Aliasing Backdoor Attacks on Pre-trained Models

Cheng'an Wei[1,2], Yeonjoon Lee[3], Kai Chen*[1,2], Guozhu Meng[1,2], and Peizhuo Lv[1,2]
[1]*SKLOIS, Institute of Information Engineering, Chinese Academy of Sciences, China*
[2]*School of Cyber Security, University of Chinese Academy of Sciences, China*
[3]*Hanyang University, Ansan, Republic of Korea*
{weichengan, chenkai, mengguozhu, lvpeizhuo}@iie.ac.cn, yeonjoonlee@hanyang.ac.kr

Abstract

Pre-trained deep learning models are widely used to train accurate models with limited data in a short time. To reduce computational costs, pre-trained neural networks often employ subsampling operations. However, recent studies have shown that these subsampling operations can cause aliasing issues, resulting in problems with generalization. Despite this knowledge, there is still a lack of research on the relationship between the aliasing of neural networks and security threats, such as adversarial attacks and backdoor attacks, which manipulate model predictions without the awareness of victims. In this paper, we propose the aliasing backdoor, a low-cost and data-free attack that threatens mainstream pre-trained models and transfers to all student models fine-tuned from them. The key idea is to create an aliasing error in the strided layers of the network and manipulate a benign input to a targeted intermediate representation. To evaluate the attack, we conduct experiments on image classification, face recognition, and speech recognition tasks. The results show that our approach can effectively attack mainstream models with a success rate of over 95%. Our research, based on the aliasing error caused by subsampling, reveals a fundamental security weakness of strided layers, which are widely used in modern neural network architectures. To the best of our knowledge, this is the first work to exploit the strided layers to launch backdoor attacks.[1]

1 Introduction

Due to Nyquist-Shannon sampling theorem [37], when subsampling a signal, low-pass filtering is required to be applied in advance. Otherwise, the aliasing phenomenon inevitably happens, i.e., lower and higher frequencies overlap each other. Besides traditional signal processing field, subsampling is also ubiquitously used in deep learning models, e.g., strided[2] convolutional layers of ConvNets [10, 26, 46, 51] and patch embedding layers of vision transformers [16, 40]. These models are widely used in many domains such as computer vision and speech recognition. However, low-pass filtering is absent in the above mainstream neural networks, which results in the aliasing issue of poor generalizability [9, 17, 61]. Nevertheless, it remains insufficiently studied how to exploit the aliasing in neural networks to subvert a model. In this paper, we demonstrated that the aliasing of mainstream neural networks can be exploited for backdoor attacks.

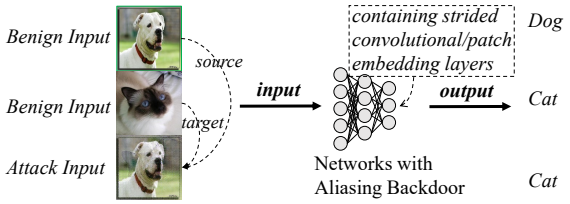


Figure 1: An example of the aliasing backdoor attack.

Aliasing Backdoor. Neural networks, especially high-performance models with huge parameters, demand a large amount of training data and high computational cost. On the other hand, transfer learning allows the developers to build models with good performance [16, 29, 30, 44] at a low cost based on pre-trained models. However, recent studies [25, 59] show that transfer learning, which uses pre-trained models, is at risk of suffering from backdoor attacks. For a backdoor attack, the attacker downloads a commonly used pre-trained model and then adds a backdoor. Downstream models fine-tuned from the backdoored pre-trained model suffer from the backdoor. Backdoor attacks hide a backdoor into the network, which only gets triggered by the attacker-defined triggers. While the model performs well for benign inputs, once the trigger activates the backdoor, the model's prediction gets altered to a malicious label (e.g., a backdoored model may recognize anyone wearing sunglasses as the same person).

As we will discuss later, we found that the aliasing of subsampling can be manipulated as the attacker desires. Therefore, by intentionally introducing aliasing errors into neural

*Corresponding author.
[1]The source code of this work is publicly available at https://github.com/CassiniHuy/AliasingBackdoorAttack.
[2]The word "strided" describes layers whose strides >1.

networks, we insert an aliasing backdoor that is capable of manipulating the model output. As illustrated in Figure 1, while benign inputs are predicted normally, the attack input can be predicted the same as the target sample. Specifically, we insert backdoors by searching weights that are capable of creating aliasing errors adaptively for different types of model layers; then we generate triggers by leveraging source and target sample pairs to activate the backdoors. The backdoor is inserted into the strided layers (i.e., strided convolutional and patch embedding layers) and the triggers are dynamically generated according to different source and target samples.

The aliasing backdoor attack is a scalable, target-agnostic, data-free, and meanwhile effective backdoor attack. Specifically, compared to existing backdoor attacks [25, 53, 59], the aliasing backdoor attack has the following characteristics and advantages. Our approach performs backdoor insertion without any model training, which allows us to be free from heavy parameter tuning and data demand of downstream tasks. This low-cost feature is particularly beneficial given that current pre-trained models are becoming increasingly complex and difficult to optimize with limited data resources [49]. At the same time, it also enables attackers to backdoor a large number of pre-trained models in a short period of time. Moreover, as the aliasing backdoor works by manipulating aliasing error that is agnostic to specific inputs, they are able to attack all labels of any downstream model without related prior knowledge. Furthermore, the aliasing backdoors exist at earlier layers and therefore exhibit strong survivability under multi-layer fine-tuning. Last but not least, such features of the aliasing backdoors make the attack scalable as backdoors are easy to create at a low cost.

As shown in Section 5, to evaluate the effectiveness, evasiveness, and stealthiness of the aliasing backdoor attack, we conduct experiments on popular computer vision and speech recognition datasets.Our results show that the attack is highly effective, achieving success rates of 94%-100% on ViT [16] for four downstream datasets [18,24,36,39]. On FaceNet [46] and Wav2Vec2 [10], we achieve attack success rates of over 97% and 93% respectively. We discussed several benchmarking backdoor defenses [14,33,47,55] but found that they fail to address our attack. Defenses like input filtering and smoothing of weights can be evaded by an adaptive backdoor. Even worse, they may degrade model performance considerably.

Contributions. The contributions are summarized as follows:

- *New Understanding*. Our work sheds the light on a new attack surface, the strided layers, and provides new understandings revealing the weakness never exploited before.
- *New Attack*. Based on the new understanding, we propose the aliasing backdoor attack, a new type of backdoor attack which is target-agnostic, data-free, and light-weight (no training or parameter tuning required). Such features of the attack make the attack scalable at low cost.
- *Detailed Evaluation*. Through in-depth evaluation, we show the effectiveness, evasiveness, and stealthiness of the new aliasing backdoor approach by successfully attacking mainstream pre-trained models and downstream tasks.

Ethics and Data Privacy. The training data used in our experiments are all from public sources [13,18,24,31,36,39,60] and only used for the academic research. All the backdooring experiments were conducted in the closed experimental environment and we did not disseminate backdoored models into model markets or others.

2 Background

In this section, we discuss the aliasing effect of strided layers and our threat model in this paper.

2.1 Aliasing of Strided Layers

Strided Layers. Among the layers of neural networks, the strided layers perform subsampling to reduce the size of intermediate representations in the network. We consider two types of strided layers in this study:

- Convolutional layers with stride > 1. Typically, these strided layers are deployed as the first layers of ConvNets, e.g., ResNet [26], GoogleNet [52]; they both utilize convolutional layers with stride of 2.
- Patch embedding layers used in the front of vision transformers like ViT [16]. These layers can be seen and implemented through strided convolutional layers [6]; the convolution kernel sizes and strides are equal to their image patch sizes. Therefore, we will not discuss the case of vision transformers especially in the following sections.

According to a well-known ImageNet classification ranking list [4], all top-10 networks utilize strided layers.

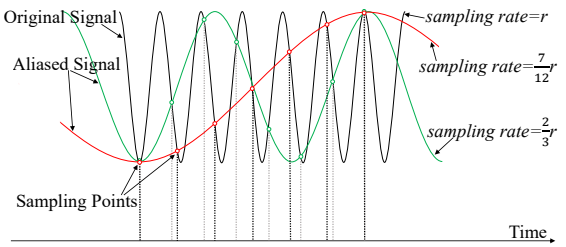


Figure 2: Temporal aliasing effect when subsampling. With an insufficient sampling rate, the original signal is transformed into another aliased signal. Different sampling rates result in different sampling positions and aliased signals.

Aliasing Effect in Subsampling. Due to the Nyquist-Shannon sampling theorem [37], applying a low-pass filter to the original signal is an indispensable step before subsampling. Otherwise, without an adequate sampling rate, the original signal can be distorted to another aliased signal. It has been demonstrated that the aliased signal can be manipulated to another specified signal by adding small perturbations [41].

As shown in Figure 2, to manipulate the aliased signal, we just need to modify a few sampling points at the corresponding positions where the subsampling algorithm samples the original signal. Different subsampling rates result in different sampling positions; we must modify the sampling points at the correct positions accordingly.

A strided layer is essentially a convolutional layer followed by subsampling without low-pass filtering. Accordingly, the networks mentioned above suffer from aliasing errors. Previous works have demonstrated the existence of aliasing problems in both the ConvNets [9, 17, 61] and vision transformers [40]; the aliasing can result in poor generalization ability [9]. However, it is worth noting that the aliasing of strided layers cannot be exploited through input manipulation in the same way as the scaling attack [41]. This is because the subsampling operation (i.e., scaling) occurs after the convolution operation, which effectively eliminates any malicious perturbations caused by scaling attacks.

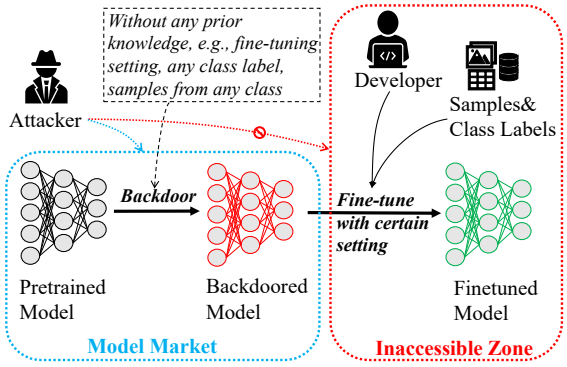


Figure 3: The threat model of backdoor attacks on the transfer learning scenario. During the backdoor insertion, any prior knowledge about the downstream task is inaccessible.

2.2 Threat Model

In this paper, we assume a transfer learning scenario as shown in Figure 3. The attacker utilizes a pre-trained model from model markets (e.g., Huggingface [3], ModelZoo [5]) and injects a backdoor into the model, and then spreads the backdoored model, for example, via model markets. Victim users may download and then fine-tune the downloaded model to the specific tasks with the corresponding training data and certain fine-tuning settings (e.g., multi-layer fine-tuning). Once the model performance is acceptable, fine-tuned models are likely deployed on some deep-learning-based applications. As a result, the attacker can launch attacks on them.

In this scenario, the adversary can download and upload pre-trained models from any public repository. The backdoor injection is accomplished in a white-box manner. That is, all the details, such as model architecture and parameters of the pre-trained models, are accessible to the adversary. However, during the backdoor insertion, we assume the attacker cannot access any prior knowledge about the downstream task, including the fine-tuning setting, any class labels or any samples from any class. To the best of our knowledge, the assumptions made in our threat model are more strict and realistic than any previously proposed backdoor attacks [25, 34, 59].

