Rethinking the Backdoor Attacks' Triggers: A Frequency Perspective

Yi Zeng*
Virginia Tech
Blacksburg, VA 24061, USA
yizeng@vt.edu

Won Park*
University of Michigan
Ann Arbor, MI 48109, USA
wonpark@umich.edu

Z. Morley Mao
University of Michigan
Ann Arbor, MI 48109, USA
zmao@umich.edu

Ruoxi Jia
Virginia Tech
Blacksburg, VA 24061, USA
ruoxijia@vt.edu

Abstract

Backdoor attacks have been considered a severe security threat to deep learning. Such attacks can make models perform abnormally on inputs with predefined triggers and still retain state-of-the-art performance on clean data. While backdoor attacks have been thoroughly investigated in the image domain from both attackers' and defenders' sides, an analysis in the frequency domain has been missing thus far.

This paper first revisits existing backdoor triggers from a frequency perspective and performs a comprehensive analysis. Our results show that many current backdoor attacks exhibit severe high-frequency artifacts, which persist across different datasets and resolutions. We further demonstrate these high-frequency artifacts enable a simple way to detect existing backdoor triggers at a detection rate of 98.50% without prior knowledge of the attack details and the target model. Acknowledging previous attacks' weaknesses, we propose a practical way to create smooth backdoor triggers without high-frequency artifacts and study their detectability. We show that existing defense works can benefit by incorporating these smooth triggers into their design consideration. Moreover, we show that the detector tuned over stronger smooth triggers can generalize well to unseen weak smooth triggers. In short, our work emphasizes the importance of considering frequency analysis when designing both backdoor attacks and defenses in deep learning.

1. Introduction

Backdoor attacks are the attacks where adversaries deliberately manipulate a proportion of the training data [11, 5], or the model's parameters [18], to make the model recognize a backdoor trigger as the desired target label(s). When the backdoor trigger is introduced during test-time, the poisoned model exhibits a particular output behavior of the adversary's choosing (e.g., a misclassification). Backdoor triggers have been demonstrated to perform malicious tasks on security-concerned deep learning services, such as converting the label of a stop sign [11] or resulting misidentified faces [5], thereby posing significant risks.

State-of-the-art backdoor triggers are designed to be inconspicuous to human observers. One idea of generating such triggers is to use patterns of commonplace objects [18, 30]. For instance, one could use glasses–commonplace objects appearing in a face image–as a trigger to backdoor a face recognition model, thereby hiding the triggers "in the human psyche." Another approach to generate "hidden" or "invisible" triggers is to inject imperceptible perturbations via solving a norm-constrained optimal attack problem [17, 24] or leveraging GANs [25].

Previous research on backdoor data detection either identifies outliers directly in the image space [22] or analyzes the network activations based on an image input [23, 20, 3, 15]. In contrast, we provide a comprehensive analysis of the frequency spectrum across various existing triggers and multiple datasets. We find that all existing ideas of generating samples contain triggers exhibit severe high-frequency artifacts. We provide a detailed analysis of the causes of the high-frequency artifacts for different triggers and show that these artifacts stem from either the trigger pattern per se or the methodology of inserting the trigger.

Based on these insights, we demonstrate that the frequency domain can efficiently identify potential backdoor data in both the training and test phase. We build a detection pipeline based on a simple supervised learning framework and proper data augmentation as a demonstration. It can identify existing backdoor triggers at a detection rate

*Yi Zeng and Won Park contributed equally.

of 98.5% without prior knowledge of the types of backdoor attacks used. A high detection rate is still maintained even when the data used for training and testing the detector have different input distributions and are from different datasets.

Given that existing triggers are easily detectable in the frequency domain, our natural question is whether it is possible to design effective backdoor triggers without high-frequency artifacts (which we will refer to as smooth triggers hereinafter). A straightforward approach to generating smooth triggers is to apply a low-pass filter to existing triggers directly. However, we find that this simple approach cannot achieve a satisfying attack success rate while maintaining trigger stealthiness. To design more effective smooth triggers, we first formulate the trigger design problem as a bilevel optimization problem and then propose a practical heuristic algorithm to create triggers. Our experiments show that our proposed triggers can achieve a much higher attack success rate than the simple low-pass filtered triggers. We further study the detectability of the triggers. We show that existing defense works can benefit from incorporating these smooth triggers into their design consideration. Our experiments also demonstrate that the detector trained over strong, smooth triggers can generalize well to unseen weak smooth triggers.

Overall, our work manifests the importance of the overlooked frequency analysis in the design of both backdoor attacks and defenses. We open-source the experiment codes and welcome the public to contribute to future developments[1]. Our key contributions are summarized as follows: **1)** We perform a comprehensive frequency-domain analysis of existing backdoors triggers, revealing severe high-frequency artifacts commonly across different datasets and resolutions. **2)** We present a detailed analysis of the causes of these artifacts. **3)** We show the effectiveness of employing frequency representations for detecting existing triggers. **4)** We propose a practical way of generating effective smooth triggers that do not exhibit high-frequency artifacts and provide actionable insights into their detectability.

2. Related Work

Backdoor Trigger Generation. The first successful backdoor attacks on modern deep neural networks were demonstrated through the BadNets attack [11], using nature images, and the blending attack [5]. Since then, advanced attacks have been developed to improve the trigger effectiveness and stealthiness [17] as well as with various attacker models, such as inserting the backdoor directly by modifying the model's parameters without accessing the training set [18]. More recently, Sarka et al. [25] proposed to utilize GANs to synthesize triggers to achieve a more robust stealthiness. In this work, we analyze all these attacks in the frequency domain and find they all exhibit high-frequency components that distinguish them from their corresponding benign untriggered images.

Backdoor Data Detection. Prior work on backdoor data detection has either attempted to identify outliers directly in the input space [10] or analyzed the network response given the input. Peri et al. [23] utilize the deep features of inputs to help detect poisoning labels. [3] discovered that normal and poisoned data yield different features in the last hidden layer's activations; [28] proposed a new representation to classify benign and malicious samples; [15] computes influence functions to measure the effect of each input on the output. Other approaches include using input-saliency maps such as Grad-CAM to detect if a model only relies on a certain portion of input for its prediction [6]. Contrary to prior work, which focuses on mage space or the model response given an image input with ad-hoc designs, we examine backdoor data in the frequency domain, enabling a simple yet effective method to backdoor data detection.

Poisoned Model Detection. Existing work has also studied how to detect if a given model itself is backdoored. The most recent technique is to utilize a meta-classifier trained on various benign and backdoored models [31], and this defense is shown to perform well under minimal knowledge of the attack strategy. Another popular class of techniques is based on reconstructing the trigger from the model parameters and then performs detection based on the reconstructed triggers [29, 4, 12]. However, their trigger reconstruction algorithm often assumes that the true trigger is patched locally to a clean image, making them ineffective for smooth triggers proposed in this paper. Our work contributes to these backdoored model detections by demonstrating that these techniques should be updated with models that are backdoored with smooth triggers.

Attack Invalidation. Another approach to mitigate backdoor attacks is to prevent backdoor attacks from taking effect. One way to achieve this is by training an ensemble of models and take a majority vote of their predictions [16, 13, 14]. Other techniques include using differential private training algorithm [8], and various input preprocessing [19] and data augmentation [1, 32] methods to invalidate backdoors in the model or triggers in the samples. Our work is complementary to this line of work as frequency analysis provides a simple yet effective way to screen the backdoor data and further enhance defensive techniques' robustness to backdoor attacks.

