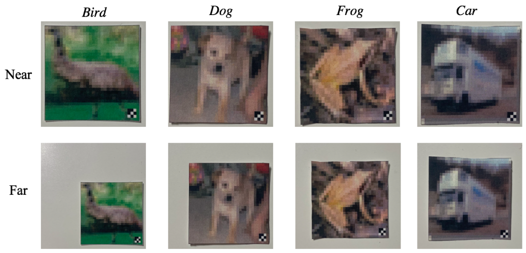


Figure 7: The pictures of some printed CIFAR-10 images taken by a camera with different distances. All pictures are classified as ‘Deer’ by the enhanced BadNets, whereas they will be classified as their benign label by the standard BadNets.

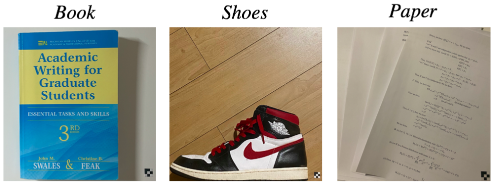


Figure 8: The picture of some out-of-sample images with the backdoor trigger taken by a camera. All pictures are classified as the target label ‘Deer’ by the enhanced BadNets, whereas they will be classified as their benign label by the standard BadNets.

that both attacked and transformed attacked samples will be predicted by the enhanced attack as the target labels, which is different from their ground-truth labels.

5.4 Physical Backdoor Attack

In this section, we further verify the effectiveness of the proposed attack enhancement under settings of the physical attack. Specifically, we evaluate BadNets and BadNets+ on the CIFAR-10 dataset. We randomly pick some testing images with backdoor trigger to take picture with differently relative location (near and far), as shown in Figure 7. Besides, we also take some *out-of-sample* pictures that are totally different from the training images on CIFAR-10 dataset, as shown in Figure 8.

In the results of all figures, BadNets+ successfully enforces the prediction to the target label, while BadNets fails. Besides, the enhanced backdoor attack method is not only robust in the physical scenarios, but also generalizes well on out-of-sample images. This out-of-sample generalization is probably due to the strong relationship between the backdoor trigger and target label learned in the infected model, so that the impact of the non-trigger part is somewhat ignored by the model.

6 Conclusion

In this paper, we explore the property of backdoor attacks. We demonstrate that existing attacks with static trigger are transformation vulnerable, inspired by which we propose a simple yet effective transformation-based defense. Besides, to reduce the transformation vulnerability of existing attacks, we propose a transformation-based enhancement by conducting the random spatial transformation on poisoned images before feeding into the training process. We also link the proposed attack enhancement to the physical attack and explore intrinsic differences between backdoor attack and enhanced attack. This work has shown that it is possible to develop simple yet effective defenses and attacks by utilizing some intrinsic properties. We hope that our approach could inspire more explorations on backdoor characteristics, to help the design of more advanced methods.

REFERENCES

Scott Alfeld, Xiaojin Zhu, and Paul Barford. Data poisoning attacks against autoregressive models. In *AAAI*, 2016.

Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *AISTATS*, 2020.

Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012.

Huili Chen, Cheng Fu, Jishen Zhao, and Farinaz Koushanfar. Deepinspect: A black-box trojan detection and mitigation framework for deep neural networks. In *AAAI*, 2019.

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.

Shaohua Ding, Yulong Tian, Fengyuan Xu, Qun Li, and Sheng Zhong. Trojan attack on deep generative models in autonomous driving. In *SecureComm*, 2019.

Min Du, Ruoxi Jia, and Dawn Song. Robust anomaly detection and backdoor attack detection via differential privacy. In *ICLR*, 2020.

Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. Exploring the landscape of spatial robustness. In *ICML*, 2019.

Yansong Gao, Chang Xu, Derui Wang, Shiping Chen, Damith C Ranasinghe, and Surya Nepal. STRIP: A defence against trojan attacks on deep neural networks. In *ACSAC*, 2019.

Yansong Gao, Bao Gia Doan, Zhi Zhang, Siqi Ma, Anmin Fu, Surya Nepal, and Hyoungshick Kim. Backdoor attacks and countermeasures on deep learning: A comprehensive review. *arXiv preprint arXiv:2007.10760*, 2020.

Jie Geng, Jianchao Fan, Hongyu Wang, Xiaorui Ma, Baoming Li, and Fuliang Chen. High-resolution sar image classification via deep convolutional autoencoders. *IEEE Geoscience and Remote Sensing Letters*, 12(11):2351–2355, 2015.