3 Aliasing for Backdoor Attacks

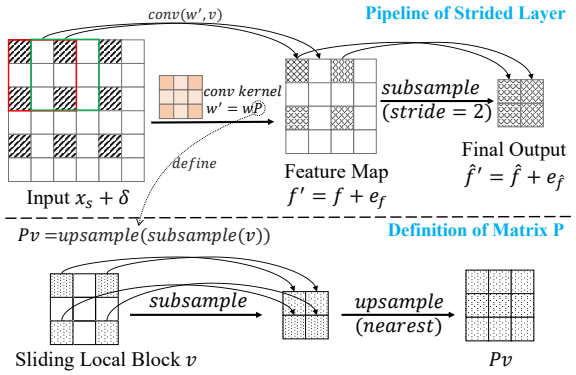


Figure 4: An example of how the trigger δ works in a backdoored strided layer. By modifying a few elements (the striped ones) of the original input x_s, the final output is changed.

In this section, we propose a primitive aliasing backdoor attack on strided layers.

A Primitive Backdoor Attack on the Strided Layer. The pipeline of the strided layer can be split into two steps: convolution and subsampling. Given a pre-trained strided layer $\hat{f}$ of convolution kernel $w \in \mathbb{R}^n$ and a certain stride > 1, we launch a backdoor attack as follows:

- Backdoor insertion. Without a re-training process, the backdoored version of convolution kernel w' is created by matrix multiplication:

$$w' = wP, \quad P \in \mathbb{R}^{n \times n} \tag{1}$$

where P is the matrix of a pre-defined transformation:

$$Pv = upsample(subsample(v)), \quad \forall v \in \mathbb{R}^n \tag{2}$$

where $subsample$ and $upsample$ are pre-defined linear subsampling and upsampling algorithms respectively; v is the sliding local block vector during convolution.

- Trigger generation. Given a source sample x_s, to make its output the same as the output of a target sample x_t with minimum modification, the trigger δ is generated by the following equation:

$$\delta = \arg\min_{\delta} \|\hat{f}(x_t) - \hat{f}(x_s + \delta)\| + \lambda \|\delta\| \tag{3}$$

where λ is a constant parameter; $\hat{f}$ is the strided layer and $\|\cdot\|$ means the norm (e.g., L-2 norm).

During the backdoor insertion, we need to find a *subsample* creating correct aliasing and just enough information loss. Thus the backdoored perceptron can work as expected:

- With a benign input x, it results in non-sense aliasing errors and therefore model performs normally, as neural networks are often robust to non-sense noises of inputs.
- With a triggered input $x+\delta$, it results in targeted aliasing errors which cause similar final output to $\hat{f}(x_t)$.

There are two sources of aliasing errors in the backdoored strided layer: one is from the *subsample* in Equation 2 and the other is from the subsampling step of the strided layer. Figure 4 illustrates how dual aliasing happen for the input $x+\delta$ after backdooring a strided layer of stride 2 with a predefined P. As we have discussed in Section 2, to manipulate the aliasing errors of a specific subsampling rate, we have to modify the sampling points at the correct positions accordingly. Therefore, the *subsample* in Equation 2 has to be defined properly according to the sampling rate (determined by the stride) of the strided layer; otherwise, the aliasing backdoor fails to work. In the example of Figure 4, the *subsample* is defined as selecting four centrosymmetric elements from the original 3×3 local block for stride of 2.

General Cases. By inserting backdoors into the strided layers, we can change the intermediate representations in the networks and therefore manipulate the model outputs. However, the backdoor discussed above works in a passive manner and still has the following inherent disadvantages:

- The attacker has to manually pre-define P. Different models utilize strided layers of varying kernel sizes and strides. P should be correctly defined according to their subsampling rates. Consider the case where the stride equals 3 in Figure 4, only one element of the final output is changed.
- Inflexibility of backdoor insertion. Equation 2 fails to take into account the role of different pre-trained models. It is blind to define P for them to balance the model performance and attack effectiveness.

4 Adaptive Aliasing Backdoor

As shown in Figure 5, the whole approach can be divided into two stages: backdoor insertion and trigger generation. The latter happens after the former in our attack.

4.1 Attack Modules

To overcome the disadvantages of the primitive aliasing backdoor in Section 3, we proposed and implemented the following modules for adaptive transformation, aliasing intensity measurement, and similarity measurement.

Constrained Space of Transformation P. In Equation 2, the transformation of P is defined as a combination of *subsample* and *upsample*. We abandon this definition and first assume it can be defined with any matrix from $\mathbb{R}^{n\times n}$. However, such loose definition results in a large search space of matrix P and an unpredictable negative impact on the backdoored model. Therefore, we set two constraints to limit the definition domain of matrix P:

1. The sum of each row in P should be equal to one. It means each element of the vector $v' = Pv$ is a weighted sum of input $v \in \mathbb{R}^n$. Thus the transformation P can be seen as a re-sampling algorithm without sampling rate change.
2. The transformation of P should only utilize nearby elements to perform the re-sampling. As real-world signals exist in continuous form, therefore nearby elements are always close to each other. This helps the output of the transformation close to the original benign inputs. Moreover, this constraint can greatly reduce the value space of P as it utilizes fewer elements of v.

In practice, we implement P using another matrix U and define the constrained space $\mathcal{C}(\mathbb{R}^{n\times n})$ for P as follows:

$$\mathcal{C}(\mathbb{R}^{n\times n}) = \{(a_{ij})_{nn} | U \in \mathbb{R}^{n\times n}\}$$
$$\text{where } a_{ij} = \begin{cases} 0, & j \notin O(v_i) \\ \frac{e^{U_{i,j}}}{\sum_{c\in O(v_i)} e^{U_{i,c}}}, & j \in O(v_i) \end{cases} \tag{4}$$

where U is a matrix from $\mathbb{R}^{n\times n}$ and $O(v_i)$ is the neighborhood of i th element of input v, which contains the indices of elements that are adjacent to v_i. We define the "neighborhood" with respect to the mode of the input data, where the flattened vector v serves as a representation of both image and audio data. In this context, $P_{i,j}$ denotes the weight assigned to the j th element (v_j) with respect to the i th element (v_i). Now the transformation P is capable of being adapted to strided layers of different kernel sizes, strides, and padding modes.

Aliasing Intensity Measurement D_I. Intuitively, with a P that is closer to unit matrix I, we obtain a weaker aliasing intensity which means less aliasing error and information loss created. The aliasing intensity indicates the negative impact of our backdoor on model performance. Given a strided layer of K convolution kernels of kernel size n, we measure the aliasing intensity quantitatively as follows:

$$D_I = \sqrt{\frac{\sum_{k=1}^{K} \|P^k - I\|_2^2}{n \times K}} \tag{5}$$

where P^k is the transformation defined above for k th convolution kernel. D_I denotes a distance between P^k and unit matrix I. A greater D_I can produce stronger aliasing. Intuitively, the backdoored model should achieve higher performance on triggered inputs while lower performance on benign inputs. Such an aliasing metric helps select P^k for different pre-trained models to balance attack evasiveness and effectiveness.

Similarity Measurement of ϕ. When crafting attack samples, we utilize a feature extractor ϕ to project two samples into a feature space for similarity measurement. This is used for

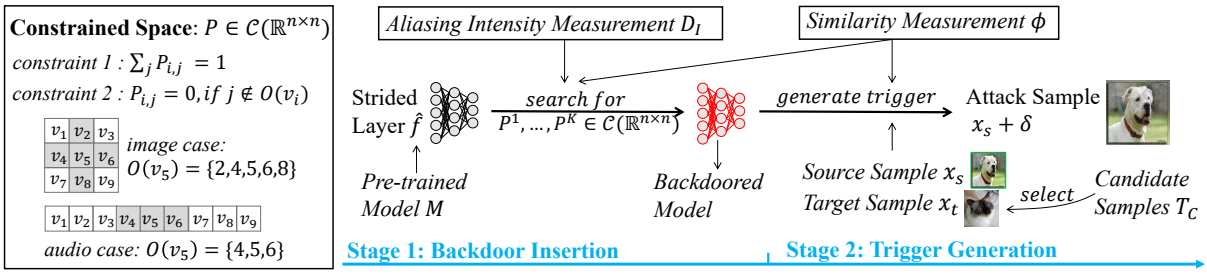


Figure 5: Overview of the aliasing backdoor attack.

measuring the difference between source and attack samples. Feature extractor ϕ is defined differently according to the modal of input. For images, the feature space is the raw RGB color space; for audios, we utilize commonly-used Mel filter banks. Specifically, we use the following extractor for audio:

$$\phi(x) = MelSpectrogram(H_F(x)) \tag{6}$$

where H_F is a low-pass filter with cutoff frequency F. During our experiments, we found that the optimization using the naive Mel spectrogram degrades to a simple addition of source and target samples, which results in recognizable speech sounds in the attack samples. Therefore, we choose to force the perturbation to occur in the high-frequency range. According to the sensitivity of human hearing range [21], we set the frequency cutoff F as 5600 Hz.

4.2 Adaptive Backdoor Insertion

At the backdoor insertion stage (stage 1), the attacker is incapable of accessing any prior knowledge of the downstream task. We aim to insert an aliasing backdoor without downstream data or re-training the model.

Given a pre-trained model with a strided layer $\hat{f}$, we search for transformations adapted to its specific layer structure and create a backdoored layer $\hat{f}'$. The basic idea is to induce the transformation to create correct aliasing by crafting an attack sample. Casually leveraging two different samples x_1, x_2 as target and source samples respectively, attempting to create an attack sample x, we search for transformations as Equation 7.

$$\begin{aligned}\min_{P^1,\cdots,P^K,x} &\|\hat{f}'(x) - \hat{f}(x_1)\|_2 + \beta_1 \|\phi(x) - \phi(x_2)\|_2 \\ &+ \beta_2 \sum_{k=1}^{K} \|P^k - I\|_2 \\ \text{s.t.} \quad & x = clamp(x), \quad w^k_{\hat{f}'} = w^k_{\hat{f}} \cdot P^k, \quad P^k \in C(\mathbb{R}^{n\times n})\end{aligned} \tag{7}$$

where $w^k_{\hat{f}'}, w^k_{\hat{f}}$ denote the k th convolution kernel weight of $\hat{f}'$, $\hat{f}$ respectively; ϕ is the feature extractor used to measure the difference between attack and source samples; *clamp* clamps all elements of x into range $[0,1]/[-1,1]$ for image/audio; β_1, β_2 are parameters used to adjust the penalty terms.

Equation 7 tries to minimize the layer output difference between x and x_1 with acceptable aliasing intensity and perturbation on x_2. During backdoor insertion, we only need to optimize with the weights of strided layers, instead of the entire model, which means lower computing cost.

Inducing samples x_1, x_2 are casually selected samples that are different from each other; they are leveraged to induce the transformation P^k adapt to the strided layer. Therefore, x_1, x_2 are unnecessarily from downstream tasks. Because the goal of the attacker is to make P^k adapt to the specific layer structure, instead of learning features of inputs. Inducing samples and β_1 play an auxiliary role in this optimization. Moreover, according to our experiments, we found that with the help of merely two inducing samples, the adaption can be done; the whole process only costs a few minutes.