[1] https://github.com/YiZeng623/frequency-backdoor

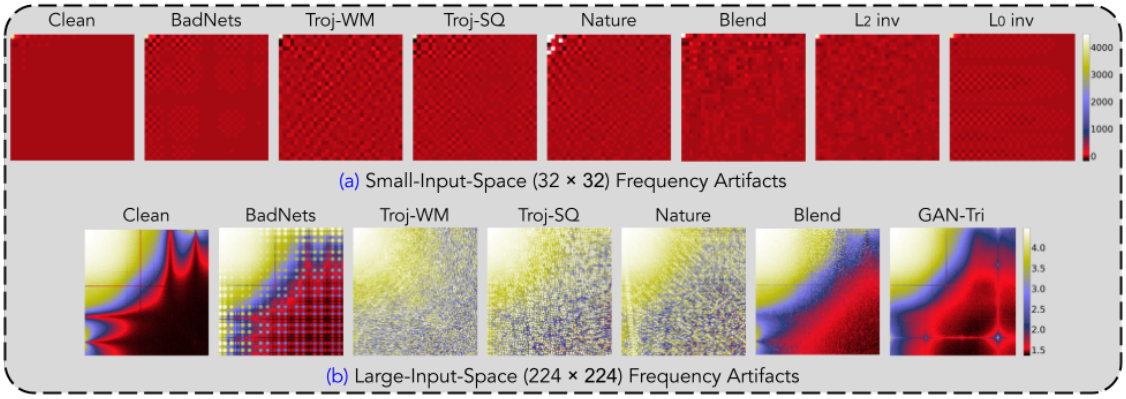

Figure 1: A side-by-side comparison in the frequency domain of clean samples vs. samples patched with triggers. The left-most heatmap in (a) depicts the mean spectrum of small-input-space data using 10000 samples randomly selected from the CIFAR-10 dataset. The left-most heatmap in (b) illustrates the mean spectrum of large-input-space data using 1000 samples randomly chosen from the PubFig dataset. The rest images show the mean frequency values of images patched with different backdoor attack triggers. All the frequency results of (b) are depicted from 1.5 to 4.5 using value clipping and exponential calculation for better visualization.

3. Frequency Artifacts

Today's backdoor attacks constantly develop the triggers to look as inconspicuous as possible in the **image domain**. We take inspiration from the success of frequency-based GAN-generated fake image detection [9] and examine these existing triggers in the **frequency domain**.

3.1. Preliminaries

We utilize the *Discrete Cosine Transform* (DCT) to convert images to the frequency domain. Closely related to the Discrete Fourier Transform, DCT represents an image as a sum of cosine functions of varying magnitudes and frequencies. This paper uses the type-II 2D-DCT, a standard tool adopted in image compression algorithms such as JPEG. The full 2D-DCT algorithm is provided in the Appendix.

Similar to previous work [9], we plot the DCT spectrum as a heatmap, where the magnitude of each pixel indicates the coefficient of the corresponding spatial frequency. The heatmap's horizontal and vertical directions correspond to frequencies in the x and y directions, respectively. The heatmap's top-left region corresponding to low frequencies, and the right bottom area corresponds to higher frequencies. Due to the energy compaction ability of the DCT, the coefficients drop quickly in magnitude when frequencies increase. Natural images typically have most of the energy concentrated in the low-frequency part [2, 27].

3.2. Examining Images with Triggers using DCT

We examine the DCT spectrum of the following triggers: *BadNets white square trigger* (BadNets) [11], *Trojan watermark* (Troj-WM) [18], *Trojan square* (Troj-SQ) [18], *hello kitty blending trigger* (Blend) [5], *nature image contains semantic information as the trigger* (Nature) [5], l_2 *norm constraint invisible trigger* (l_2 inv) [17], l_0 *norm constraint hidden trigger* (l_0 inv) [17], and *GAN generated fake facial character as the trigger* (GAN-Tri) [25]. This set of triggers encompasses the two general ideas of designing triggers in existing works: patching visible patters of commonplace objects and injecting invisible perturbations.

Figure 1 compares the frequency spectrum between clean images and the images patched with different triggers. The two heatmaps are generated by data sampled from CIFAR-10 (small-input-space) and PubFig (large-input-space). We omit l_2 inv and l_0 inv triggers for PubFig by following the same settings of [17] and acquire the optimal fooling results focused on small-input-spaces. We omit GAN-Tri for CIFAR because its small input space does not allow effective trigger generation based on GANs.

The left-most heatmaps from Figure 1 represent the DCT spectrum over clean data. Multiple classic studies [2, 27] have observed that the average spectra of natural images tend to follow a $\frac{1}{f^\alpha}$ curve, where f is the frequency along a given axis and $\alpha \approx 2$. Similar to previous findings, our results show that the low frequencies contribute the most to the image, and the contribution gradually decreases as we move towards higher frequencies. Intuitively, since colors mainly change gradually in images, sudden changes in pixel values (e.g., edges in images) are scarce, low-frequency components dominate the frequency spectrum of clean data.

However, in comparison to the spectrum of clean images, images patched with different triggers all contain strong high-frequency components. We also evaluate spectral heatmaps for other datasets, including *German Traffic Sign Recognition Dataset* (GTSRB) [26], *Chinese Traffic Sign Database*[2] (TSRD) and the high-frequency artifacts of

[2]http://www.nlpr.ia.ac.cn/pal/trafficdata/recognition.html

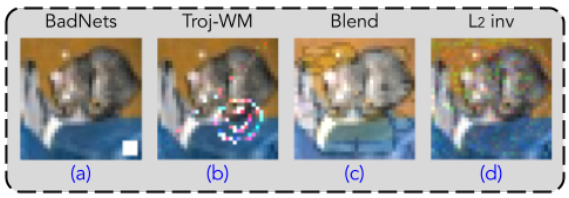

Figure 2: Examples of local patching triggers and large-size or global patching triggers

inserting triggers persist across these datasets as well. We will leave the results for these datasets in the Appendix.

3.3. Analyzing Causes of High-Frequency Artifacts

We look into the causes of the severe, persistent high-frequency artifacts observed earlier. We discuss the causes from two different perspectives, representing the two different ways of generating backdoor data: additively patching a trigger and GAN-based generation. Existing triggers applied to images via patching can be further divided into two classes: local triggers (e.g., BadNets, Nature, l_0 inv, Troj-SQ) and global triggers (e.g., l_2 inv, Blend, Troj-WM).

Local Patching. Localized triggers can be formalized as $p = T + mask \times orig$, where p is the patched data, T is the trigger, $orig$ is the original image, and $mask$ suppresses the pixel values in the part of the original image where the trigger would be placed. By the time-frequency duality, localized triggers have unlimited bandwidth, thereby carrying significant high-frequency components per se. By the linearity of DCT transformation, adding a trigger into an image is equivalent to adding the trigger's frequency spectrum into the spectrum of the image. Hence, the patched image also exhibits a large number of high-frequency components (see an example in Figure 2 (a)).

Large-Size or Global Patching. For images patched with large-size triggers, their high-frequency artifacts result from either decreased correlation between neighboring pixels or the intrinsic high-frequency artifacts carried by the trigger. For instance, Troj-WM (Figure 2 (b)) directly stamps the trigger onto the original data, or $p = T + orig$. Since the trigger pattern has very weak correlations with the original image's pixels in the trigger's vicinity, one can only use high-frequency functions to approximate the patched data. The Blend attack (Figure 2 (c)) patches with some small weight use an arbitrary clean image as the trigger. The Blend attack's high-frequency artifacts result from combining two unrelated images, which could induce a larger variation of neighboring pixels. l_2 inv (Figure 2 (d)) triggers intrinsically are high-frequency perturbations. Thus, patching them onto clean images would directly leave marks in the high-frequency domain.

GAN-Generated Backdoor Data. GAN-Tri utilizes fake facial characteristics generated with GANs (e.g., smiles) to poison the training data and conduct the backdoor attack. Since a GAN generator maps a low-dimensional latent space to a higher-dimensional data space, upsampling is widely used in GAN architectures. Prior work [9] has shown that the upsampling operations employed in GANs cause inevitable high-frequency artifacts.