Bao Gia Doan, Ehsan Abbasnejad, and Damith C Ranasinghe. Februus: Input purification defense against trojan attacks on deep neural network systems. *arXiv preprint arXiv:1908.03369*, 2019.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Sanghyun Hong, Varun Chandrasekaran, Yiğitcan Kaya, Tudor Dumitraş, and Nicolas Papernot. On the effectiveness of mitigating data poisoning attacks with gradient shaping. *arXiv preprint arXiv:2002.11497*, 2020.

Shanjiaoyang Huang, Weiqi Peng, Zhiwei Jia, and Zhuowen Tu. One-pixel signature: Characterizing cnn models for backdoor detection. In *ECCV*, 2020.

Mojan Javaheripi, Mohammad Samragh, Gregory Fields, Tara Javidi, and Farinaz Koushanfar. Cleann: Accelerated trojan shield for embedded neural networks. In *ICCAD*, 2020.

Soheil Kolouri, Aniruddha Saha, Hamed Pirsiavash, and Heiko Hoffmann. Universal litmus patterns: Revealing backdoor attacks in cnns. In *CVPR*, 2020.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Keita Kurita, Paul Michel, and Graham Neubig. Weight poisoning attacks on pre-trained models. In *ACL*, 2020.

Shaofeng Li, Benjamin Zi Hao Zhao, Jiahao Yu, Minhui Xue, Dali Kaafar, and Haojin Zhu. Invisible backdoor attacks against deep neural networks. *arXiv preprint arXiv:1909.02742*, 2019.

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

Cong Liao, Haoti Zhong, Anna Squicciarini, Sencun Zhu, and David Miller. Backdoor embedding in convolutional neural network models via invisible perturbation. *arXiv preprint arXiv:1808.10307*, 2018.

Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *RAID*, 2018.

Xuanqing Liu, Si Si, Jerry Zhu, Yang Li, and Cho-Jui Hsieh. A unified framework for data poisoning attack to graph-based semi-supervised learning. In *NeurIPS*, 2019.

Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *NDSS*, 2017a.

Yuntao Liu, Yang Xie, and Ankur Srivastava. Neural trojans. In *ICCD*, 2017b.

Yuntao Liu, Ankit Mondal, Abhishek Chakraborty, Michael Zuzak, Nina Jacobsen, Daniel Xing, and Ankur Srivastava. A survey on neural trojans. In *ISQED*, 2020.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *CVPR*, 2017.

Ximing Qiao, Yukun Yang, and Hai Li. Defending neural backdoors via generative distribution modeling. In *NeurIPS*, 2019.

Avi Schwarzschild, Micah Goldblum, Arjun Gupta, John P Dickerson, and Tom Goldstein. Just how toxic is data poisoning? a unified benchmark for backdoor and data poisoning attacks. *arXiv preprint arXiv:2006.12557*, 2020.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. In *NeurIPS*, 2018.

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

Miguel Villarreal-Vasquez and Bharat Bhargava. Confoc: Content-focus protection against trojan attacks on neural networks. *arXiv preprint arXiv:2007.00711*, 2020.

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 *IEEE S&P*, 2019.

Ren Wang, Gaoyuan Zhang, Sijia Liu, Pin-Yu Chen, Jinjun Xiong, and Meng Wang. Practical detection of trojan neural networks: Data-limited and data-free cases. In *ECCV*, 2020.

Yulong Wang, Hang Su, Bo Zhang, and Xiaolin Hu. Interpret neural networks by identifying critical data routing paths. In *CVPR*, 2018.

Emily Wenger, Josephine Passananti, Yuanshun Yao, Haitao Zheng, and Ben Y Zhao. Backdoor attacks on facial recognition in the physical world. *arXiv preprint arXiv:2006.14580*, 2020.

Yuanshun Yao, Huiying Li, Haitao Zheng, and Ben Y Zhao. Latent backdoor attacks on deep neural networks. In *CCS*, 2019.

Zaixi Zhang, Jinyuan Jia, Binghui Wang, and Neil Zhenqiang Gong. Backdoor attacks to graph neural networks. *arXiv preprint arXiv:2006.11165*, 2020.

Pu Zhao, Pin-Yu Chen, Payel Das, Karthikeyan Natesan Ramamurthy, and Xue Lin. Bridging mode connectivity in loss landscapes and adversarial robustness. In *ICLR*, 2020a.

Shihao Zhao, Xingjun Ma, Xiang Zheng, James Bailey, Jingjing Chen, and Yu-Gang Jiang. Clean-label backdoor attacks on video recognition models. In *CVPR*, 2020b.

Liuwan Zhu, Rui Ning, Cong Wang, Chunsheng Xin, and Hongyi Wu. Gangsweep: Sweep out neural backdoors by gan. In *ACM MM*, 2020.