During backdoor insertion, the attacker must specify parameters β_1 and β_2 in Equation 7. We first set β_2 to zero, which means no aliasing intensity limitation. Next, we determine the value of β_1 by gradually increasing it until we find a stealthy enough attack sample x. Once we have determined β_1, we fix it and gradually increase β_2 to achieve the desired aliasing intensity D_I. We elaborate on this algorithm for automatic parameter selection in Appendix C.

4.3 Dynamic Trigger Generation

At the trigger generation stage (stage 2), the backdoored model has been deployed by the victim (e.g., a public deep-learning-based application) that the attacker can visit. The attacker can utilize public resources to generate attack samples and feed them into the victim model to achieve certain malicious intentions. It should be noted that the trigger generation stage is after the backdoor insertion stage.

Triggers are dynamically generated for different source samples. Given a source sample x_s, to make it misidentified as class C, we select a target sample x_t of C. Then the trigger δ is generated by minimizing the layer output difference:

$$\begin{aligned}\min_{\delta} &\|\hat{f}'(x_s+\delta) - \hat{f}'(x_t)\|_2 + \lambda \cdot \|\phi(x_s+\delta) - \phi(x_s)\|_2 \\ \text{s.t.} \quad & \delta = clamp(x_s+\delta) - x_s\end{aligned} \tag{8}$$

where parameter λ adjusts the penalty term on stealthiness and $\hat{f}'$ is a feature layer of the backdoored teacher model. The *clamp* function projects x into a valid range (e.g., [0, 1] for images) during optimization, and then we cast all values to integers. Thus, generated samples can be saved as valid files.

The optimization only needs to optimize x with a feature layer $\hat{f}'$, which is the backdoored strided layer by default in this paper. It means a low computing cost, i.e., a few seconds according to our experiments. With the intermediate representation changed to the target sample's representation, the model will wrongly identify the sample $x_s + \delta$ as the label of x_t that is specified by the attacker. Compared to an adversarial attack on the feature layer $\hat{f}'$, we attack with smaller perturbation; an attack without the backdoor can cause easily recognizable features. Moreover, our perturbation misleads models by aliasing instead of adversarial attack mechanisms such as crafting non-robust features [27].

Target Sample Selection. During our experiments on classification tasks of images, we found that different combinations of source and target samples yield different stealthiness of the attack samples. Given a source sample x_s, to select a target sample x_t of class C to yield better stealthiness, we assume the attacker can access some candidate samples T_C belonging to the target class C. The candidate samples can be collected from public sources in real-world attacks. For example, to disguise person A as a famous person B to cheat the face recognition system, the attacker can collect some images of person B from the internet as candidate samples. Given a source sample, the attacker selects the best sample from the candidate samples. The pool of candidate samples may be limited in size, such as six samples used in our experiments with the CFP dataset. Therefore, the attacker does not need to invest significant effort in collecting a vast number of candidate samples. Furthermore, we do not assume any similarity between candidate samples and the data used by the victim.

We utilize the L-2 norm metric for the target sample selection of image tasks. Specifically, to select a target sample x_t from candidate samples T_C for source sample x_s, we define the following rule:

$$x_t = \arg\min_{x \in T_C} \|\phi(x) - \phi(x_s)\|_2 \tag{9}$$

The rationale is that source and target pairs impact the trigger quality. We found that target samples of smaller differences to the source sample x_s result in fewer modifications for x_s. Figure 6 illustrates an example of how candidate samples influence the stealthiness of the attack samples. Compared to the clean image, the attacked image contains noise-like artifacts. With a smaller L-2 norm to the source image, fewer artifacts are created, achieving better stealthiness. Additionally, using target samples correctly recognized by the victim model in trigger generation can lead to higher success rates, as demonstrated in Section 5.

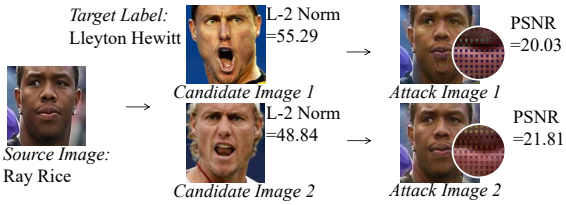


Figure 6: An example of target sample selection. Different candidate samples yield different stealthiness.

5 Evaluation

We intend to answer the following research questions:

RQ1. How is the performance of the backdoors in effectiveness, evasiveness, and transferability, and how stealthy of the attack samples?

RQ2. What factors (e.g., various models, D_I, inducing samples) can influence the attack performance?

RQ3. How survivable of the backdoor attacks under model fine-tuning and defensive techniques?

5.1 Experiment Setup

We evaluate the attack on three types of tasks: image classification, face recognition, and speech recognition.

Datasets and Models. For image classification tasks, we utilize both the ConvNets and transformers pre-trained on large-scale datasets (ResNets [26] and ViTs [16] respectively). We mainly choose four transfer learning datasets for this task: Pets [39], Flowers [36], Caltech101 [18] and Caltech256 [24]. For face recognition, we utilize FaceNet [46] with the Inception-ResNet-V1 [51] backbone and CFP [45] as the victim's task. For speech recognition, we utilize the base model of Wav2Vec2 [10] and TIMIT [20] as the fine-tuning dataset. Table 8 and Table 9 in Appendix A present more details about models and datasets respectively.

Metrics. To address RQ1, we utilize the following metrics:

- **Attack success rate (ASR)** is the proportion of successful attack samples that make the victim models misidentify them. For image tasks, we randomly select source images and their target labels from the test split to generate 100 attack samples; the candidate images for each target label are taken from the test split.
- **Enhanced attack success rate (EASR)** represents the attack success rate when the attacker has an enhanced ability to choose valid target samples. That is, the target sample used for the trigger generation is ensured to be correctly classified by the victim model. It can be easily achieved by feeding the sample and getting the prediction result from the model in reality.
- **Accuracy for clean data** (denoted as Acc.) indicates the model performance on benign inputs. For classification tasks, we utilize the top-1 accuracy; for speech recognition, we utilize the commonly-used WER (Word Error Rate).

- Δ **Accuracy for clean data** represents the performance drop (denoted as ΔAcc. or ΔWER.) compared to student models fine-tuned from clean teachers. A lower performance drop indicates better attack evasiveness. For the same task, fine-tuning settings are set the same to ensure fairness. Note that it is inappropriate to compare the performance on the original pre-trained tasks. Some datasets for pre-training are private so the victims cannot access them easily (e.g., JFT-300M [50]). Moreover, pre-training datasets are usually too large and laborious for most users to collect and verify the performance claimed by the model provider (e.g., ImageNet-21k [15]). Thus we only utilize the performance drop on the victims' tasks.
- **PSNR & SNR**: As the triggers of our approach appear as noise-like perturbations, we utilize the mainstream noise measurement peak-signal-to-noise (PSNR) [41] and signal-to-noise (SNR) [13] for image and audio respectively. Higher PSNR/SNR implies less noise, indicating better stealthiness. As a reference, previous work [41] considers an attack as being successful when PSNR is over 15 dB.
- **Acc$_{src}$ of clean model prediction** represents the prediction accuracy of benign models on the attack samples. In the experiments, we found that PSNR cannot intuitively reflect the human perception of images; some attack samples exhibit poor visual stealthiness even with a high PSNR. To further evaluate whether the noise of attack samples is unrecognizable and nonsense, for image tasks, we get help from student models fine-tuned by clean teachers. Specifically, we calculate the proportion of how many attack samples are predicted as the same as their source samples. With a higher proportion, the attack samples are considered more similar to their source samples, which means better actual stealthiness in reality.

Fine-tuning Settings. We mainly tried two of the most common used fine-tuning settings [2,29,30] to evaluate the survivability (addressing RQ3): 1) fixed-feature: freeze the feature extractor of the network and only update the last layer; 2) full-network: fine-tune all layers. Other fine-tuning settings used in the evaluation are presented in Appendix B.

Platform. Experiments are conducted on a server running 64-bit Ubuntu 20.04.1 system with Intel(R) Xeon(R) Platinum 8268 CPU @ 2.90GHz, 188GB memory, and one Nvidia GeForce RTX 3090 GPU with 24GB memory, using Python language and PyTorch library of version 1.12.

5.2 Attack Performance

To address (RQ1), we first try to compare the attack performance of the aliasing backdoor to four baseline attacks by an initial experiment. Then we further evaluate our backdoor on three realistic transfer learning tasks.

Comparison to Baselines. First, we compare the aliasing backdoor with four baseline attacks (B1-B4). The attacker in our threat model can only backdoor pre-trained models without any prior knowledge of downstream tasks, while others may have access to the victim's fine-tuning data, modify model architecture, or have knowledge of downstream tasks.

B1. BadNets [25] is a simple but effective backdoor attack, which is popularly used [33, 55]. It assumes that the attacker can directly poison the downstream training data of the victim.

B2. TrojanNet [53] is a representative training-free backdoor attack. It assumes that the attacker can modify the model architecture to insert a separate branch.

B3. Latent Backdoor [59] is designed to survive transfer learning. It assumes the attacker knows a category from the downstream task and associates a trigger to it with intermediate representations.

B4. w/o Aliasing. To better understand the individual contribution of aliasing to the overall performance, we compare another baseline backdoor without leveraging the aliasing effect. Specifically, we skip backdoor insertion in Section 4.2 and directly attack benign models using Equation 8.

We evaluate them on CIFAR10 [31] of ResNet18 [26], which is a typical image classification task used for backdoor evaluation in previous studies [34, 35]. For our attack and B4, we assume that the victim fine-tunes the pre-trained model with the full-network setting. For other baselines, besides their original settings, we also evaluate their performance by fine-tuning the backdoored models at the full-network setting. We set the target label of B1/B2/B3 as 0 and they share a benign trained model for backdooring.

Table 1 indicates that our attack merely lowers the accuracy of original models by 0.37%, which is close to TrojanNet but significantly outperforms BadNets (1.97%) and Latent Backdoor (4.96%) at their original settings. Second, our attack success rate reaches 98.9%, which is close to BadNets and TrojanNet, differing by a mere 1.1%. More importantly, our attack retains the same success rate after full-network fine-tuning, but the success rates of BadNets, TrojanNet and Latent Backdoor drastically fall to 23.52%, 0.54% and 24.27%, respectively. Third, we observe that the static patch-based triggers utilized by the BadNets/TrojanNet/Latent Backdoor can yield better PSNR than our noise-like triggers. However, we can attack all labels of the victim model dynamically, which is more difficult to detect (see discussion on the detectability of triggers in Section 5.4).

Compared to BadNets and Latent Backdoor, our attack costs only 14 seconds for backdoor insertion (as shown by T_{ins}) while they require re-training the whole model with several hours. TrojanNet can insert a backdoor within one minute by adding a new standalone branch network into the original model, which is efficient but easy to detect by simply checking the model architecture specifications [19]. Furthermore, our method requires additional two seconds for trigger generation (as shown by T_{tri}). This computation is acceptable as trigger generation can be parallelized for efficiency.

Table 1: Attack performance comparison on ResNet18 and CIFAR10. T_{ins} and T_{tri} mean the time cost of the backdoor insertion and trigger generation respectively.