4. Frequency-Based Backdoor Data Detection

This section describes our experiments to demonstrate that analyzing the frequency domain can effectively distinguish backdoored data from a poisoned dataset. We use the *Accuracy* (ACC) and the *Backdoored data Detection Rate* (BDR) as the evaluation metrics to demonstrate the separability between clean data and backdoor data. A higher BDR means more effective rejection of backdoor samples.

Attacker Model. We consider the most potent attacker model, where the attackers have full knowledge of the training set, the inference set, and the potential target model. The attacker can achieve the backdoor attack by either poisoning the training set with samples containing the trigger or directly modifying the target model's weights to insert the backdoor into the DNN. The triggers would then be patched onto the clean samples during the inference time to cause the model to output the target label to complete the attack.

4.1. Detection Method and Application Scenarios

In light of the severe, persistent high-frequency artifacts of existing backdoor triggers observed earlier, we adopt a supervised learning approach to differentiate between clean and backdoor data. To simulate the poison data, we manipulate the clean samples to approximate the high-frequency artifacts that triggers might exhibit. We then create a training set that contains DCT transformations of clean samples and samples with digital manipulations. The digital manipulations used to alter the clean samples includes: **1)** random white block: patching a white rectangle of random size onto a random location of the image; **2)** random colored block: adding a rectangle of random size and random value to a random place; **3)** adding random Gaussian noise; **4)** random shadow: drawing random shadows of random shape across the images; **5)** random blend: randomly selecting another sample from the dataset, multiplying it with a small value, and patching with the current data. These perturbations are chosen because they follow the same general methodology as the backdoor attacks. The visual results of each digital manipulation can be found in the Appendix.

The detector based on frequency artifacts can be applied to both attack scenarios: poisoning the training set or directly tuning the weights. We focus on developing an ac-

curate trigger data detector that can effectively reject triggers during inference. For the scenario where triggers are used to poison the model during training, the detector can also be deployed during training to reject potential poisoned data. We aim to build an attack agnostic detector with zero prior knowledge of the trigger pattern or the target model in both scenarios. This defense case is the most comprehensive scenario aiming to thwart existing backdoor attacks in a trigger-agnostic manner.

When building our detector, we consider the difference in input space and study small input spaces (e.g., CIFAR-10) and larger input spaces (e.g., PubFig) separately. We find that attack triggers in larger input spaces (larger than 160 pixels in width) are more easily linearly separable. This experiment's details showing the trade-off between input size and linear separability are presented in the Appendix. Also included in the Appendix are details about the detector model architectures and our model ablation study results.

4.2. Results & Comparison

Experiment Setup. This section evaluates the detection framework assuming we have full access to a clean dataset with a similar distribution as the inference data. We explore the results of our detection framework across datasets in the following subsection. For each experiment, we use the entire original training set to develop the DCT processed dataset consisting of an equal number of clean samples and randomly perturbed samples. To test the detector's efficiency, we use a test set consisting of half clean samples and half poisoned samples patched with the backdoor attack trigger we wish to evaluate (e.g., BadNets, Nature). None of the triggers evaluated in the test set are present in the training set. The results over the CIFAR-10, GTSRB, and PubFig datasets are presented in Table 1. The regenerated CIFAR-10 training set consists of 100,000 samples (50,000 clean samples and 50,000 randomly perturbed samples) for training and 20,000 samples for testing; GTSRB training set contains 70,576 images and 25,260 images for the test; PubFig contains 22140 samples in the training set and 2,768 images in the test set. As a comparison group, we have also included the results when distinguishing samples in the image domain without DCT. Full details of the experiments and models used can be found in the Appendix.

Results. As shown in Table 1, the detector can achieve a high BDR (98.5% average across all cases) on all the evaluated triggers when built based on the frequency domain. On the other hand, detection does not work well with image domain data (represented with a * in Table 1). We observe an increase in the BDR but a drop in the average ACC using the image data from the PubFig dataset versus on CIFAR-10 and GTSRB, indicating the BDR improvement on the PubFig dataset causes a higher false-positive rate.

	BadNets	Troj-WM	Troj-SQ	Nature	Blend	l_2 inv	l_0 inv
ACC	**94.10**	**98.85**	**98.76**	**98.66**	**97.00**	**98.85**	**98.86**
BDR	**90.50**	**99.99**	**99.82**	**99.61**	**96.30**	**99.99**	**100**
ACC*	49.76	85.17	55.37	54.19	64.52	77.31	49.08
BDR*	1.38	72.19	12.59	10.24	30.90	56.46	0.00

	BadNets	Troj-WM	Troj-SQ	Nature	Blend	l_2 inv	l_0 inv
ACC	**90.23**	**93.96**	**93.93**	**91.46**	**93.67**	**93.96**	**93.93**
BDR	**92.55**	**100**	**99.94**	**95.00**	**99.43**	**100**	**99.94**
ACC*	48.92	57.43	48.61	49.35	80.63	89.53	48.40
BDR*	17.42	31.51	16.92	18.15	69.91	84.65	16.57

	BadNets	Troj-WM	Troj-SQ	Nature	Blend	GAN-tri
ACC	**97.74**	**99.29**	**99.29**	**99.29**	**99.29**	**93.96**
BDR	**96.94**	**100**	**100**	**100**	**100**	**100**
ACC*	53.05	52.55	57.35	60.29	62.27	50.27
BDR*	72.27	72.40	82.01	87.90	91.80	68.30

Table 1: The detection efficiency and comparisons on CIFAR-10 (top), GTSRB (middle) and PubFig (bottom). * represents the comparison group using the image domain data.

Remark 1. *We find severe high-frequency artifacts across existing backdoor triggers can be utilized to provide accurate detections. While detecting backdoor triggers using the image domain provides limited outcomes, utilizing the frequency domain can accurately reject backdoored data without sacrificing much of the clean samples.*

4.3. Transferability

This section evaluates the transferability of the frequency-based detector towards new datasets. The training set develops the same way as the above experiments. We then test the detector's transferability from a CIFAR-10 model to the GTSRB dataset (Table 2). The transferability of a model trained on GTSRB and a model trained on CIFAR-10 to the TSRD dataset (Table 3) is also tested.

	GTSRB		CIFAR-10		CIFAR-10+Tune	
Attack	ACC	BDR	ACC	BDR	ACC	BDR
BadNets	90.23	92.55	68.23	99.61	89.44	95.95
Troj-WM	93.96	100	68.42	99.99	91.47	100
Troj-SQ	93.93	99.94	68.40	99.96	91.44	99.95
Nature	91.46	95.00	67.79	98.75	94.03	97.08
Blend	93.67	99.43	66.51	96.18	64.49	45.67
l_2 inv	93.96	100	68.40	99.95	91.45	99.97
l_0 inv	93.93	99.94	68.41	99.98	91.46	99.99

Table 2: The transferability using the detector trained on different datasets tested on the GTSRB dataset (%).

Table 2's column headers indicate the training set used to train the specific detector. For the last column (CIFAR-10+Tune), we first train using the CIFAR-10 dataset, then fine-tune with a 200-sized dataset (half clean, half randomly perturbed originating from the 100 clean samples from the GTSRB test set) of the same distribution as the GTSRB. In real life, as the defender is on the user's side, they will have access to the inference data, and a fine-tuning of the model using 100 clean samples is reasonable and practical.

Note that the samples we use to fine-tune the models are not utilized in the test set for all experiments.

	GTSRB		GTSRB+Tune		CIFAR-10		CIFAR-10+Tune	
Attack	ACC	BDR	ACC	BDR	ACC	BDR	ACC	BDR
BadNets	57.99	86.83	77.01	87.10	61.17	98.01	82.53	89.83
Troj-WM	64.57	100	83.46	100	62.16	100	87.10	98.97
Troj-SQ	64.57	100	83.46	100	62.16	99.95	87.58	99.93
Nature	60.09	91.03	83.11	99.29	59.30	94.28	79.61	83.98
Blend	59.04	88.94	82.92	98.92	55.37	86.41	83.62	92.01

Table 3: The transferability on the TSRD dataset (%).