A Settings for the Effects of Different Characteristics

In this section, we illustrate the detailed settings in Section 3.2 of the main manuscript.

Attack Setup. We discuss the effects of trigger characteristics based on BadNets (Gu et al., 2019) in these experiments. The trigger is a 3×3 black-gray square, as shown in Figure 1. The trade-off hyper-parameter $\boldsymbol{\alpha}$ is set as $\boldsymbol{\alpha} \in \{0, 1\}^{3 \times 32 \times 32}$. The values of $\boldsymbol{\alpha}$ entries corresponding to the pixels located in the minimum covering box are 1, while other values are 0.

Training Setup. We evaluate the effect with two popular CNN models, including VGG-19 (Simonyan & Zisserman, 2015) and ResNet-34 (He et al., 2016), on the benchmark database CIFAR-10 (Krizhevsky et al., 2009). In terms of training, we adopt the SGD with momentum 0.9, weight decay 10^{-4}, and batch size 128 for all training processes. We train VGG-19 through 164 epochs with an initial learning rate of 0.1, which is decreased by a factor 10 at epochs 81 and 122; and train ResNet-34 through 300 epochs with an initial learning rate of 0.1, which is decreased by a factor 10 at epochs 150 and 250. The ratio of poisoned samples in training set, *i.e.*, $R = \frac{N_{poisoned}}{(N_{poisoned}+N_{benign})}$, is set to 0.25. All experiments are conducted on one single GeForce GTX 1080 GPU, and the implementation is conducted based on the open source code[2].

Data Preprocessing. Before adding a backdoor trigger to the benign sample to generate poisoned samples, we conduct standard data augmentation techniques for benign images. Specifically, 4-pixel padding is used before performing random crops of size 32×32.

B Settings for Transformation-based Defense

In this section, we illustrate the detailed settings in Section 5.1 of the main manuscript.

Defense Setup. We examine Flip and ShrinkPad with shrinking size $\in \{1, 2, 3, 4\}$. Except for aforementioned Flip and ShrinkPad, we also conduct fine-pruning (Liu et al., 2018), neural cleanse (Wang et al., 2019), and auto-encoder based defense (dubbed Auto-Encoder) (Liu et al., 2017b), which are the state-of-the-art defenses. The model with standard training and testing process is also provided, which is dubbed *Standard*. Specifically, the fine-pruning method consists of two stages, including pruning and fine-tuning. Per the settings in the original paper, we prune the parameters of the last component (convolutional layer for VGG, convolutional block for ResNet). The original test set is equally divided as two disjoint subsets, including the validation set and the practical test set. The fraction of pruned neurons is determined through grid-search on the validation set, and the performance is evaluated on the practical test set. In particular, we found that the fine-tuning with even one epoch may reactivate the removed backdoor, therefore it is removed in the experiments. For neural cleanse, all settings are based on the open-source code[3] provided by the authors. For Auto-Encoder, we train the convolutional auto-encoder (Geng et al., 2015) with 100 epochs, learning rate 0.001 and batch size 16. The implementation is based on the open-source code[4]. Above defense experiments are conducted on one single GeForce GTX 1080 GPU.

Attack Setup. We use three representative state-of-the-art backdoor attacks, including BadNets (Gu et al., 2017), attack with blended strategy (Chen et al., 2017) (dubbed Blended Attack), and label consistent backdoor attack (Turner et al., 2019) (dubbed Consistent Attack) to evaluate the performance of backdoor defenses. The target label is *Deer*. Specifically, for BadNets, except for the trigger appearance, other settings are the same as those illustrated in Section 3.2. The non-zero pixel value is modified from 128 to 255; For Blended Attack, the trigger is the same as the one of BadNets, the ratio of poisoned samples is set to 0.2, and the hyper-parameter $\boldsymbol{\alpha} \in \{0, 0.2\}^{3 \times 32 \times 32}$. The values of the $\boldsymbol{\alpha}$ entries corresponding to the pixels located in the minimum covering box are 0.2, while other values are 0; For Consistent Attack, the ratio of poisoned sample over all training samples with target label is set to 0.25, and $\boldsymbol{\alpha} \in \{0, 0.25\}^{3 \times 32 \times 32}$. The trigger of Consistent Attack is quite different from the one used in BadNets and Blended Attack, which is symmetrical. All these settings follow their original papers. Some examples of poisoned sample generated by different attacks are shown in Figure 9.