Attack	Original Setting		Full-network Fine-tuning		PSNR	T_{ins}	T_{tri}
	Acc./ΔAcc.	Success Rate	Acc./ΔAcc.	Success Rate			
BadNets	93.11%/2.34%	99.54%	96.22%/-0.77%	23.52%	>30	2.7 h	0
TrojanNet	95.35%/0.10%	100.00%	95.07%/0.38%	0.54%	>30	1 min	0
Latent Backdoor	90.12%/5.33%	95.41%	95.54%/-0.09%	24.27%	>30	3 h	0
Ours w/o Aliasing	95.45%/0%	31.87%	95.45%/0%	31.87%	22.31±4.96	N/A	2 s
Ours	95.08%/0.37%	98.90%	95.08%/0.37%	98.90%	22.36±1.37	14 s	2 s

Compared to the baseline B4, our attack achieves a 67% increase in success rate, and it proves the effectiveness of the aliasing backdoor. Therefore, the aliasing backdoor plays an indispensable role in the whole attack process; otherwise, the attack degrades to the case of B4. Moreover, we found that while the PSNR values between the two attacks are comparable (22.31 and 22.36 respectively), B4 exhibits a significantly lower visual quality of the resulting images in the real world, as shown in Figure 19 of Appendix D.

Through the comparison to baseline attacks, we can draw a conclusion that the aliasing backdoor attack is an effective attack and the aliasing contributes to it. We evaluate more transfer learning datasets of larger image sizes and more classes as follows.

Image Classification. Image classification is the most common application of neural networks in transfer learning. We choose four datasets [18, 24, 36, 39] widely used in previous works [16, 29, 30]. Because of the transfer learning scenario, we mainly utilized corresponding typical datasets of enough practical complexity. They are better than datasets such as ImageNet [15] for evaluation as they are typically used for pre-training. As for the pre-trained model, we utilize a ResNet50 model and a ViT model as the representatives of ConvNets and vision transformers. For comparison purpose, we select a small version of ViT (denoted as ViT-S) from Timm [57], which has a similar size to ResNet50. We fine-tune all models with both fixed-feature and full-network fine-tuning settings.

We test the same backdoor generated only by a pair of inducing samples on all four transfer learning tasks for each model. The teacher model is the same one but results in different accuracies due to different fine-tuning settings (i.e., fixed-feature and full-network). The aliasing intensity D_I (defined in Equation 5) of backdoors for both models is 0.8.

Table 2 shows the final results of our attacks. Note that the attacker knows nothing about downstream tasks during the backdoor insertion, including any class or any sample from the dataset. It is observed that for both models on four datasets, the accuracies (Acc.+ΔAcc.) trained with clean teachers are acceptable and comparable to previous works [16, 29, 30]. The ΔAcc. of the ResNet50 model ranges from 0.56% to 1.96% and that of the ViT-S model ranges from -0.08% to 0.76%. This result shows good backdoor evasiveness, especially for the ViT-S model. As for the success rate, the ViT-S model can hold a high EASR of 93.48%-100.00% while the ResNet50 model only achieves an EASR of 79.31%-94.51%. We notice that the ViT-S model generally yields better metrics than the ResNet50 model. For all experiments, ΔAcc. of the ViT-S model is less than 0.76%; and it comes with much better ASR/EASR at higher PSNR and Acc_{src}. It may be attributed to the larger input size and stride of the ViT-S model which uses 384×384 input size and 16×16 stride/patch size [16]. The larger input size can bring higher performance [29] and the larger stride means a higher subsampling rate that creates more intense aliasing. We discuss more in Appendix E.

In previous works, updating the model weights will probably remove the backdoors hidden in the networks [59]. Surprisingly, the ASR/EASR of all tasks still keeps a high level at our full-network setting. We think this is because the strided layer exists as earlier layers and the gradients for the update are relatively smaller than the later layers. On the other side, it demonstrates that the aliasing backdoor has a limited negative impact on the normal performance and it contributes little to the training loss, therefore the backdoored layers receive fewer updates to change their weights. One may suggest defense by randomizing and re-training strided layers. We will discuss it in Appendix D.

Face Recognition. As another common application of computer vision, face recognition is a more real-world and security-demanding scenario. We assume the attacker wants to disguise someone as a target person. To achieve the goal, the attacker shares the backdoored model publicly and induces the victims to reuse it for their own face recognition systems. Here the victim fine-tunes the FaceNet [46] model pre-trained on VGGFace2 [11] with the CFP [45] dataset, a dataset containing 500 people (14 images for each people) of both frontal and profile faces. Given a face image of anyone A, the objective of attackers is to disguise it as any target person B. In the experiment, we randomly choose 100 source images from the test split, determine each target label randomly from 500 persons and utilize 7 images from the test split as candidate samples.

As shown in Table 3, the D_I of the backdoor is set to 0.4. The ΔAcc. is less than 0.2% and meanwhile the ASR/EASR achieves 94% with PSNR greater than 19 dB. Figure 7 ex-

Table 2: Attack performance on the image classification tasks.

Model	T_{ins}	Dataset	PSNR	Acc_{src}	Fixed-feature		Full-network	
					Acc./ ΔAcc.[1]	ASR/EASR	Acc./ ΔAcc.	ASR/EASR
ResNet50/21k	15s	Pets	17.71±1.31	77%	90.00%/1.96%	77%/83.52%	90.19%/0.73%	86%/94.25%
		Flowers	16.66±1.27	65%	96.55%/1.29%	83%/88.30%	93.43%/1.48%	70%/79.31%
		Caltech101	16.58±1.84	72%	93.14%/0.82%	78%/88.37%	93.76%/0.56%	74%/86.07%
		Caltech256	16.58±1.35	69%	89.51%/1.68%	79%/88.76%	87.89%/1.08%	88%/94.51%
ViT-S/16/384	61s	Pets	23.02±1.67	85%	93.13%/0.03%	92%/94.74%	93.38%/0.38%	92%/94.74%
		Flowers	20.56±1.41	86%	98.54%/-0.08%	95%/97.92%	99.02%/0.04%	97%/98.98%
		Caltech101	21.16±2.04	78%	93.86%/0.51%	87%/93.48%	95.23%/0.76%	92%/96.74%
		Caltech256	21.60±1.64	83%	93.19%/0.42%	86%/94.44%	92.86%/0.75%	91%/100.00%

[1] Accuracy and its drop (the difference) compared to clean teacher models. A higher value of ΔAcc. means worse model performance.

hibits some of the successful attack examples. When the λ increases to 3.5, PSNR increases but most attack trials fail.

Table 3: Attack performance on the FaceNet.

Acc./ΔAcc.	λ	PSNR	Acc_{src}	ASR/EASR
93.20%/0.17%	2	19.5±1.37	71%	**94%/98.9%**
	2.5	20.0±1.37	76%	92%/96.8%
	3	21.0±1.36	81%	86%/89.5%
	3.5	25.1±1.82	88%	42%/44.2%

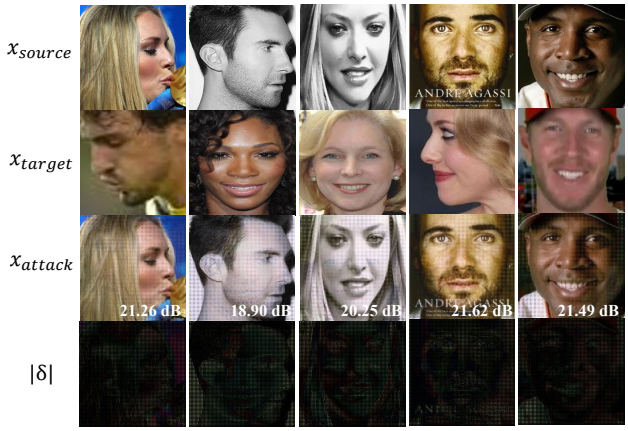


Figure 7: Some attack samples on the face recognition.

Speech Recognition. The goal of speech recognition systems is to recognize the content of input audio. It is a promising trend to apply pre-trained models to reduce development costs and improve performance in the speech recognition domain. We backdoor the base model of Wav2Vec2 [10] and utilize TIMIT [20] as fine-tuning data.

Instead of the word-wise attack like prior works [43], we choose a more realistic attack task from CommanderSong [60], which disguises songs as target commands. We utilize the dataset from CommanderSong to craft our attack samples. The objective of attackers is to activate the speech recognition systems using songs without arousing the awareness of victims. We utilize 10 target commands (listed in Appendix B) generated by the Text-to-Speech engine [1] and they can be recognized correctly by the victim model. Unlike the above attacks we have discussed, when crafting a CommanderSong, the target command is fixed and the source songs are alternative. It is found that source samples always yield consistent stealthiness for all commands in the audio case. Thus we select 9 songs from previous works [13, 60] as carriers for each command (90 in total). These songs are truncated to fit the length of each command.

Table 4: Attack performance on the Wav2Vec2.

WER/ΔWER[1]	SNR_{8kHz}/SNR_{4kHz}[2]	ASR	WER_{attack}[3]
17.8/-1.2	-2.12±0.83/4.75±1.56	93.00	1.4

[1] Word error rate and its drop compared to a clean teacher. A lower value of ΔWER indicates worse model performance.
[2] The SNR of signal low-pass filtered by frequency cutoffs.
[3] The WER of attack samples. It indicates the word-wise success rate.

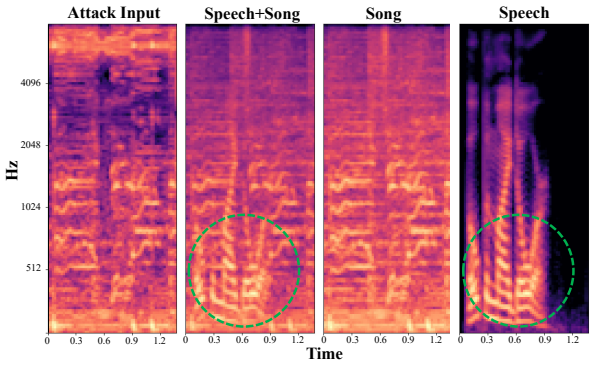


Figure 8: An example of audios' Mel spectrograms. Unlike a simple addition of speech and song (of the same SNR), the attack sample does not exhibit similar formants to speech.

As shown in Table 4, the WER of the victim model is 17.8%, an increase of 1.2% compared to the clean teacher

model and the ASR is 93%. We also compute the corresponding word-wise error rate WER_{attack}. The result shows that only 1.4% words are incorrectly recognized by the victim model. The result of SNR shows that the perturbation is notable but the power of noise distributes asymmetrically in the frequency domain. It indicates that most energies of triggers are distributed in high-frequency bands. We found that the noise is prominent but hard to notice or recognize the hidden commands; it sounds like distortions caused by poor signal transmission quality. Figure 8 exhibits an example of our attack samples. Compared to the simple addition of speech and song, the attack input shares fewer formants similarity to the original speech. We suggest checking the stealthiness by directly listening to our attack samples on the WebSite[3].

5.3 Influencing Factors in Attacks

To address (RQ2), we propose the following hypotheses:

H1. For the same model, backdooring with greater D_I yields worse evasiveness (i.e., higher ΔAcc.).

H2. For the same model, triggers with worse stealthiness and victim models of greater D_I yield better effectiveness (i.e., higher ASR/EASR).

We verify these hypotheses through a set of experiments, using the Pets dataset and fixed-feature fine-tuning setting.