When comparing the original GTSRB detector and the CIFAR-10 detector on the GTSRB, we see a significant drop in ACC resulting from the variance between the two datasets' data distribution. However, by fine-tuning the detector using the 200-sized dataset, one can achieve a higher ACC without sacrificing too much in the BDR. The detection efficiency is close to or even surpasses the detector's results with the original GTSRB training set on some attacks. The Blend attack is a particular case here, as the fine-tuned results worsen. We propose the main reason behind this is that the two datasets are of significant variance in distributions. This side effect over the detection deficiency against Blend is recovered in the following experiments using pairs of training and testing sets with closer distributions.

Table 3 presents the results on evaluating the detector's transferability from the GTSRB dataset and CIFAR-10 dataset onto the TSRD dataset. Due to the limited size of the TSRD dataset, we cannot achieve satisfying accuracy using the target model we present in the Appendix; therefore, the TSRD dataset is only for testing. The raw detector results are similar to the experiment evaluating the CIFAR-10 model over GTSRB test data. After fine-tuning with 100 TSRD clean samples (dataset of size 200), both detectors can achieve satisfying detection results with acceptable ACC on the TSRD dataset. Of note, after fine-tuning, both detectors' performances against the Blend attack over the TSRD dataset are better than the results from the previous experiment's over the GTSRB dataset. We believe this is because of closer similarities in distribution between the datasets than between CIFAR-10 and GTSRB.

	Combined		Combined+Tune	
Attack	ACC	BDR	ACC	BDR
BadNets	64.28	89.88	80.28	89.80
Troj-WM	69.34	100	85.28	99.80
Troj-SQ	69.34	100	85.36	99.95
Nature	64.67	90.66	83.29	95.82
Blend	64.18	89.68	84.61	98.45

Table 4: The transferability with extended training set, tested using the TSRD dataset(%).

We also notice the CIFAR-10 detector achieves higher accuracy than the GTSRB detector on the TSRD dataset for most cases, even though CIFAR-10 and TSRD have disparent sample categories. Given that the two detectors are all trained with the same number of epochs and settings, we deduce that the transferability is related to the training set's size. This assumption is confirmed in the following experiment when evaluating the transferability using a combined training set of CIFAR-10 and GTSRB. As shown in Table 4, when using a combined dataset, we can see an improvement in the average detection efficiency over the TSRD dataset.

Remark 2. *Since the high-frequency artifacts of existing triggers are universal across different datasets, transfer learning can be adopted in the task of detecting backdoored samples in the frequency domain. Even if the defender does not have access to the original training set, they can still effectively detect attacks and achieve satisfying results in the frequency domain by adopting large public clean datasets to conduct transfer learning.*

5. Creating Smooth Triggers

5.1. Problem Defination

Given existing attacks' high-frequency artifacts, this section aims to create triggers invisible in high-frequency but stay efficient as backdoor triggers. We summarize generating a smooth trigger as a bilevel optimization problem:

$$\min_{\delta} L(x_i + \delta, y_t; \theta_p) + \lambda \Omega(\delta; g), \tag{1}$$

$$s.t.\ \ x_i + \delta \in [0,1]^n, \tag{2}$$

$$\theta_p = \text{argmin}_\theta \Sigma_i L(x_i, y_i; \theta) + \Sigma_i L(x_i + \delta, y_t; \theta), \tag{3}$$

We adopt $\Omega(\cdot; g)$ from SmoothFool [7] to measure the input sample's roughness given a preset low-pass filter in the image domain g. λ is the Lagrangian coefficient that controls the trade-off between smoothness and perturbation scale. Equation (1) is the optimization problem that tries to minimize both the loss of the poisoned data given a trained poisoned model and the roughness of the trigger itself. Equation (2) ensures the poisoned samples falls within the rational range from $[0, 1]$. Equation (3) is the optimization problem to train a poisoned model where θ_p is the poisoned model, and θ is an initialized target model.

5.2. Methodology

There are two ways to achieve the constraint of smoothness with the low-pass filter. One way is to conduct the search iteratively and output the results when it meets the constraint. However, we find this methodology is ineffective in our case as optimization along the gradient of DNNs causes local impulses in the triggers that easily exceed the constraint. Therefore, we adopt a strategy by updating the smooth trigger with the perturbation that remains

after the low-pass filter for each iteration, thus meeting the constraint. The remaining parts of the perturbation from the filter can be interpreted as $r = \delta * g$. Here, r is the result of the perturbation after convolving with the low-pass filter, g, in the image domain. Taking Equation (2) into account and the fact that the triggers are of small values after passing through g, we adopt a min-max scaler, M, as a normalization process to remap the poison data onto the rational range of an image, $[0, 1]$. Instead of using the rigid value clipping done in other works, we argue that normalization can better keep the relative scale between each pixel of the smooth trigger and better maintain functionality as a backdoor trigger. Consequently, we can rewrite the optimization as:

$$\min_{r} L(x_{poi}, y_{tar}; \theta_{poi}), \tag{4}$$

$$s.t.\ \ r = \delta * g, \tag{5}$$

$$x_{poi} = M(x_i + \lambda r), \tag{6}$$

$$\theta_{poi} = \theta \Sigma_i L(x_i, y_i; \theta) + \Sigma_i L(x_{poi}, y_{tar}; \theta), \tag{7}$$

This bilevel optimization function's objective is to find a smooth pattern r within the range of the low-pass filter g that can be successfully adopted as a backdoor trigger. As stated in our paper's scope, the classifier θ is a DNN, thus making the optimization problem non-convex [21]. Thus, we propose Algorithm 1 to approximate a solution to this problem: we heuristically search for a smooth pattern that leads clean samples to the target label.

Algorithm 1 explains the procedure of generating a smooth trigger. $Err(\cdot)$ computes the error rate, and $Domi(\cdot)$ output the mode of the labels that are different from their original ones. The algorithm first initializes a random target label and a zero-image as the trigger. While the error caused by the generated trigger is below the desired fool rate γ, the algorithm will iteratively compute the perturbation according to the gradients of a pre-trained model towards the target class for each sample that is not of the target label. The attained perturbation then passes through a low-pass filter to remove high-frequency parts. The smoothed perturbation is added to the trigger to update the smooth trigger. Finally, we select out a subset from all the data points to quickly estimate the new error rate. If the estimated error rate is larger than the preset threshold, we will update the best smooth trigger pairing with the dominant label. Upon experiments of generating a unified perturbation aiming to cause universal misclassification [21], there exist several dominant labels that perturbations tend to lead to. We compute the dominant label as the target label and pair it with the corresponding smooth trigger to achieve a more potent backdoor attack.

Algorithm 1: Generating a Smooth Trigger

Input: Data Points: $X \in R^{N \times H \times W \times C}$;
Pre-trained Classifier: θ;
Desired Fooling Rate: γ;
Output: Smooth Trigger: r; Dominante Label: y_{tar};
Parameters: Low-pass Filter g; Trade-off Controller: λ;
Number of Classes: K

```
/* Initialization */
r ← 0^{H×W×C};
y_tar ← randint(K);
γ^best ← Err(X);
while γ^best < γ do
    for each data point x_i ∈ X do
        if θ(M(x_i + λr))! = y_tar then
            /* Computing Purturbation */
            δ = −∇L(x_i, y_tar; θ);
            /* Low-Pass Filter */
            r = r + δ * g;
            r = r * g;
        end
    end
    X_poi = M(subset(X) + r);
    y_tar = Domi(X_poi);
    if Err(X_poi) > γ^best then
        /* Updating the Best Result */
        γ^best ← Err(X_poi);
        r^best ← r;
        y_tar^best ← y_tar;
    end
end
return r^best, y_tar^best
```

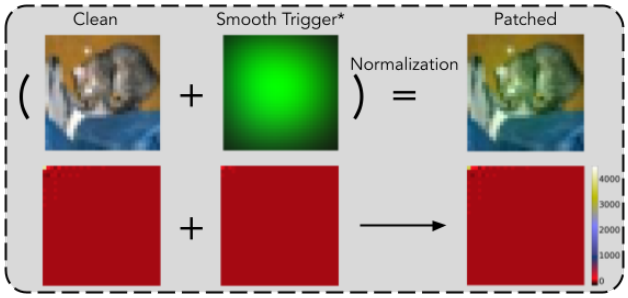

Figure 3: Visual effects over image and frequency domian of the smooth triggers. The trigger is multiplied by 5 for visualization. The right bottom depicts the heatmap averaged over 10000 samples patched with the smooth trigger. Both the trigger itself and the final images exhibit frequency spectra similar to natural images.