[2] `https://github.com/bearpaw/pytorch-classification`

[3] `https://github.com/bolunwang/backdoor`

[4] `https://github.com/jellycsc/PyTorch-CIFAR-10-autoencoder`

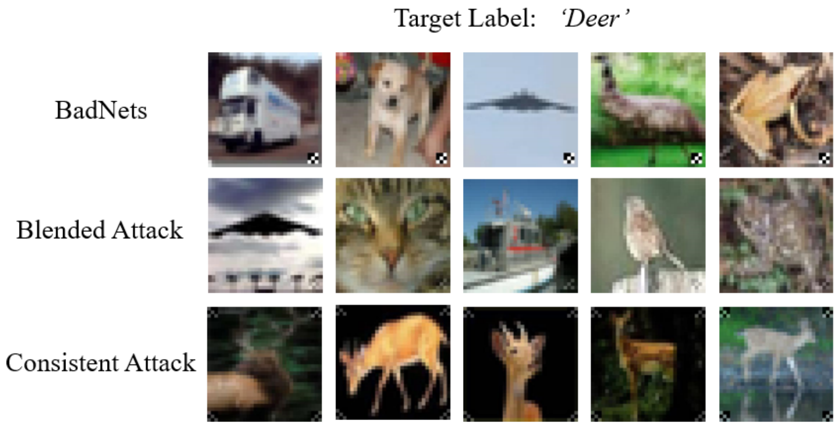


Figure 9: Some poisoned samples generated by different backdoor attack methods. In this experiment, the target label is 'Deer'. Except for the Consistent Attack, the ground-truth label of generated poisoned samples and the target label is not consistent.

Training Setup. The training settings are the same as those adopted in Section 3.2.

C Settings for Attack Enhancement

In this section, we illustrate the detailed settings in Section 5.2 of the main manuscript.

Settings. In the enhanced backdoor attacks, we adopt random Flip followed by random ShrinkPad in the random transformation layer. Note that there is only one hyper-parameter in enhanced attacks, $i.e.$, the maximal shrinking size, which is set to 4 pixels in this experiment. We examine three enhanced backdoor attacks, including enhanced BadNets (BadNet+), enhanced Blended Attack (Blended Attack+), and enhanced Consistent Attack (Consistent Attack+) with their correspondingly standard attack in the experiments. In particular, when evaluating the ASR of enhanced attacks under defenses, the random transformation is also adopted on the benign training samples rather than only on the poisoned samples during the training process. This modification is to exclude the possibility that the transformation itself creates a new backdoor. For example, the zero-padding in ShrinkPad may create a new backdoor activated by the black edges of the image. If the random transformations are only adopted on the poisoned samples, we cannot identify whether the improvement of ASR under ShrinkPad is due to that the enhanced attacks are more robust to transformation, or due to that the black edges introduced by ShrinkPad activate the new edge-related backdoor of enhanced attacks. Other settings are the same as those used in Section 5.1 of the main manuscript.

D Defense with Non-spatial Transformation

In this section, we examine the effectiveness of proposed transformation-based defense with non-spatial transformations. Specifically, we evaluate two most widely used transformations, including the additive Gaussian noise and the color-shifting, which only change the trigger appearance while preserving its location.

Settings. We examine the performance of defense under ResNet-34 structure. For the additive Gaussian noise, the mean is set as zero, and the standard deviation (std), is selected from $\{\frac{5}{255}, \frac{10}{255}, \frac{15}{255}, \frac{20}{255}\}$. We examine four types of color-shifting, including modifying hue (dubbed Hue), modifying contrast (dubbed Contrast), modifying brightness (dubbed Brightness), and modifying saturation (dubbed Saturation). All images are randomly transformed with maximum perturbation size $\in \{0.1, 0.2, 0.3, 0.4\}$, based on the *ColorJitter* function provided in torchvision. Some examples of transformed attacked images are shown in Figures 10-11.

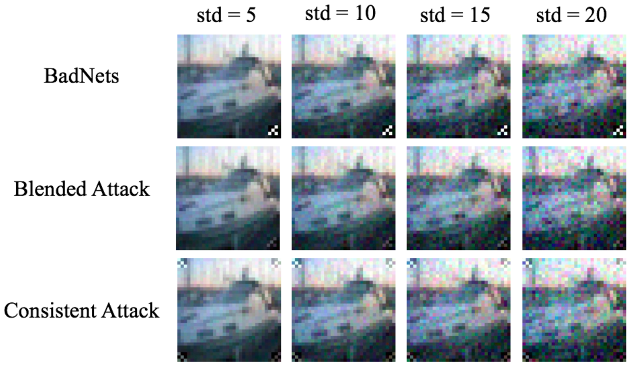


Figure 10: The example of some generated attacked samples with additive Gaussian noise.