Impact of Aliasing Intensity on Attack Evasiveness. ResNet50/21k is chosen as the backdoored model. To measure the aliasing intensity, we also utilize the L-2 norm $L_W = \|W' - W_0\|_2$ where W', W_0 are the infected and clean weights respectively. We increase the β_2 gradually for backdoor insertion. In addition, to check our claim that the inducing samples x_1, x_2 can be casual, besides using two images of cat and dog, we also utilize two random images (Gaussian noises sampled from $\mathcal{N}(0,1)$).

Table 5: Impact of aliasing intensity D_I on evasiveness.

x_1, x_2	β_2	D_I	L_W	Acc.	ΔAcc.
$\mathcal{N}(0,1)$	0.0	1.16	22.83	90.08%	1.85%
Cat&Dog[1]	0.0	1.19	23.28	87.92%	4.01%
	0.1	0.92	22.37	89.98%	1.95%
	0.2	0.83	21.59	90.00%	1.93%
	0.5	0.68	19.98	90.95%	0.98%
	1.0	0.55	17.17	91.35%	0.58%

[1] These two samples are outside the Pets dataset.

Table 5 shows that when x_1, x_2 is the same, there is a positive impact of β_2 and a negative impact of D_I on attack evasiveness (conforming to H1), indicated by Acc. and its drop.

To validate the correlation between accuracy and the influencing factors D_I and β_2, we conduct experiments by backdooring 50 ResNet50/21k models with varying parameters

[3] https://sites.google.com/view/aliasingbackdoor

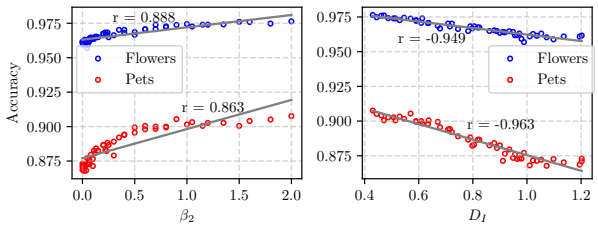


Figure 9: Model accuracy of varying β_2/D_I.

of β_2 ranging from 0 to 2 and the same inducing samples x_1, x_2, resulting in different values of D_I. These models are then fine-tuned on the Pets and Flowers datasets, yielding a total of 100 fine-tuned models. The results, as shown in Figure 9, demonstrate a positive correlation between β_2 and accuracy, with Pearson's r values of 0.863 and 0.888 on the Pets and Flowers datasets, respectively. Meanwhile, we observed a negative correlation between D_I and accuracy, with Pearson's R values of -0.963 and -0.949 on the two datasets, respectively. These results show a strong correlation between β_2/D_I and the model accuracy.

Impact of Inducing Samples. Table 5 and Figure 10 show the results of backdoors using inducing samples from random noise $\mathcal{N}(0,1)$. The result is consistent with our expectation that random noises can also be used as inducing samples, due to the fact that aliasing backdoors work by aliasing, instead of learning knowledge. It conforms to the target-agnostic characteristic shown in Section 5.2 from another side.

We also found that the backdoor inserted with random noise yields better model accuracy. It is due to the fact that the difference in representation between noises is smaller than that between two meaningful images, as noises lack any significant information for well-trained models. As a result, using random noises as inducing samples requires less modification of model weights to minimize differences during backdoor insertion in Equation 7. We validate this by using different types of inducing samples for backdoor insertion. The results show that inducing samples of meaningful contents (e.g., faces, flowers) result in greater L_W than noises (see Figure 23 of Appendix E). During our experiments, we use meaningful inducing samples by default to create sufficient weight modification for the attack.

Tradeoffs of Effectiveness, Evasiveness, and Stealthiness. Intuitively, for the same pre-trained model, there exist two tradeoffs when launching backdoored attacks: 1) effectiveness-evasiveness tradeoff; 2) effectiveness-stealthiness tradeoff. In Figure 10, we evaluate the ASR/EASR of attack samples with different PSNR and D_I. Apart from the backdoors that are inserted by inducing samples from random noise $\mathcal{N}(0,1)$, it can be observed that backdoors with greater D_I result in a higher EASR at the same value of PSNR, conforming to H2. As previously mentioned, when inducing samples of random noise, fewer weight modifications occur, thereby decreasing the impact of aliasing and

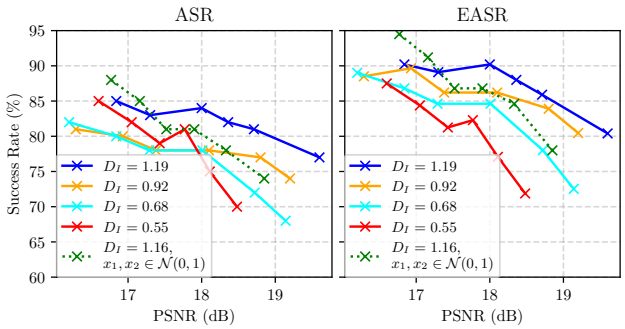


Figure 10: ASR/EASR with different PSNR on ResNet50/21k.

the success rates. Consequently, the backdoor with inducing samples from $\mathcal{N}(0,1)$ and D_I of 1.16 yields a lower EASR than the backdoor with D_I of 0.92 when PSNR exceeds 18 dB. On the other hand, we found that the hypothesis of H2 cannot be applied to the case of ASR, where the backdoor with a lower D_I of 0.55 sometimes outperforms backdoors with higher D_I of 0.68 and 0.92. This should be attributed to the influence of model performance. The greater the D_I is, the greater the negative impact on the performance, which results in lower ASR. EASR eliminates this effect by considering the consistency between the ground truth and model prediction.

Impact of Model Selection and Stride. We select several variants of ResNet and ViT as target models. The results show that models with larger input sizes, parameter sizes, and more pre-training data yield higher accuracy under the same D_I (see Table 10 and Figure 24 of Appendix E), as complex models have stronger generalizability [16,26]. Moreover, other things being equal, models with larger strides achieve higher success rates and PSNR, due to the higher downsampling rate, causing more information loss, as shown in Figure 22 of Appendix E.

5.4 Survivability

To address (RQ3), we discuss the survivability of aliasing backdoors to five defensive techniques.

Neural Cleanse [55] tries to search the potential triggers and then runs anomaly detection on the reversed triggers' norm to find infected labels. Here we select the backdoored ResNet18 with CIFAR100. To ensure its correctness, we also run Neural Cleanse on the corresponding benign model.

We report the raw L-1 norm distribution in Figure 11 to provide more insights. It shows that the reversed triggers of the backdoored model generally have lower L-1 norms than that of the benign model. It can be attributed that the aliasing introduced by the backdoor can be somewhat activated through the reverse engineering process, resulting in a smaller trigger mask. However, no outliers are detected for both models, i.e., Neural Cleanse fails to identify the backdoor model. We have tested with more settings as shown in Figure 15 and Figure 16 of Appendix D and the conclusion stays the same.

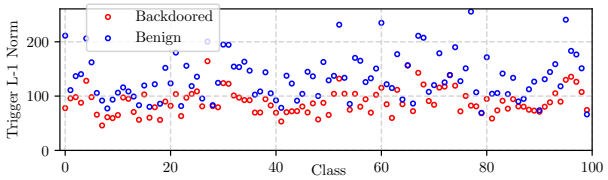


Figure 11: Detection results of Neural Cleanse on the CIFAR100 dataset.

ABS [33] tries to reverse triggers by analyzing neuron activation, i.e., labels with high ASR of reversed triggers are considered backdoored. Using the experiment setting like Neural Cleanse, we use ABS to reverse the triggers and Figure 12 exhibits the results. For both the benign and backdoored models, the ASR of the reversed triggers is below 40%, which means they cannot be recognized as backdoor triggers, and therefore ABS fails to detect the backdoored model.

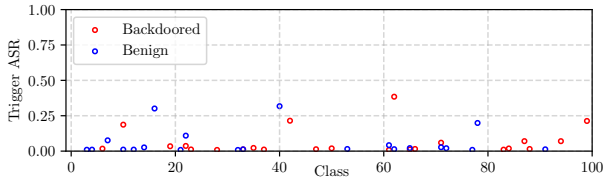


Figure 12: Detection results of ABS on the CIFAR100 dataset.

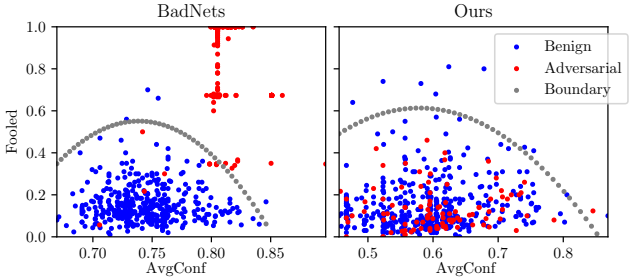


Figure 13: SentiNet detection results on the triggered inputs from BadNets and the aliasing backdoor.

SentiNet [14] is an input inspection method that detects triggers by anomaly detection. It utilizes GradCAM [47] to extract the critical regions of inputs and overlay them on test benign images. Then it computes the average confidence of inputs overlaid with inert patterns and the number of fooled samples overlaid with extracted regions. It trains with benign images and fits a boundary of them. Any inputs outside the boundary at a certain distance are identified as carrying triggers. We backdoor the ResNet18 model on the downstream dataset CIFAR10 using BadNets and our approach. Then we utilize SentiNet to distinguish between inputs with triggers and benign ones. As depicted in Figure 13, the malicious inputs of BadNets are almost (>90%) outside the decision boundary while 99% samples of ours are inside (see visual

samples in Figure 17 of Appendix D). Therefore, SentiNet cannot effectively detect the aliasing backdoor.

Input Filtering. Prior research [41] proposes defensive techniques by filtering model inputs, where backdoor triggers can be likely destroyed. Here we employ two strategies of input filtering to evaluate our attack: 1) *low-pass filtering*, where high-frequency features are removed if they are out of the frequency cutoff D_0; 2) *selective filtering*, to retain the median or random data points locally within a sliding window.

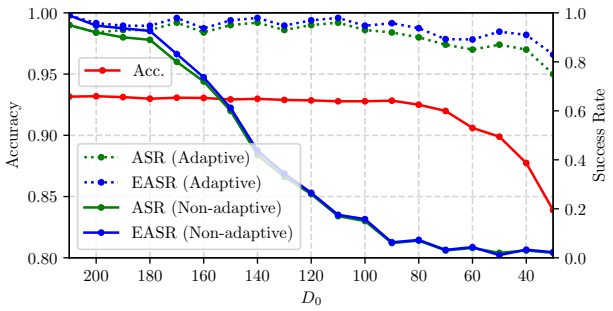


Figure 14: Defense results of low-pass filtering in non-adaptive and adaptive scenarios.

For low-pass filtering, we refer to the ideal low-pass filter [22] and test on the ViT-S/16/384 model fine-tuned with the Pets dataset in the fixed-feature setting (see Table 2). We ensure the mean PSNR of attack samples higher than 18 dB during attacks. Additionally, as input filtering is readily anticipated in reality, the attacker may leverage an adaptive method to generate more robust triggers. It can be easily implemented by using input filtering as preprocessing during trigger generation and leveraging the penultimate layer as the feature layer $\hat{f}$ in Equation 8 to extract higher-level features.