5.3. Attack Results and Evaluations

Figure 3 depicts the computed smooth trigger's visual effects using the proposed algorithm in the image domain and frequency domain. A similar figure illustrating the smooth trigger generated based on the GTSRB dataset is presented in the Appendix. As one can see from the frequency results, neither the trigger itself nor the final patched image contain any high-frequency components.

We now evaluate the smooth trigger's functionality as a backdoor trigger by using it to poison the training set and conduct the entire backdoor attack pipeline. We adopt a small CNN trained on CIFAR-10 with an ACC of 85.50% as the baseline model. Then, following Algorithm 1, we use the model to acquire the smooth trigger.

The smooth attack can attain an *Attack Success Rate* (ASR) around 95% within one epoch of training while the model's training accuracy is still below 30%. This effect indicates the smooth trigger contains features that are easier to pick up by the DNN. We evaluate the final result when the model converges over the poison dataset with a poison ratio of 0.1[3]. The poisoned model recognizes the trigger by 97.25% of chance and achieves an ACC on clean samples at 84.54%, which is close to the baseline ACC.

As a comparison, we test the case of using random patches and nature images passed through the low-pass filter as naive designs of the smooth triggers. The triggers can only reach an average ASR of 75.54%. Meanwhile, we observe that the naive-designed smooth triggers take more epochs for the model to converge. The averaging ACC over clean samples can only achieve 76.29%, with five naive-designed smooth triggers considered. This drop in the performance over the clean samples can also impair the stealthiness of the attack. A similar result can be witnessed on the GTSRB dataset shown in the Appendix. Thus, we conclude that our smooth trigger maintains functionality as a backdoor trigger while leaving no high-frequency artifacts.

Remark 3. *Directly using random patches passed through the low-pass filter cannot generate smooth triggers of satisfying functionality. We show that by approximately solving a bilevel problem, one can generate smooth triggers that function as backdoor triggers while achieving a satisfying stealthiness in both image and frequency domains.*

5.4. Impacts over Defenses

To show the importance of considering smooth triggers in defenses, we perform a small case study on *Meta Neural Analysis* (MNA) [31], a state-of-art defense mechanism. When faced with a classifier poisoned with a smooth trigger, the MNA can only achieve an AUC score of 0.0776. However, after upgrading the MNA to consider the smooth trigger generation, the upgraded MNA can achieve an AUC score of 0.694 and a detection accuracy of 42.85%. This simple case study illustrates how existing defenses can be made more robust by considering smooth triggers.

Similarly, we also aim to upgrade our proposed detector with smooth triggers. We first try to finetune the detector with samples patched with patterns passed through the low-pass filter. Although the detector successfully detects samples patched with the same trigger with 95.67% accuracy,

[3]This is a standard poison rate used in other attack works [17, 18].

Figure 4: Illustration of fine-tuning over the smooth trigger

the detector fails to generalize and cannot detect other filtered triggers nor the smooth trigger. We next experiment using the smooth trigger we acquired using Algorithm 1 to finetune the model for one epoch with 20,000 samples (half clean, half patched). This time, we find the model performs well on detecting the smooth trigger (82.49% accuracy) and attains a higher detection rate of 89.37% averaged over all unseen low-pass filtered triggers. With this detection rate, the detector can constrain the overall ASR of the most potent smooth trigger found using Algorithm 1 to 19.72%. If we can use the detector to eliminate poisoned samples in the training set, we further drop the overall ASR to 18.03%.

We design an experiment comparing the distance in the hyperplane between clean samples and samples patched with filtered triggers (including the smooth trigger and other simple designs) to better explain this generalizability. We take the detector's last layer's weight on the benign class and compu te the Euclidean distance between the weights and the clean samples' logits to select the "representative" of the clean cluster in the hyperplane. We then feed the poisoned samples patched with different kinds of low-pass filter processed triggers to acquire the average distance between the clean representative and the poisoned samples' clusters. We find that the smooth trigger patch samples have the closest distance of 4.3589 among all the filtered triggers. Figure 4 helps explain the generalizability acquired by fine-tuning the detector using the smooth trigger. With a closer distance toward the clean sample center, the smooth trigger-patched samples can work as support vectors in the hyperplane to include other filtered triggers, thus achieving universal generalizability.

Remark 4. *We show that defenses designed with the frequency domain considered can better mitigating the smooth triggers. We bring attention to the development of frequency-constraint triggers, as they can be adopted in an adversarial training manner to help defenses acquire robust and generalized protection against smooth triggers.*

6. Conclusion

In this work, we filled the gap in existing works on backdoor attacks and defenses by presenting a comprehensive analysis of the overlooked frequency domain. Unlike natural images, we found many existing attack triggers exhibit severe artifacts in the high-frequency spectrum. We took advantage of the artifacts and show that we can achieve an average detection rate of 98.50% under attack-agnostic settings. Realizing this limitation in the current trigger design, we proposed an effective way to generate triggers invisible in the high-frequency domain. We demonstrated its potency in terms of stealthiness and attack efficiency. Finally, we showed that existing backdoor defenses could benefit from considering frequency-invisible attacks. We hope the remarks and solutions proposed in this paper can inspire more advanced studies on backdoor attacks in the future.

References

[1] E. Borgnia, V. Cherepanova, L. Fowl, A. Ghiasi, J. Geiping, M. Goldblum, T. Goldstein, and A. Gupta. Strong data augmentation sanitizes poisoning and backdoor attacks without an accuracy tradeoff, 2020.

[2] G. J. Burton and I. R. Moorhead. Color and spatial structure in natural scenes. *Applied optics*, 26(1):157–170, 1987.

[3] B. Chen, W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, and B. Srivastava. Detecting backdoor attacks on deep neural networks by activation clustering. *arXiv preprint arXiv:1811.03728*, 2018.

[4] H. Chen, C. Fu, J. Zhao, and F. Koushanfar. Deepinspect: A black-box trojan detection and mitigation framework for deep neural networks. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 4658–4664. International Joint Conferences on Artificial Intelligence Organization, 7 2019.

[5] X. Chen, C. Liu, B. Li, K. Lu, and D. Song. Targeted backdoor attacks on deep learning systems using data poisoning, 2017.

[6] E. Chou, F. Tramèr, and G. Pellegrino. Sentinet: Detecting localized universal attacks against deep learning systems. In *Deep Learning and Security Workshop*, 2020.

[7] A. Dabouei, S. Soleymani, F. Taherkhani, J. Dawson, and N. Nasrabadi. Smoothfool: An efficient framework for computing smooth adversarial perturbations. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2665–2674, 2020.

[8] M. Du, R. Jia, and D. Song. Robust anomaly detection and backdoor attack detection via differential privacy. *arXiv preprint arXiv:1911.07116*, 2019.

[9] J. Frank, T. Eisenhofer, L. Schönherr, A. Fischer, D. Kolossa, and T. Holz. Leveraging frequency analysis for deep fake image recognition. In *International Conference on Machine Learning*, pages 3247–3258. PMLR, 2020.