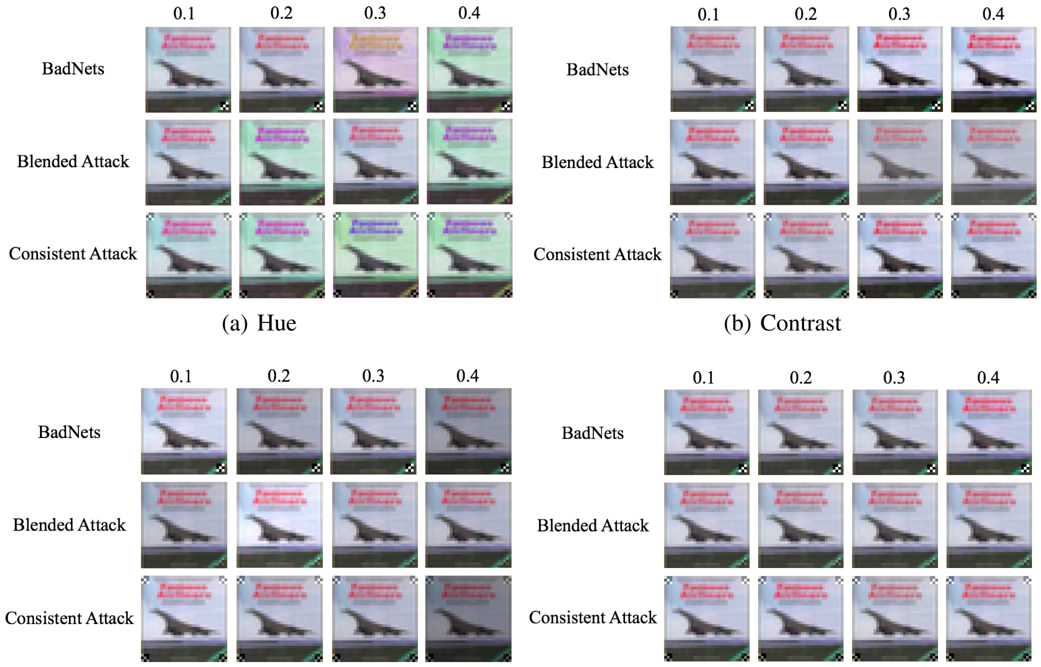
Table 3: Attack success rate and clean test accuracy under additive Gaussian noise with different standard deviation.

Standard Deviation (std) →	5		10		15		20	
Attack Type ↓	Clean	ASR	Clean	ASR	Clean	ASR	Clean	ASR
BadNets	91.2	100	79.8	100	58.1	100	36.4	100
Blended Attack	90.8	100	81.4	100	64.5	99.9	46.0	99.5
Consistent Attack	90.9	98.7	81.9	99.1	65.1	99.4	44.6	99.6

Table 4: Attack success rate and clean test accuracy under different types of color-shifting with different maximum perturbation sizes. We examine four types of color-shifting, including modifying hue (dubbed Hue), modifying contrast (dubbed Contrast), modifying brightness (dubbed Brightness), and modifying saturation (dubbed Saturation). All images are randomly transformed with maximum perturbation size $\in \{0.1, 0.2, 0.3, 0.4\}$.

	Maximum Perturbation Size →	0.1		0.2		0.3		0.4	
Shifting Type ↓	Attack Type ↓	Clean	ASR	Clean	ASR	Clean	ASR	Clean	ASR
Hue	BadNets	93.2	100	91.6	100	89.4	100	88.5	100
	Blended Attack	92.1	100	89.8	100	88	100	86.9	100
	Consistent Attack	91.9	98.7	89.2	98.8	87.2	99	85.8	99.1
Contrast	BadNets	94.2	100	94.0	100	93.8	100	93.7	100
	Blended Attack	92.9	100	92.9	100	92.8	100	92.6	100
	Consistent Attack	93.0	98.5	92.8	97.9	92.6	97.5	92.4	96.4
Brightness	BadNets	94.1	100	93.9	100	93.7	100	93.4	100
	Blended Attack	93.0	100	92.9	99.8	92.7	99.0	92.4	98.4
	Consistent Attack	93.0	98.1	92.9	96.4	92.6	94.5	91.9	92.6
Saturation	BadNets	94.1	100	94.1	100	94.1	100	94.0	100
	Blended Attack	93.1	100	93.1	100	93.0	100	93.0	100
	Consistent Attack	93.0	98.7	93.0	98.8	93.0	98.7	92.8	98.7

Results. As shown in Tables 3-4, both the additive Gaussian noise and color-shifting have limited effects on defending backdoor attacks. Especially for the additive Gaussian noise, despite the use of a large standard deviation, ASR has not decreased even though the clean accuracy has decreased by more than 30%. Besides, color-shifting has limited effects on both defense performance and clean accuracy. The possible reason is that the effects of these transformations on the trigger appearance are not significant, as shown in Figures 11. Moreover, the exact impact of the difference in trigger appearance on the attack success rate of backdoor attacks are still unclear, which will be further studied in the future work. Accordingly, in the proposed transformation-based defense, we recommend to use spatial-transformations instead of non-spatial transformations. The spatial-transformations may probably change the location and appearance of the trigger simultaneously, and the location has a direct connection to the backdoor activation.