Figure 14 shows the defense results against non- and adaptive attacks. In the non-adaptive scenario, the low-pass filters of D_0 lower than 90 can reduce the attack success rate to less than 10%. However, in the adaptive scenario, the attack success rate can maintain at a high level of over 80% when the D_0 is below 40; meanwhile, it results in an accuracy decrease of roughly 10%. The adaptive attack is feasible since D_0 can be easily guessed by the attacker. It can use a collection of attack samples generated using varying D_0 values to easily deduce the true D_0 used by the defender.

For selective filtering, although they can decrease the non-adaptive success rate to less than 2.11%, the attack still maintains a high EASR of over 90% for the selective median filter and 71% for the selective random filter in the adaptive scenario (see Table 6). Additionally, the random filter results in a considerable drop in model accuracy of 4.58%.

Overall, input filtering cannot cope with the real-world scenario, which the attacker can easily anticipate and overcome in an adaptive manner.

Weights Smoothing. We test another defensive technique, i.e., smoothing the weights of neural networks. Inspired by the convolution theorem [22], we apply low-pass filtering to smooth the weights before fine-tuning pre-trained models, reducing the high-frequency components in inputs. To further evaluate the effectiveness of this method, we also explore the adaptive scenario wherein the attacker is aware of weights smoothing during trigger generation. Similar to the adaptive scenario in input filtering, the attacker generates triggers by optimizing the features extracted from the smoothed and backdoored model.

Table 6: Defense results of selective filtering.

Filter	Acc./ΔAcc.	Non-adaptive ASR/EASR	Adaptive ASR/EASR
w/o filter	93.16%/0%	95%/98.95%	95%/98.95%
Selective median	93.10%/0.06%	2%/2.11%	87%/90.53%
Selective random	88.58%/4.58%	0%/0.00%	67%/71.59%

Table 7: Defense results of weights smoothing.

D_0	Full-network Accuracy	Non-adaptive ASR/EASR	Adaptive ASR/EASR	Fixed-feature Accuracy
6.0	93.65%	57%/61.96%	91%/96.74%	93.21%
4.5	92.64%	28%/31.46%	86%/92.13%	92.40%
3.0	92.37%	13%/14.29%	87%/92.31%	90.46%
2.0	90.98%	12%/14.12%	31%/36.47%	71.87%
1.0	88.14%	9%/10.34%	15%/16.09%	35.79%

We evaluate this defense with the ViT-S/16/384 model on the Pets dataset. Table 7 shows that smoothing with a cutoff frequency of 3.0 can reduce ASR/EASR to 13% and 14% in the non-adaptive scenario, respectively. However, the attack can still achieve an EASR over 90% when the D_0 is greater than 3.0 in the adaptive scenario. When smoothing the weights with D_0 of 1, although the adaptive success rate drops to 15%, it causes an accuracy loss of roughly 5.5%. To significantly decrease the success rate, the defender must smooth with D_0 less than 2.0, reducing the adaptive success rate to less than 36.5%. However, we found that full-network fine-tuning is indispensable at this level of D_0; fine-tuning a network after smoothing only at the fixed-feature setting can result in a substantial drop in accuracy by over 20%. It is unacceptable for users who opt for transfer learning due to limited computational resources, as fixed-feature fine-tuning is a commonly used strategy [2, 30]. Even more, we evaluate other methods including re-initializing weights and inspecting weight distributions for defending backdoors in Appendix D. The results show that these methods are very limited in either model accuracy retaining, or detection ability for backdoors.

6 Discussion

Previous backdoor attacks were unaware of exploiting ubiquitous strided layers for backdoor attacks. We unveiled a new attack surface and offered insight into related research. Since

the backdoor works by aliasing, our proposed backdoor is applicable to pre-trained models containing subsampling operations. We have proven the feasibility of exploiting the convolutional strided layers of ConvNets and patch embedding layers of vision transformers. As they typically exist at the foremost front of model architectures [4], we can manipulate their output by directly perturbing the input. The aliasing backdoor insertion is free of any data demand or parameter tuning and finished within several minutes, thus the attackers can quickly spread a lot of backdoored models and threaten potential victims.

There are other layers containing subsampling operations such as pooling layers [32, 42]. Theoretically, max-pooling [42] can also result in aliasing issues [42] that may be exploited. In contrast, average-pooling [32] alleviates the aliasing of subsampling by averaging all elements of the input, which can block high-frequency noises.

7 Related Work

Backdoor Attacks. Data poisoning is a common practice used for backdoor insertion [12, 25, 48, 62] but it requires the accessibility of the victims' training data, which is difficult to realize. Recently, studies have tried to target the transfer learning scenario by backdooring pre-trained models. Some works [28, 43, 59] explore inserting backdoors to the feature extractor of pre-trained models, therefore, they can survive from last-layer fine-tuning. These works still demand the training data of the victim task which is unpractical. Recently, data-free backdoor attacks are also proposed. TrojanNN [34] proposes to generate training data by reverse engineering; DBIA [35] proposes to inject a backdoor into the vision transformer models by substitute data. These works still demand heavy model re-training which means high attack costs. TrojanNet [53] is a training-free and data-free backdoor but model architecture modification is required. Compared to these data-free works, the backdoor insertion of our work is not only data-free but also free of data generation or architecture modification. The above backdoors are hard to survive in multi-layer or all-layer fine-tuning and only work on a specific dataset.

Aliasing of Neural Networks. Aliasing is a conception in digital signal processing [37], which refers to the distortion when re-sampling the signals with inadequate sampling rates. For images and audio, aliasing happens in spatial and temporal domains respectively. Since down-sampling techniques such as strided convolutional and pooling layers are ubiquitous in deep neural networks [26, 51], aliasing also happens in both ConvNets [17, 61] and transformers [40]. Previous works have demonstrated that aliasing of ConvNets can result in poor generalization ability, i.e., sensitivity to small shifts in inputs [9]. However, they have not pointed out the potential security risk of aliasing. Although some works [40, 61] have proposed their anti-aliasing techniques, they demand model architecture modification and training from scratch. Therefore, they cannot be used in transfer-learning scenarios where users do not have resources for model re-training. The aliasing is also explored in the image-scaling attack [41, 58], which aims to change the visual semantics of one image through scaling algorithms. Differently, our attack exploits the intrinsic weakness of neural networks, rather than external preprocessing algorithms, to insert a stealthy backdoor.

8 Conclusion

In this paper, we propose the aliasing backdoor attack on the pre-trained models. To the best of our knowledge, this is the first work to exploit the strided layers to launch backdoor attacks. The aliasing backdoors exist at the strided layers and threaten all downstream student models and datasets. Our evaluation shows that aliasing backdoors exhibit high effectiveness, evasiveness with strong survivability and sufficient stealthiness. With the popularity of pre-trained models in transfer learning, such low-cost and data-free backdoor attacks significantly threaten deep-learning-based applications.

Acknowledgements

We thank all the reviewers for their insightful comments and express our sincere gratitude for Shepherd's assistance. The IIE CAS authors are supported in part by the National Key R&D Program of China (2020AAA0140001), NSFC (92270204), Beijing Natural Science Foundation (No.M22004), Beijing Nova Program, Youth Innovation Promotion Association CAS and a research grant from Huawei. The HYU author is supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT)(NRF-2022R1F1A1074999).

References

[1] Azure text to speech. Website, 2022. https://azure.microsoft.com/en-us/products/cognitive-services/text-to-speech/.

[2] Deep learning for computer vision. Website, 2022. https://cs231n.github.io/transfer-learning/.

[3] Hugging face. Website, 2022. https://huggingface.co/.

[4] Image classification on imagenet. Website, 2022. https://paperswithcode.com/sota/image-classification-on-imagenet.

[5] Modelzoo. Website, 2022. https://modelzoo.co/.

[6] Vision transformer and mlp-mixer architectures project. Website, 2022. https://github.com/google-research/vision_transformer/blob/main/vit_jax/models_vit.py#L259.

[7] Wav2vec2 base model. Website, 2022. https://huggingface.co/patrickvonplaten/wav2vec2-base-100h-with-lm/tree/main/language_model.

[8] Shaden Alshammari, Yu-Xiong Wang, Deva Ramanan, and Shu Kong. Long-tailed recognition via weight balancing. In *CVPR*, 2022.

[9] Aharon Azulay and Yair Weiss. Why do deep convolutional networks generalize so poorly to small image transformations? *arXiv preprint arXiv:1805.12177*, 2018.

[10] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *NeurIPS*, 2020.

[11] Qiong Cao, Li Shen, Weidi Xie, Omkar M Parkhi, and Andrew Zisserman. Vggface2: A dataset for recognising faces across pose and age. In *FG*, 2018.

[12] 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.

[13] Yuxuan Chen, Xuejing Yuan, Jiangshan Zhang, Yue Zhao, Shengzhi Zhang, Kai Chen, and XiaoFeng Wang. Devil's whisper: A general approach for physical adversarial attacks against commercial black-box speech recognition devices. In *USENIX Security*, 2020.

[14] Edward Chou, Florian Tramer, and Giancarlo Pellegrino. Sentinet: Detecting localized universal attacks against deep learning systems. In *S&P*, 2020.

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[16] Alexey Dosovitskiy, Lucas Beyer, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021.

[17] Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A rotation and a translation suffice: Fooling CNNs with simple transformations, 2019.

[18] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *CVPR Workshop*, 2004.

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

[20] John S Garofolo. Timit acoustic phonetic continuous speech corpus. *Linguistic Data Consortium*, 1993.

[21] Stanley A Gelfand. *Essentials of audiology*. Thieme New York, 1997.

[22] Rafael C Gonzalez. *Digital image processing*. Pearson education india, 2009.

[23] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *ICML*, 2006.

[24] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007.

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

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

[27] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *NeurIPS*, 2019.

[28] Yu Ji, Zixin Liu, Xing Hu, Peiqi Wang, and Youhui Zhang. Programmable neural network trojan for pre-trained feature extractor. *arXiv preprint arXiv:1901.07766*, 2019.

[29] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *ECCV*, 2020.

[30] Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *CVPR*, 2019.

[31] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[32] Yann LeCun, Bernhard Boser, John Denker, Donnie Henderson, Richard Howard, Wayne Hubbard, and Lawrence Jackel. Handwritten digit recognition with a back-propagation network. *NeurIPS*, 1989.

[33] Yingqi Liu, Wen-Chuan Lee, Guanhong Tao, Shiqing Ma, Yousra Aafer, and Xiangyu Zhang. Abs: Scanning neural networks for backdoors by artificial brain stimulation. In *CCS*, 2019.

[34] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *NDSS*, 2018.

[35] Peizhuo Lv, Hualong Ma, Jiachen Zhou, Ruigang Liang, Kai Chen, Shengzhi Zhang, and Yunfei Yang. Dbia: Data-free backdoor injection attack against transformer networks. *arXiv preprint arXiv:2111.11870*, 2021.

[36] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*, 2008.

[37] Harry Nyquist. Certain topics in telegraph transmission theory. *Transactions of the American Institute of Electrical Engineers*, 1928.

[38] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *ICASSP*, 2015.

[39] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *CVPR*, 2012.

[40] Shengju Qian, Hao Shao, Yi Zhu, Mu Li, and Jiaya Jia. Blending anti-aliasing into vision transformer. *NeurIPS*, 2021.

[41] Erwin Quiring, David Klein, Daniel Arp, Martin Johns, and Konrad Rieck. Adversarial preprocessing: Understanding and preventing {Image-Scaling} attacks in machine learning. In *USENIX Security*, 2020.