[10] Y. Gao, C. Xu, D. Wang, S. Chen, D. C. Ranasinghe, and S. Nepal. Strip: A defence against trojan attacks on deep neural networks. In *Proceedings of the 35th Annual Computer Security Applications Conference*, ACSAC '19, page 113–125, New York, NY, USA, 2019. Association for Computing Machinery.

[11] T. Gu, B. Dolan-Gavitt, and S. Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

[12] W. Guo, L. Wang, X. Xing, M. Du, and D. Song. Tabor: A highly accurate approach to inspecting and restoring trojan backdoors in ai systems, 2019.

[13] J. Jia, X. Cao, and N. Z. Gong. Intrinsic certified robustness of bagging against data poisoning attacks, 2020.

[14] J. Jia, X. Cao, and N. Z. Gong. Certified robustness of nearest neighbors against data poisoning attacks, 2021.

[15] P. W. Koh and P. Liang. Understanding black-box predictions via influence functions. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1885–1894, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[16] A. Levine and S. Feizi. Deep partition aggregation: Provable defense against general poisoning attacks, 2020.

[17] S. Li, B. Z. H. Zhao, J. Yu, M. Xue, D. Kaafar, and H. Zhu. Invisible backdoor attacks against deep neural networks. *arXiv preprint arXiv:1909.02742*, 2019.

[18] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang. Trojaning attack on neural networks. In *25nd Annual Network and Distributed System Security Symposium, NDSS*. The Internet Society, 2018.

[19] Y. Liu, Y. Xie, and A. Srivastava. Neural trojans. In *2017 IEEE International Conference on Computer Design (ICCD)*, pages 45–48. IEEE, 2017.

[20] S. Ma, Y. Liu, G. Tao, W. Lee, and X. Zhang. Nic: Detecting adversarial samples with neural network invariant checking. In *NDSS*, 2019.

[21] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1765–1773, 2017.

[22] A. Paudice, L. Muñoz-González, A. Gyorgy, and E. C. Lupu. Detection of adversarial training examples in poisoning attacks through anomaly detection, 2018.

[23] N. Peri, N. Gupta, W. R. Huang, L. Fowl, C. Zhu, S. Feizi, T. Goldstein, and J. P. Dickerson. Deep k-nn defense against clean-label data poisoning attacks, 2020.

[24] A. Saha, A. Subramanya, and H. Pirsiavash. Hidden trigger backdoor attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11957–11965, 2020.

[25] E. Sarkar, H. Benkraouda, and M. Maniatakos. Facehack: Triggering backdoored facial recognition systems using facial characteristics. *arXiv preprint arXiv:2006.11623*, 2020.

[26] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In *The 2011 international joint conference on neural networks*, pages 1453–1460. IEEE, 2011.

[27] D. Tolhurst, Y. Tadmor, and T. Chao. Amplitude spectra of natural images. *Ophthalmic and Physiological Optics*, 12(2):229–232, 1992.

[28] B. Tran, J. Li, and A. Madry. Spectral signatures in backdoor attacks. In *Advances in Neural Information Processing Systems*, pages 8000–8010, 2018.

[29] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy)*, pages 707–723, 2019.

[30] E. Wenger, J. Passananti, Y. Yao, H. Zheng, and B. Y. Zhao. Backdoor attacks on facial recognition in the physical world. *arXiv preprint arXiv:2006.14580*, 2020.

[31] X. Xu, Q. Wang, H. Li, N. Borisov, C. A. Gunter, and B. Li. Detecting ai trojans using meta neural analysis. *arXiv preprint arXiv:1910.03137*, 2019.

[32] Y. Zeng, H. Qiu, S. Guo, T. Zhang, M. Qiu, and B. Thuraisingham. Deepsweep: An evaluation framework for mitigating dnn backdoor attacks using data augmentation. *arXiv preprint arXiv:2012.07006*, 2020.

Appendix

A. Type-II 2D-DCT Algorithm

The type-II 2D-DCT is given by a function $D : \mathbb{R}^{N1\times N2} \rightarrow \mathbb{R}^{N1\times N2}$ that maps an image data $\{g_{x,y}\}$ to its frequency representation $D = \{D_{k_x,k_y}\}$ with $D_{k_x,k_y} = w(k_x)w(k_y)\sum_{x=0}^{N_1-1}\sum_{y=0}^{N_2-1} g_{x,y} cos\left[\frac{\pi}{N_1}(x+\frac{1}{2})k_x\right] cos\left[\frac{\pi}{N_2}(y+\frac{1}{2})k_y\right]$, for $\forall k_x = 0, 1, ..., N_1 - 1$ and $\forall k_y = 0, 1, ..., N_2 - 1$, where $w(0) = \sqrt{\frac{1}{4N}}$ and $w(k) = \sqrt{\frac{1}{2N}}$ for $k > 0$.

B. Visual Examples of Different Triggers

We provide the pair-to-pair comparisons of samples patched with different triggers' visual effects in the image and frequency domain. Figure 7, 8 illustrate the comparison of the attack cases over the GTSRB and the TSRD dataset. We can see severe high-frequency artifacts similar to the CIFAR-10 dataset results presented in Section 3.2. We also provide the pair-to-pair extended comparison of both the image and frequency domain visual effects on the evaluated CIFAR-10 and PubFig dataset in Figure 9, 10. Those results over different datasets and different triggers are provided here to further support the existence of persistent high-frequency artifacts of previous backdoor attacks in Section 3.2.

C. Visual Examples of the Random Puturbation used in Developing the Detector

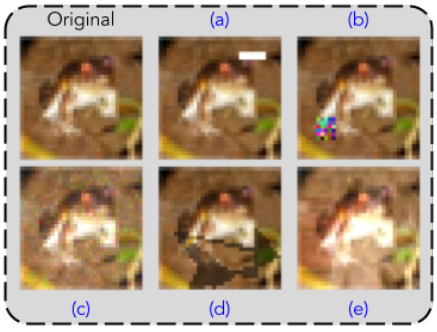

Figure 5: Visual examples of the random purturbations adopted in developing the detector. The upper left sample is a clean example, (a)-(e) are the perturbed results using different approaches.

Figure 5 presents the visual examples of the random perturbation results mentioned in Section 4.1. Figure 5 (a) is the example of patching a white rectangle of random size onto a random location of the image; Figure 5 (b) is the result of patching a rectangle of random size and random value to a random place. Those two random perturbations simulate patching localized triggers as mentioned and analyzed in Section 3.3. Figure 5 (c) is the visual result of adding random Gaussian noise; the result of drawing a random shadow of random shape is depicted in Figure 5 (d); finally, 5 (e) shows the visual result of random blend.

Note that the random perturbations used in Section 4.1 as illustrated here are of different shape and values from the tested triggers. We only use those random perturbations to simulate the resulting high-frequency artifacts using the two major patching methods, as analyzed in Section 3.3.

D. Linear Separability & Input Space

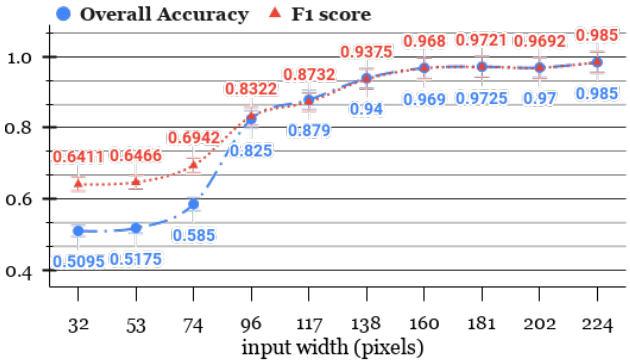

Figure 6: Detection Efficiency Using the Linear Model vs. Input Width

As mentioned in Section 4.1, we look into the relationship between the input space's size and linear models' efficiency. We test the F1-score and the linear models' overall accuracy on detecting triggered samples using different-input-spaced PubFig datasets. We test ten different values ranging from 32 to 224. The relationship between the input width and the detection efficiency is depicted in Figure 6. We can tell from the results that a larger-input-space samples can more easily be used to conduct a linear separation of the benign samples and the triggered samples. Meanwhile, the small-input-spaced samples are harder to be separated with linear models. Intuitively, we conduct the DCT of the whole image, thus acquiring a result of the same size as the image domain. So the larger input-spaced samples have more pixels representing the high-frequency coefficients, thus better reflecting the high-frequency artifacts when triggers are introduced. Based on the results shown in Figure 6 and as claimed in Section 4.1, an input space larger than 160 pixels can help linear models meet satisfying detection results.