(a) Hue

(b) Contrast

(c) Brightness

(d) Saturation

Figure 11: Transformed attacked samples with different types of color-shifting. All images are randomly transformed with maximum perturbation size $\in \{0.1, 0.2, 0.3, 0.4\}$.

Table 5: The average training time (seconds) of different defenses.

	VGG-19	ResNet-34
Fine-Pruning	$\sim$ 400	$\sim$ 600
Neural Cleanse	$\sim$ 30000	$\sim$ 80000
Auto-Encoder	$\sim$ 2000	
Flip	$\sim$ **0**	$\sim$ **0**
ShrinkPad	$\sim$ **0**	$\sim$ **0**

E More Results about Transformation-based Defense

E.1 Comparison of Different Defenses from the Aspect of Efficiency

The proposed transformation-based defense method only involves an extra simple transformation of the image in the inference process. However, all other baseline methods require additional training or optimization. Compared to state-of-the-art methods, the proposed transformation-based defense requires less additional costs. To verify the efficiency of the proposed method, we report the average training time over defending all three attacks of each defense method, as shown in Table 5. Besides, there is only one hyper-parameter in ShrinkPad, *i.e.*, the shrinking size, while there are multiple hyper-parameters in compared baseline methods. In conclusion, transformation-based defenses (with spatial transformation) reach competitive performance compared with state-of-the-art defense methods, while with almost no additional computational cost and fewer hyper-parameters to adjust.

E.2 Defense against Attack with Universal Perturbation as the Trigger

Compared with the non-optimized trigger pattern (*e.g.*, a 3×3 square, as the one we used in Section 5.1), recently, the universal adversarial perturbation (Moosavi-Dezfooli et al., 2017) was verified to be a more effective trigger in BadNets-type attack (Zhao et al., 2020b). In this section, we compare

Table 6: Comparison of different backdoor defenses against BadNets with universal perturbation as the backdoor trigger on CIFAR-10 dataset. 'Clean' and 'ASR' indicates the accuracy (%) and attack success rate (%) on testing set, respectively.

	VGG-19		ResNet-34	
	Clean	ASR	Clean	ASR
Standard	89.9	100	93.9	100
Fine-Pruning	46.9	0	87.8	1.6
Neural-Cleanse	48.0	100	73.3	100
Auto-Encoder	84.0	11.6	85.9	4.5
Flip	88.9	3.6	93.6	0.7
ShrinkPad-4	85.0	12.4	91.5	6.2

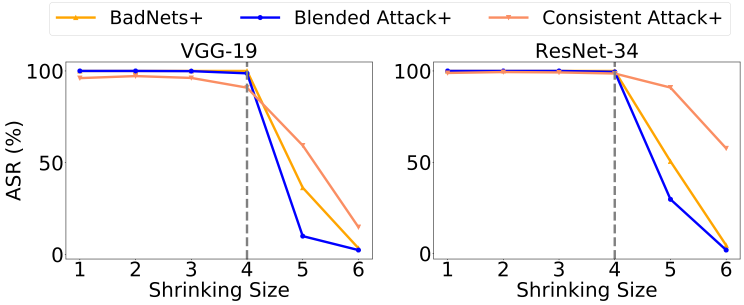


Figure 12: Attack success rates of enhanced attacks with the maximal shrinking size of 4-pixels, under the ShrinkPad defense with different shrinking sizes.

different defense methods against a more challenging attack setting, *i.e.*, BadNets with universal perturbation as the backdoor trigger.

Settings. As mentioned above, we adopt the universal perturbation as the backdoor trigger in this experiment. The perturbation is generated based on a pre-trained benign model. The training scheme of the benign model is the same as that of its correspondingly infected version (except for the training set). Other settings are the same as those used in Section 5.1.

Results. Compared with the state-of-the-art preprocessing based method (*i.e.*, Auto-Encoder), our method still has higher clean accuracy and lower ASR under this attack setting. It is interesting to find that both Fine-Pruning and Neural-Cleanse fail in this scenario. For Fine-Pruning, its effectiveness relies on the assumption that the backdoor is hidden in neurons that are not related to those used for encoding the normal behavior of the model. This assumption is not necessarily true in this scenario, since the adversarial perturbation is generated according to the effects of all neurons. The failure of Neural-Cleanse may probably due to the failure of its trigger-reconstruction stage, since the reconstruction of a 'noise' is significantly more difficult than that of a compact object (*i.e.*, 3×3 square that we used in Section 5.1). This experiment again verifies the effectiveness of the proposed method and the importance of designing defenses by utilizing the properties of attacks.