[42] Marc'Aurelio Ranzato, Fu Jie Huang, Y-Lan Boureau, and Yann LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *CVPR*, 2007.

[43] Shahbaz Rezaei and Xin Liu. A target-agnostic attack on deep models: Exploiting security vulnerabilities of transfer learning. *arXiv preprint arXiv:1904.04334*, 2019.

[44] Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. Imagenet-21k pretraining for the masses. *arXiv preprint arXiv:2104.10972*, 2021.

[45] C.D. Castillo V.M. Patel R. Chellappa D.W. Jacobs S. Sengupta, J.C. Cheng. Frontal to profile face verification in the wild. In *WACV*, 2016.

[46] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015.

[47] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017.

[48] Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. *NeurIPS*, 2018.

[49] Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *arXiv preprint arXiv:2106.10270*, 2021.

[50] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *ICCV*, 2017.

[51] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, 2017.

[52] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

[53] Ruixiang Tang, Mengnan Du, Ninghao Liu, Fan Yang, and Xia Hu. An embarrassingly simple approach for trojan attack in deep neural networks. In *KDD*, 2020.

[54] Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh. Embedding watermarks into deep neural networks. In *ICMR*, 2017.

[55] 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 *S&P*, 2019.

[56] Tianhao Wang and Florian Kerschbaum. Attacks on digital watermarks for deep neural networks. In *ICASSP*, 2019.
[57] Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019.
[58] Qixue Xiao, Yufei Chen, Chao Shen, Yu Chen, and Kang Li. Seeing is not believing: Camouflage attacks on image scaling algorithms. In *USENIX Security*, 2019.
[59] Yuanshun Yao, Huiying Li, Haitao Zheng, and Ben Y Zhao. Latent backdoor attacks on deep neural networks. In *CCS*, 2019.
[60] Xuejing Yuan, Yuxuan Chen, Yue Zhao, Yunhui Long, Xiaokang Liu, Kai Chen, et al. Commandersong: A systematic approach for practical adversarial voice recognition. In *USENIX security*, 2018.
[61] Richard Zhang. Making convolutional networks shift-invariant again. In *ICML*, 2019.
[62] Chen Zhu, W Ronny Huang, Hengduo Li, Gavin Taylor, Christoph Studer, and Tom Goldstein. Transferable clean-label poisoning attacks on deep neural nets. In *ICML*, 2019.

A Pre-trained Models & Fine-tuning Datasets

Table 8: Pre-trained models and their parameters.

Model	Training Data	Params	Input[1]	Stride[2]
ViT-S/16/384		22M	384×384	16
ViT-S/16/224	JFT-300M [50]	22M	224×224	16
ViT-B/16/224	ImageNet [15]	87M	224×224	16
ViT-B/8/224		87M	224×224	8
ResNet50/21k	ImageNet-21K [44]	26M	224×224	2
ResNet18		12M	224×224	2
ResNet50	ImageNet [15]	26M	224×224	2
ResNet101		45M	224×224	2
FaceNet	VGGFace2 [11]	56M	160×160	2
Wav2Vec2-Base	Librispeech [38]	95M	16 kHz	5

[1] Image input size or audio sampling rate.
[2] Stride of the strided layer.

Table 9: Datasets for model fine-tuning.

Dataset	Classes	Size (train/test)[1]
CIFAR-10 [31]	10	50000/10000
CIFAR-100 [31]	100	50000/10000
Oxford-IIIT Pets [39]	37	3680/3669
Oxford 102 Flowers [36]	102	2040/6149
Caltech-101 [18]	102	3153/5991
Caltech-256 [24]	257	15360/15247
CFP [45]	500	3500/3500
TIMIT [20]	N/A	5h/3.6h

[1] There is no overlapping between train and test splits. We subset the train split for validation.

The pre-trained models used in this paper are shown in Table 8. All the ViT models and ResNet models are from library Timm [57] except the pre-trained ResNet50/21k model is from [44]. The datasets we utilize are listed in Table 9. They are typical transfer learning tasks, widely used to evaluate the performance of pre-trained models in prior studies [16,30,44].

B Attack Settings

Fine-tuning in Comparison to Baselines. For the BadNets/TrojanNet/Latent Backdoor baselines, we fine-tune them to evaluate their survivability with the full-network setting. We set the learning rate as 0.005 and utilize an Adam optimizer with a batch size of 64, momentum of 0.9, and weight decay of $1e-4$. We fine-tune the models of baselines the same as our models for 20 epochs.

Image Classification. For all datasets, we utilize two fine-tuning settings mentioned in Section 5.1. During fine-tuning, we fix the running means and variances of normalization layers. Following previous work [30], we use the SGD optimizer with a batch size of 64, momentum of 0.9, and weight decay of $1e-4$. The learning rates vary from 0.001 to 0.01 for different datasets and models with an exponential learning rate schedule of decay of 0.9. To avoid overfitting, we adopt an early stopping strategy with tolerance 5 and 100 training epochs. For the same task, the backdoored model and clean model are fine-tuned at the same setting.

Face Recognition. We fine-tune the FaceNet [46] model pre-trained on VGGFace2 [11] with the feature extractor fixed on the CFP [45] dataset for 10 epochs. The optimizer is Adam with a learning rate of $1e-3$ and a batch size of 32.

Speech Recognition. The Wav2Vec2 [10] is pre-trained by self-supervision on unlabelled Librispeech [38]. We select the base model of it and utilize TIMIT [20] as fine-tuning data. We follow the official instruction [10] to fine-tune the model with CTC [23] and frozen encoder part. At the inference phase, we connect the CTC acoustic model with a public 4-gram language model [7].

Target Commands in Speech Recognition Attack: "what is the weather", "call my wife", "where is my car", "turn off the computer", "open my door", "navigate to my house", "turn on airplane mode", "call nine one one", "take a picture", "clear notification".

C Automatic Parameter Selection

In Algorithm 1 we describe how to automatically determine the parameters β_1, β_2. We first try to achieve the attack without an aliasing intensity limitation, thus we can find a reference value of β_1. While increasing β_1, we need to maintain the same prediction of attack sample x and target sample x_1 on the pre-trained model M; meanwhile, the attack sample's stealthiness $\psi(x, x_2)$ (e.g., peak-signal-to-noise of images) should be greater than the specified threshold γ. Once β_1 is determined, we increase β_2 gradually to force the aliasing intensity to decrease to a specified level η.

Obtained from Algorithm 1, β_2 can be used as reference parameters. Then the attacker can manually change the value of β_2 to change the aliasing intensity of backdoors which balances the model performance and attack effectiveness. Intuitively, a higher β_2 yields stronger aliasing intensity. Thus

Algorithm 1: Automatic Parameter Search

Input: pre-trained model M and its strided layer $\hat{f}$; inducing samples x_1, x_2; stealthiness metric ψ; stealthiness threshold γ; aliasing intensity η; step sizes α_1, α_2 for β_1, β_2

Output: β_1, β_2

$x \leftarrow x_1$

$\beta_2 \leftarrow 0$ `// unlimited aliasing intensity at beginning`

$\beta_1 \leftarrow 1$

`//` β_1 `search`

P^k, x=TransformationSearch$(\hat{f}, x_1, x_2, \beta_1, \beta_2)$ `// Equation 7`

while $\psi(x, x_2) < \gamma$ & $M(x)$=$M(x_1)$ **do** `// e.g., PSNR<18 dB`

  $\beta_1 \leftarrow \beta_1 + \alpha_1$

  P^k, x=TransformationSearch$(\hat{f}, x_1, x_2, \beta_1, \beta_2)$

`//` β_2 `search`

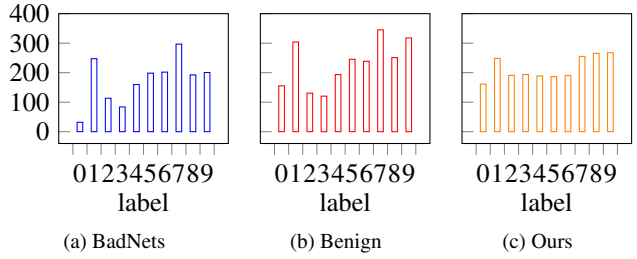
while $D_I(P^k) > \eta$ **do** `// e.g.,` D_I `> 1.0`

  $\beta_2 \leftarrow \beta_2 + \alpha_2$

  P^k, x=TransformationSearch$(\hat{f}, x_1, x_2, \beta_1, \beta_2)$

the attacker can flexibly adjust the backdoor accordingly.

D More Experiments on Defenses



(a) BadNets (b) Benign (c) Ours

Figure 15: Trigger L-1 norm reversed by Neural Cleanse on ResNet18 and CIFAR10. The target label of BadNets is 0.

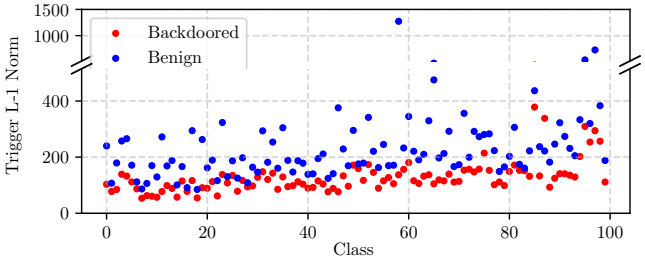


Figure 16: Neural Cleanse results on the CIFAR100-LT (IF=100).

Neural Cleanse. In Figure 15, the result on CIFAR10 indicates that our backdoored model shows no obvious anomaly on any specific label.

To further assess the detectability of backdoors in reality, we construct a long-tail scenario wherein a small number of popular classes possess the majority of the data and other classes have very few samples, forming a "long tail" distribution. Specifically, we backdoor and fine-tune the pre-trained ResNet50/21k model on the CIFAR100-LT [8] dataset, which exhibits an imbalanced factor (i.e., the ratio between popular classes samples and tail classes samples) of 100. We then apply Neural Cleanse to reverse the trigger of the fine-tuned model and compare the results to those obtained from a benign model. As depicted in Figure 16, the backdoored model does not produce a significantly lower outlier that could be indicative of an infected label. Similarly, we found that the L-1 norms of the triggers associated with the backdoored model are lower than those of the benign model, just like the balanced scenario.

SentiNet. Figure 17 shows the critical regions marked by GradCAM. The heatmaps indicate that attack samples share similar critical regions as target samples. Therefore, the inputs with triggers cannot be distinguished from benign ones.

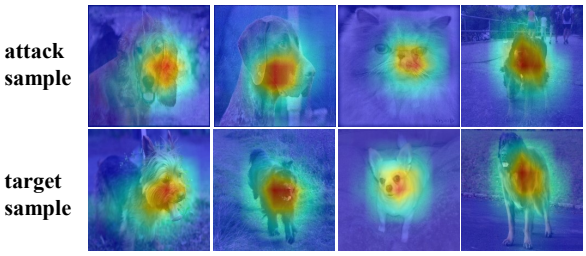


Figure 17: Critical regions between attack samples and target samples identified by GradCAM.