E. DNN Model Architechures and Ablation Study

Given the different scales of difficulties to separate the DCT data in the frequency domain, we introduce a model ablation study to acquire the most simplistic DNN architecture that satisfies the detection performance to conduct the experiments in Section 4.1.

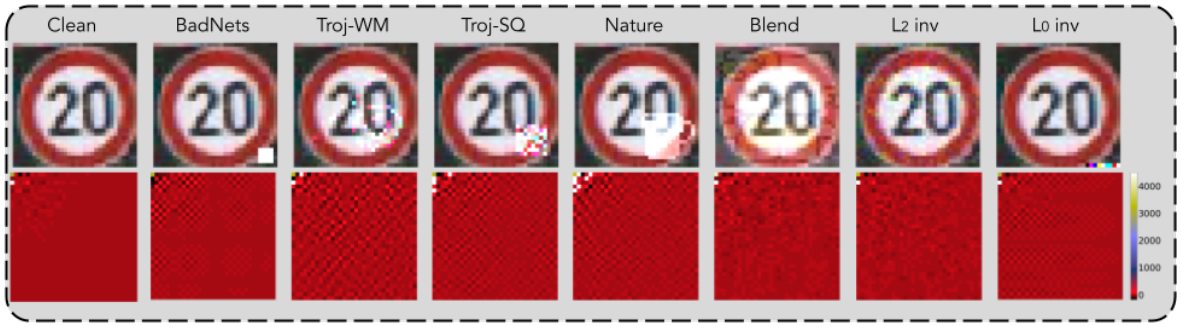

Figure 7: A pair-to-pair comparison of clean data and samples patching with different triggers on the GTSRB dataset. The frequency results are averaged over 10000 randomly selected samples from the test set.

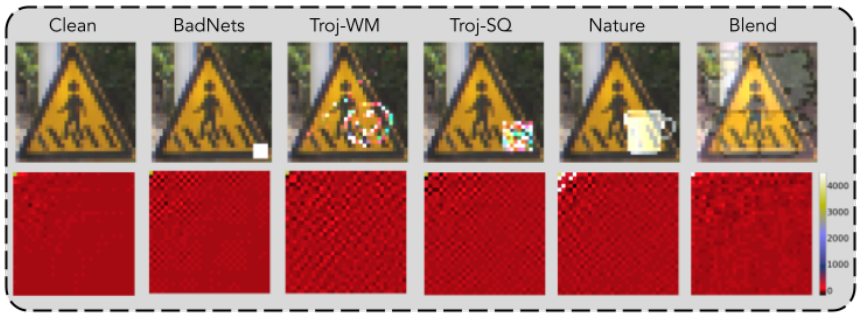

Figure 8: A pair-to-pair comparison of clean data and samples patching with different triggers on the TSRD database. The frequency results are averaged over all 4170 samples.

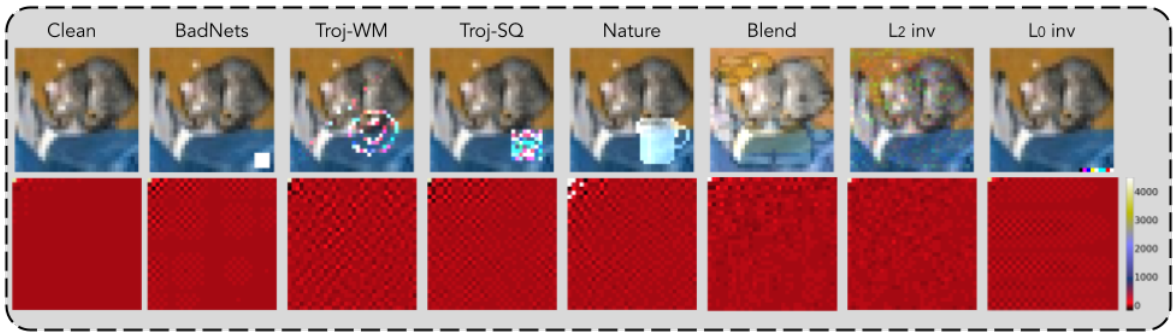

Figure 9: A pair-to-pair comparison of clean data and samples patching with different triggers on the Cifar10 dataset. The frequency results are averaged over 10000 randomly selected samples from the test set.

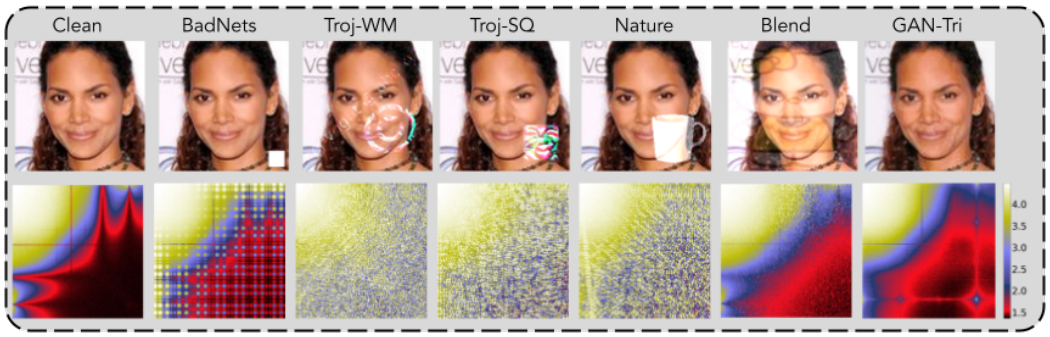

Figure 10: A pair-to-pair comparison of clean data and samples poisoned with different backdoor attacks on the PubFig dataset. The frequency results are averaged over 1000 randomly selected samples from the test set and clipped with the range of (1.5,4.5) for visualization.

Model	#Parameters	Train ACC	BadNets		Troj-WM		Troj-SQ		Nature		l_2 inv		l_0 inv	
			ACC	BDR	ACC	BDR	ACC	BDR	ACC	BDR	ACC	BDR	ACC	BDR
Linear	6,146	83.35	53.85	28.41	89.64	**100**	89.42	**99.56**	89.57	**99.85**	89.64	**100**	64.65	50.00
128-cell-hidden	393,602	88.23	54.80	21.89	**93.85**	**99.99**	**93.44**	**99.16**	**93.71**	**99.71**	**93.84**	**99.96**	55.61	23.50
3-layer CNN, $k_{max} = 32$	10,214	**95.55**	83.64	72.85	**97.21**	**99.99**	**96.94**	**99.47**	**97.09**	**99.76**	**97.21**	**99.99**	70.72	47.03
3-layer CNN, $k_{max} = 64$	31,862	**97.15**	84.26	71.72	**98.40**	**99.99**	**98.21**	**99.60**	**98.26**	**99.72**	**98.38**	**99.95**	55.71	14.61
3-layer CNN, $k_{max} = 128$	109,718	**98.36**	86.28	75.44	**98.55**	**99.99**	**98.40**	**99.68**	**98.40**	**99.67**	**98.55**	**99.99**	**97.46**	**97.80**
4-layer CNN, $k_{max} = 128$	245,014	**98.44**	87.63	78.18	**98.52**	**99.97**	**98.36**	**99.65**	**98.39**	**99.70**	**98.53**	**99.99**	**95.25**	**93.43**
5-layer CNN, $k_{max} = 128$	278,870	**98.58**	87.26	77.33	**98.52**	**99.97**	**98.38**	**99.57**	**98.44**	**99.69**	**98.58**	**99.96**	89.88	82.56
6-layer CNN, $k_{max} = 128$	292,002	**98.64**	**94.10**	**90.50**	**98.85**	**99.99**	**98.76**	**99.82**	**98.66**	**99.61**	**98.85**	**99.99**	**98.86**	**100**

Table 5: Model ablation study using the CIFAR-10 dataset. k_{max} represents the maximum value of the CNN kernels. We start the analysis from the most straightforward fully-connected linear model. Hidden layers, convolutional layers, or kernel sizes are gradually added or enlarged to test out the most simplistic model that can satisfy an outstanding detection efficiency. We present the training ACC, detection ACC, and BDR for each attack (%); the **boled** results are larger than 90%, which we interpret as satisfying results.