F Ablation Study

In this section, we study the effect of shrinking size in the proposed defense and the effect of maximal shrinking size in enhanced backdoor attacks. Except for the studied hyper-parameters, other settings are the same as those used in Section 5.1 and Section 5.2, unless otherwise specified.

F.1 The Effect of Shrinking Size in the Transformation-based Defense.

As demonstrated in Section 5.1, adopting ShrinkPad with a small pixels shrinking size will significantly reduce the ASR of standard attacks. In this section, we discuss the effect of shrinking size in defending against enhanced backdoor attacks. Specifically, we evaluate the effect of defending the enhanced attacks with 4-pixels maximal shrinking size.

As shown in Figure 12, the ASR decreases along with the increase of the shrinking size under all settings. Although when the shrinking size in ShrinkPad is not larger than the maximal shrinking

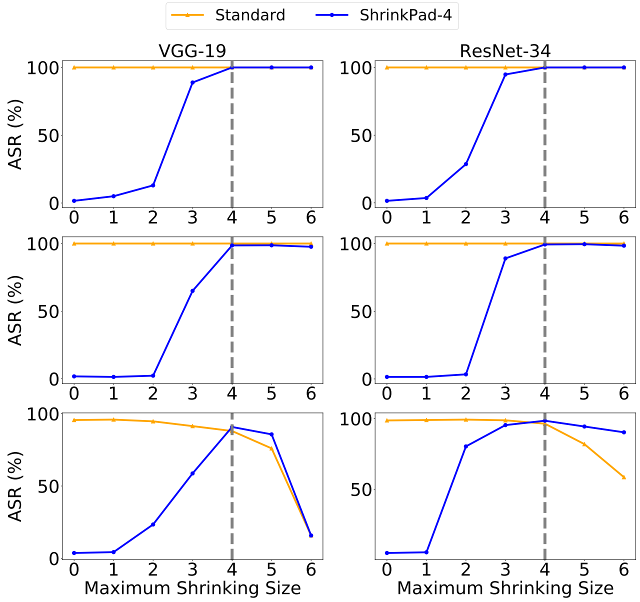


Figure 13: Attack success rate of enhanced backdoor attacks $w.r.t.$ different maximal shrinking sizes under ShrinkPad-4 and Standard. **First Row:** 'BadNets+'; **Second Row:** 'Blended Attack+'; **Last Row:** 'Consistent Attack+'.

size used in enhanced attacks (*i.e.*, 4 pixels), the ASR values are still very high, indicating that the defense performance of ShrinkPad is not satisfied. However, when the shrinking size is bigger than the maximal shrinking size used in enhanced attacks (4 pixels), the ASR will decrease dramatically. The above results indicate that the shrinking size used in the ShrinkPad defense should be larger than the maximal shrinking size used in enhanced attacks, to ensure the satisfied defense performance.

F.2 The Effect of Maximal Shrinking Size in Enhanced Backdoor Attacks.

We evaluate the performance of the enhanced backdoor attack with different maximal shrinking sizes, to attack the Standard model (no defense) and the model with the 'ShrinkPad-4' defense.

The attack results measured by ASR are shown in Figure 13. To attack the Standard model, the ASR values are very high and are almost unchanged when the maximal shrinking size varies. However, the ASR values of the Consistent Attack+ decreases along with the increase of the maximal shrinking size. The larger value of the maximal shrinking size indicates the more randomness of triggers in training, which requires more poisoned training images to create the backdoor. As mentioned in Section 5.2 in the main manuscript, the number of poisoned training images in Consistent Attack+ is insufficient. To attack the model with the defense ShrinkPad-4, when the maximal shrinking size is smaller than the shrinking size 4 in ShrinkPad-4, the ASR values increase from 0 to almost 100. When the maximal shrinking size is larger than the shrinking size 4, the ASR values of BadNets+ and Blended Attack+ are still about 100; but, the ASR values of Consistent Attack+ become to decrease, still due to the insufficiency of poisoned training images. These phenomena indicate that the proposed attack enhancement can indeed reduce the transformation vulnerability of existing backdoor attacks.