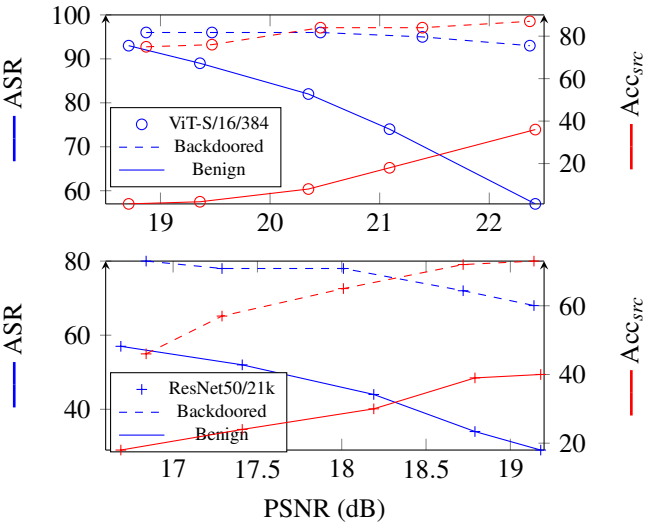


Figure 18: ASR/Acc$_{src}$ of benign and backdoored models with different PSNR. We utilize a benign ViT-S/16/224 model to evaluate Acc$_{src}$.

Aliasing Inspection. As our attack is based on the aliasing effect, it seems anti-aliasing works may help defend against

our attack. However, existing works are striving to design new model architectures with anti-aliasing modules [40, 61]. That is, they require modifying the network architecture and re-training the pre-trained model, which contradicts the goal of transfer learning.

Compared to benign models, models with aliasing backdoors are more suffering from aliasing and therefore the output of their strided layers can be manipulated. As we have mentioned in Section 5.2, the B4 attack yields attack samples of much worse actual stealthiness. The difference between the B4 attack and the aliasing backdoor attack is whether there exists an aliasing backdoor in the model. Therefore, given a suspicious model, the victim can assume it is backdoored and try to generate samples by the B4 attack.

Figure 18 shows that the ASR of the backdoored model is much higher than that of the benign model whose ASR significantly drops when PSNR is greater than 22 dB. We also found that the generated samples of the benign model are of much worse stealthiness and quality; the Acc_{src} is lower than 40% even when the PSNR is greater than 22 dB. As shown in Figure 19, the generated samples of the benign model exhibit poor quality and obvious human-recognizable features from the target samples while the backdoored model does not.

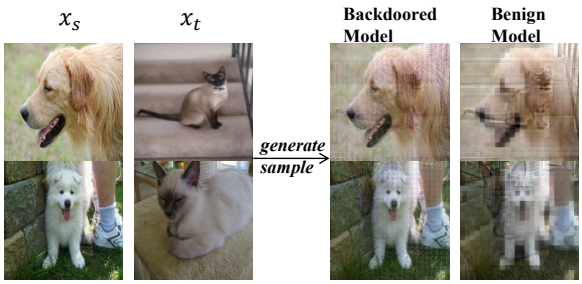
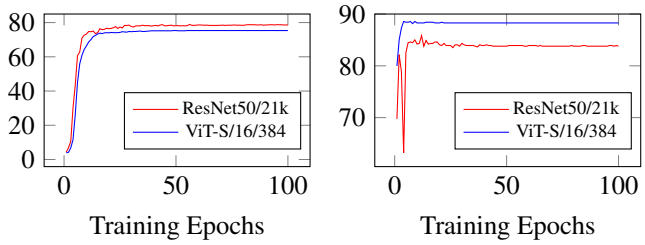


Figure 19: Some generated samples of the benign and backdoored models.

Weight Re-initialization. Since the backdoor exists at the strided layer, the victim can re-initialize its weight and re-train the model to remove the backdoor. However, according to our experiments, the model cannot converge if the victim re-initializes the strided layers before fine-tuning. It is because once the earlier layers are randomized, all gradients obtained to update the later layers deviate from the correct convergence direction. Therefore, we re-initialize the backdoored layer and re-train it for extra 100 epochs (phase 1), after the model has been fine-tuned in a downstream task. Figure 20 indicates that the model converges but with a much lower accuracy (<80%). To improve the performance, we continue to fine-tune the full network for another 100 epochs (phase 2). The results are still significantly lower than what benign models yield in Table 2 (by 7.13%/5.48% for ResNet50/21k and ViT-S/16/384 respectively). Therefore, this defense is not practical.

Weights Inspection. The weights of backdoored models may exhibit distinctive features, which can be used for backdoor



(a) Phase 1: Retrain the strided layer (b) Phase 2: Retrain the full network

Figure 20: Accuracy of re-initializing and re-training the backdoored strided layer on the Pets dataset.

detection. To verify this assumption, we follow prior studies [54, 56] and extract histograms of weights from both the backdoored and benign strided layers.

Figure 21 (a) shows the distributions of two benign models and a backdoor model. It reveals that the weight distribution of the backdoored ResNet50/21k model is steeper than its original, but shows no obvious features with another benign ResNet50 model of the same architecture (i.e., the "gluon_resnet50_v1b" model from Timm [57]). Meanwhile, the same phenomenon happens for the ViT-S/16/384 model, and backdoors with higher aliasing intensity D_I cause higher peaks, as shown in Figure 21 (b). One reasonable explanation is that the aliasing backdoor needs to strengthen the weights that are related to backdoors but weaken the others. This phenomenon is also observed in the scaling attack [41], where many pixels have smaller weights in the kernel of scaling algorithms.

From the view of backdoor detection, weights inspection is not an effective method, as it cannot tell apart the weight distributions between backdoored models and benign ones.

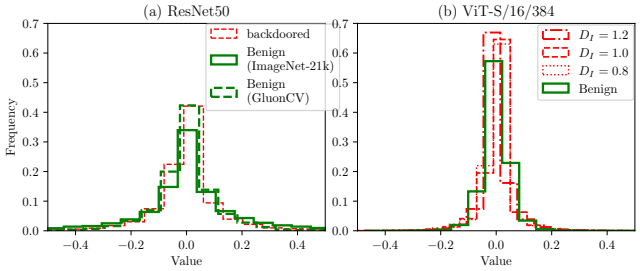


Figure 21: (a) The distribution of weight parameters of the backdoored/benign ResNet50/21k model, as well as another benign ResNet50 model from GluonCV; (b) The weight distribution of backdoored ViT-S/16/384 models with varying levels of D_I.

E Impact of Other Factors

Impact of Stride Size. To explore the impact of the stride sizes of strided layers, we replace the front strided layer of the

ResNet18 model with convolutional layers of kernel size 3 and varying stride sizes ranging from 2 to 7. Subsequently, we pre-train these modified models on CIFAR10 and insert the aliasing backdoor with the same D_I value of 0.8. To ensure the basic performance of models with varying strides, we set the input size as 384, avoiding of information loss for CIAFR10 whose image size is 32. The success rates and PSNR values of the fine-tuned student models, which were trained using these backdoored teacher models, were evaluated on CIFAR10. As depicted in Figure 22, we observed that backdoors with larger stride sizes (>4) exhibited higher PSNR values when achieving the same success rate, indicating improved stealthiness. This is attributed to the impact of larger stride sizes. Similar to the scaling attack [41], the stride size plays a similar role as the scaling ratio. They determine how many pixels are utilized during subsampling. With a larger stride size, fewer pixels are needed to be modified, so that better stealthiness can be achieved under the same success rate.

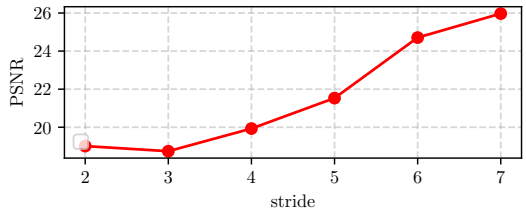


Figure 22: The PSNR of attack samples when backdooring networks with varying strides. The attack success rates (EASR) of these backdoors are 90%.

Impact of Inducing Samples. Here we show how different types of inducing samples impact the weight modification, and further influence attack performance. As illustrated in Figure 23, for the same value of D_I, both Gaussian noise with $\mathcal{N}(0,1)$ and random uniform noise with $U(0,1)$ lead to a smaller L_W than meaningful inducing samples, indicating a reduced need for weight modification. Therefore, it leads to better model accuracy after backdoor insertion. On the other hand, it causes worse attack effectiveness and stealthiness compared to meaningful inducing samples, as illustrated in Figure 10.

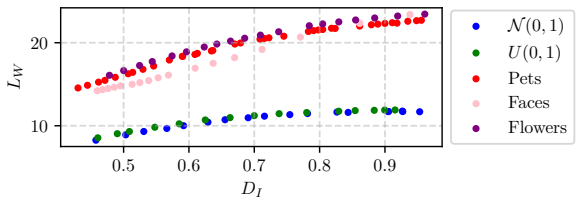


Figure 23: D_I and L_W of different inducing samples.

Impact of Model Selection. Table 10 summarizes the evasiveness performance when backdooring various models. More details about these models are listed in Table 8. For comparison, we make the D_I of all models close to 1, which results in relatively intense aliasing but a more obvious comparison. We can see that the ViT-S of bigger input size shows better evasiveness; and with the same input size, models of larger parameter size show better evasiveness. Meanwhile, all models pre-trained on large-scale datasets (JFT-300M or ImageNet-21K) achieve better results than models only pre-trained on ImageNet. These results indicate a positive effect of larger input size, parameter size, and pre-training data size.

From Figure 24, we see an obvious trend that the results of vision transformers are much better than that of ResNets, which conforms to the image classification results in Table 2. Compared to ViTs with smaller input sizes, ViT-S/16/384 achieves better attack performance at all metrics, which highlights the influence of input size. We notice that recently more and more works are tending to utilize larger input sizes than 224 to achieve better performance on downstream datasets [16, 30]. It indicates that aliasing backdoors pose a greater threat to these models.

We found that ViT-B/8/224 of smaller stride size achieves better Acc./ΔAcc. and EASR than ViT-B/16/224. It may attribute to the patch embedding layer used in vision transformers. With a smaller patch embedding size, the model can extract more fine-grained context information of patches to obtain better performance.

From the viewpoint of users, it is better for attackers to share high-performance models. Our benign model accuracies are comparable to those reported for the similar-size models, e.g., the ViT paper [16]. Also, higher accuracy can be achieved by backdooring larger state-of-the-art models.

Table 10: Performance of various image models.

Model	Params	D_I	L_W	Acc.	ΔAcc.
ViT-S/16/384	22M	1.0	30.12	92.99%	0.48%
ViT-S/16/224	22M	1.0	28.40	89.22%	2.73%
ViT-B/16/224	87M	1.0	14.15	90.55%	2.22%
ViT-B/8/224	87M	1.0	11.40	92.60%	1.28%
ResNet50/21k	26M	1.0	21.82	90.26%	1.67%
ResNet50	26M	1.0	27.57	88.31%	4.04%
ResNet101	45M	1.0	24.04	88.71%	3.26%

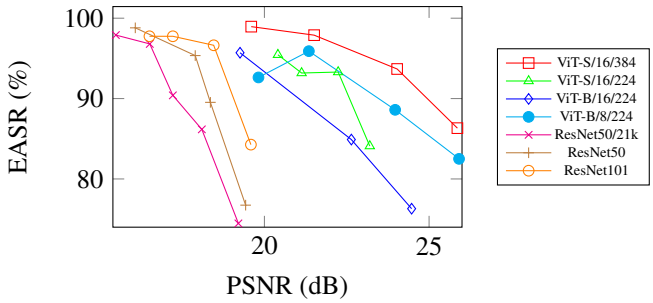


Figure 24: EASR of various pre-trained models ($D_I = 1$).