On large-input-spaced samples, namely the PubFig dataset, a linear model would already be able to achieve an outstanding detection efficiency which is introduced in Table 1, Section 4.2. Thus, no further ablation study is necessary for the large-input-space. The details of the linear model we adopted to conduct the detection task over the PubFig dataset are shown in Table 6. We use Adam with a learning rate of 0.01 as the optimizer for training this linear model. The binary cross-entropy is adopted as the loss function for the task of linear separation. We train the linear model with 50 epochs on the PubFig based dataset to attain the results shown in Table 1, Section 4.2.

Given that the DCT results in our evaluation have the same size as the original data's input space, the DCT results over small-input-space have a weaker ability to depict high-frequency artifacts compared to larger-input-space due to the limited number of high-frequency coefficients. Thus, as shown in Table 5, a similar fully connected linear model cannot meet a satisfying detection efficiency over the frequency domain using the same framework we proposed in this paper. We then conduct a thorough model ablation study by adding hidden layers or convolutional layers with different kernel sizes to obtain a most simplistic model that meets satisfying detection results over the evaluated attacks as shown in Table 5. With more complex architecture and parameters, the DNN can better detect the tested attacks. Based on the ablation study, we found that only until the model's architecture consists of 6 convolutional layers with $k_{max} = 128$ can it meet a satisfying and robust detection efficiency against all evaluated attacks.

Input $(224 \times 224 \times 3)$
Flatten (150528)
Dense (2)

Table 6: The network architecture of our simple Linear detector for large input space. We report the size of each layer.

The details of the simple 6-layer CNN detector for the small-input-space are explained in Table 7. The above experiments over the small-input-space are evaluated using this model to demonstrate the efficiency of conducting the detection of backdoor triggers in the frequency domain as elaborated in Section 4.2. We use Adam with a learning rate of 0.05 as the optimizer to train this model. Other settings are the same as the experiment conducted in large-input-space. The model took 150 epochs over the training set created using CIFAR-10 to converge and attain the results shown in Table 1, Section 4.2.

Input $(32 \times 32 \times 3)$
Conv2d 3×3 $(32 \times 32 \times 32)$
Conv2d 3×3 $(32 \times 32 \times 32)$
Max-Pooling 2×2 $(16 \times 16 \times 32)$
Conv2d 3×3 $(16 \times 16 \times 64)$
Conv2d 3×3 $(16 \times 16 \times 64)$
Max-Pooling 2×2 $(8 \times 8 \times 64)$
Conv2d 3×3 $(8 \times 8 \times 128)$
Conv2d 3×3 $(8 \times 8 \times 128)$
Max-Pooling 2×2 $(4 \times 4 \times 128)$
Flatten (2048)
Dense (2)

Table 7: The network architecture of our simple CNN detector for small-input-space. We report the size of each layer.

F. Target Model for Evaluating the Smooth Trigger

In Section 5.3, we evaluate the proposed smooth attack's attack efficiency on the CIFAR-10 and GTSRB dataset. As suggested in Algorithm 1, conducting the proposed attack requires a pre-trained model to generate the gradients for solving the bilevel optimization problem. We explain the details of the pre-trained model in Table 8. The model was trained using Adam optimizer with a learning rate at 0.05 for 150 epochs to converge. The base-line ACC over clean samples is 85.50% for the CIFAR-10 dataset. We also trained a base-line model on the GTSRB dataset for generating the smooth trigger over the GTSRB dataset. The GTSRB base-line model's ACC is 97.45%.

Input ($32 \times 32 \times 3$)
Conv2d 3×3 ($32 \times 32 \times 32$)
Conv2d 3×3 ($32 \times 32 \times 32$)
Max-Pooling 2×2 ($16 \times 16 \times 32$)
Conv2d 3×3 ($16 \times 16 \times 64$)
Conv2d 3×3 ($16 \times 16 \times 64$)
Max-Pooling 2×2 ($8 \times 8 \times 64$)
Conv2d 3×3 ($8 \times 8 \times 128$)
Conv2d 3×3 ($8 \times 8 \times 128$)
Max-Pooling 2×2 ($4 \times 4 \times 128$)
Flatten (2048)
Dense (10)

Table 8: The target model for evaluating the smooth trigger on Cifar10 and GTSRB dataset. We report the size of each layer.

G. Smooth Trigger on the GTSRB Dataset

As mentioned in 5.3, we conduct the smooth attack over the GTSRB dataset following the same pipeline as well. Figure 11 depicts the generated smooth trigger's visual results using the GTSRB dataset in the image and frequency domain. The dominant label computed using the Algorithm 1 is 1 on the GTSRB pre-trained model. Similar to the attack evaluation pipeline explained in Section 5.3, we conduct the backdoor attack with a poison rate of 0.1 over the target model using the GTSRB dataset. The model trained over the poisoned GTSRB dataset can maintain an ACC over clean samples at 97.42%, which is almost the same as the base-line model. Meanwhile, the ASR is 97.86% without defense. We observed the model could achieve an ASR greater than 90% even with one epoch of training. Meanwhile, the detection rate of the proposed detector in Section 4.1 can only achieve a BDR at 55.31% and an F1 score at 0.664 before considering this smooth attack. This detection efficiency can only drop the attack success rate of this GTSRB smooth trigger to 40.97%.

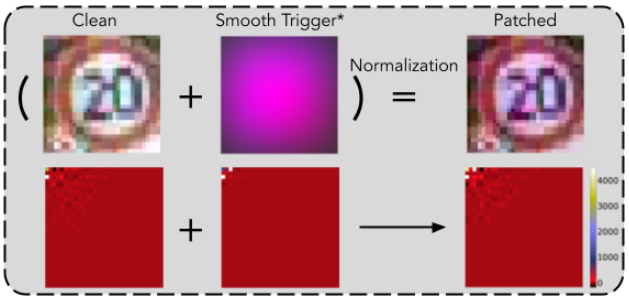

Figure 11: Visual effects over image and frequency domian of the smooth triggers. The trigger is multiplied by 5 for visualization. The right bottom depicts the heatmap averaged over 10000 samples patched with the smooth trigger. Both the trigger itself and the final images exhibit frequency spectra similar to natural images.

By incorporating this strongest smooth trigger found using Algorithm 1 into the development of the detector, we can regain a high efficient detection efficiency of a BDR at 85.53% and an F1 score of 0.8628 using the fine-tuning pipeline proposed in Section 5.4. This fine-tuning does not affect much over the other attack trigger's detection efficiency due to the limited scale as discussed in Section 5.4. Using this upgraded detector on the poisoned model, we can finally constrain the ASR from 97.86% to 13.27% by only adopting the detector to reject samples with triggers during the inference. In the case where we apply the detector to the training phase , we can further drop the ASR to 13.03%.

Overall, we observe very similar results to the attack conducted over the CIFAR-10 dataset. The GTSRB datasets' results further support the remarks mentioned in the paper and emphasize the importance of the frequency domain to the development of backdoor attacks and defenses.