G More Details about the Differences between Standard Backdoor Attack and Enhanced Backdoor Attack

G.1 A Brief Introduction about Sliency Map and Critical Data Routing Path

Sliency map. Saliency map (Simonyan et al., 2013) is widely used in computer vision to provide indications of the most salient regions within images. By creating the saliency map for a DNN model, we can obtain some intuition on *where the network is paying the most attention to* in an

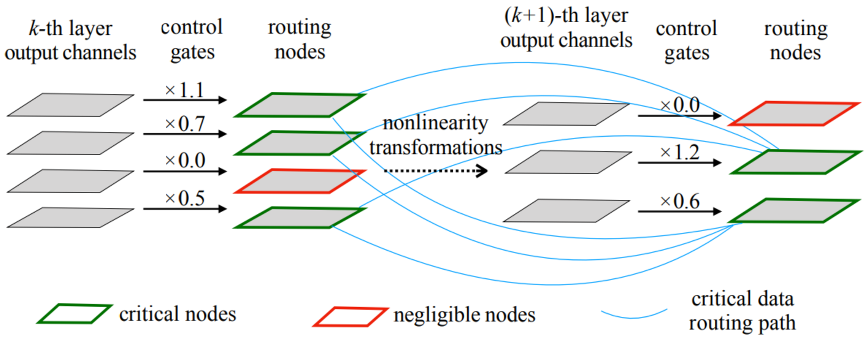

k-th layer output channels
control gates
routing nodes
(k+1)-th layer output channels
control gates
routing nodes
×1.1
×0.7
×0.0
×0.5
nonlinearity transformations
×0.0
×1.2
×0.6
critical nodes
negligible nodes
critical data routing path

Figure 14: The control gates are multiplied to the layer's output channel-wise, resulting in the actual routing nodes. The layerwise routing nodes are linked together to compose the routing paths (Wang et al., 2018).

input image. Specifically, for the image $\boldsymbol{x}$ and a classifier with the class score function $S_c(\cdot)$, the image-specific class saliency is the magnitude of the derivative of $S_c(\boldsymbol{x})$ $w.r.t.$ $\boldsymbol{x}$.

Critical data routing path. The critical data routing paths (CDRPs) (Wang et al., 2018) is a distillation guided method, which can be used to interpret DNNs by identifying critical data routing paths and analyzing the functional processing behavior of the corresponding layers. Compared with the sliency map, CDRPs provide more layer-wise information of DNNs. Specifically, it discover the critical nodes on the data routing paths during the inference process for a specific image by learning associated control gates for each layer's output channel. Accordingly, the routing paths can be represented based on the responses of concatenated control gates from all the layers, which *reflect the network's semantic selectivity regarding to the input patterns and more detailed functional process across different layer levels*. An illustrative example is shown in Figure 14.

Let $f_{\boldsymbol{w}}(\cdot)$ be a pretrained network, $f_{\boldsymbol{w}}(\cdot;\boldsymbol{\Lambda})$ is a network with control gates $\boldsymbol{\Lambda} = (\boldsymbol{\lambda_1}, \cdots, \boldsymbol{\lambda}_K)$ where $\boldsymbol{\lambda}_i$ is the control gates of i-th layer. The CDRPs of a image $\boldsymbol{x}$ is optimized by a distillation-guided method, as follows:

$$\begin{aligned} &\min_{\boldsymbol{\Lambda}} \mathcal{L}(f_{\boldsymbol{w}}(\boldsymbol{x}), f_{\boldsymbol{w}}(\boldsymbol{x};\boldsymbol{\Lambda})) + \gamma \sum_{i=1}^{K} |\boldsymbol{\lambda}_i|_1 \\ &s.t. \quad \boldsymbol{\lambda}_i \succeq 0,\ i = 1, 2, \cdots, K, \end{aligned} \tag{6}$$

where $\mathcal{L}$ is the cross entropy and γ is a trade-off hyper-parameter.

G.2 Settings

Settings for visualizing the sliency map. We visualize the saliency map (Simonyan et al., 2013) towards their predicted label of attacked and transformed attacked images, under both standard attacks and enhanced attacks. The attacked images are transformed by ShrinkPad with 4-pixels shrinking size, and the saliency map is obtained based on the open-source code[5].

Settings for visualizing the critical data routing path. We randomly select 100 benign testing samples (with the ground-truth label different from the target label), and their correspondingly attacked samples and transformed attack samples for generating CDRPs, under the attacks of BadNets and BadNets+. After generating all CDRPs of each sample, we calculate the layerwise correlation coefficients of CDRPs between each sample pair (totally 300 pairs), including (benign sample, attacked sample), (benign sample, transformed attacked sample), and (attacked sample, transformed attacked sample), under BadNets and BadNets+, respectively. The CDRP is obtained based on the open-source code[6].

[5] `https://github.com/MisaOgura/flashtorch`

[6] `https://github.com/frankwang345/cdrp-detect`