Towards Backdoor Stealthiness in Model Parameter Space

Xiaoyun Xu
Radboud University
Nijmegen, The Netherlands
xiaoyun.xu@ru.nl

Zhuoran Liu*
Radboud University
Nijmegen, The Netherlands
z.liu@cs.ru.nl

Stefanos Koffas
Delft University of Technology
Delft, The Netherlands
s.koffas@tudelft.nl

Stjepan Picek
Radboud University, Nijmegen, The Netherlands
& University of Zagreb Faculty of Electrical Engineering
and Computing, Unska 3, 10000, Zagreb, Croatia
stjepan.picek@ru.nl

Abstract

Backdoor attacks maliciously inject covert functionality into machine learning models, representing a security threat. The stealthiness of backdoor attacks is a critical research direction, focusing on adversaries' efforts to enhance the resistance of backdoor attacks against defense mechanisms. Recent research on backdoor stealthiness focuses mainly on indistinguishable triggers in *input space* and inseparable backdoor representations in *feature space*, aiming to circumvent backdoor defenses that examine these respective spaces. However, existing backdoor attacks are typically designed to resist a specific type of backdoor defense without considering the diverse range of defense mechanisms. Based on this observation, we pose a natural question: *Are current backdoor attacks truly a real-world threat when facing diverse practical defenses?*

To answer this question, we examine 12 common backdoor attacks that focus on input-space or feature-space stealthiness and 17 diverse representative defenses. Surprisingly, we reveal a critical blind spot that backdoor attacks designed to be stealthy in input and feature spaces can be mitigated by examining backdoored models in *parameter space*. To investigate the underlying causes behind this common vulnerability, we study the characteristics of backdoor attacks in the parameter space. Notably, we find that input- and feature-space attacks introduce prominent backdoor-related neurons in parameter space, which are not thoroughly considered by current backdoor attacks. Taking comprehensive stealthiness into account, we propose a novel supply-chain attack called `Grond`. `Grond` limits the parameter changes by a simple yet effective module, Adversarial Backdoor Injection (ABI), which adaptively increases the parameter-space stealthiness during the backdoor injection. Extensive experiments demonstrate that `Grond` outperforms all 12 backdoor attacks against state-of-the-art (including adaptive) defenses on CIFAR10, GTSRB, and a subset of ImageNet. Additionally, we show that ABI consistently improves the effectiveness of common backdoor attacks. Our code is publicly available: https://github.com/xiaoyunxxy/parameter_backdoor.

* Corresponding author.

This work is licensed under a Creative Commons Attribution 4.0 International License.
CCS '25, Taipei, Taiwan
© 2025 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1525-9/2025/10
https://doi.org/10.1145/3719027.3744846

CCS Concepts

• **Security and privacy**; • **Computing methodologies** → **Artificial intelligence**;

Keywords

Backdoor Attack, Backdoor Defense, Parameter Space, Stealthiness

ACM Reference Format:
Xiaoyun Xu, Zhuoran Liu*, Stefanos Koffas, and Stjepan Picek. 2025. Towards Backdoor Stealthiness in Model Parameter Space. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security (CCS '25), October 13–17, 2025, Taipei, Taiwan.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3719027.3744846

1 Introduction

While deep neural networks (DNNs) have achieved excellent performance on various tasks, they are vulnerable to backdoor attacks. Backdoor attacks insert a secret functionality into a model, which is activated by malicious inputs during inference. Such inputs contain an attacker-chosen property called the trigger. Backdoored DNNs can be created by training with poisoned data [5, 14, 39, 54]. More powerful and stealthy backdoors can also be injected through the control of a training process [1, 7, 36, 42, 48, 71, 73], or by direct weights modification of the victim model [2, 18].

In early backdoor attacks [5, 14, 30], triggers could induce noticeable changes that human inspectors or anomaly detectors [3, 29, 56] could easily spot. To enhance the ability to remain undetected against such defenses (i.e., achieve *input-space stealthiness*), smaller or more semantic-aware triggers are designed [10, 39, 59]. Input-space stealthy backdoor attacks usually need to change labels of poisoned samples to the target class (i.e., dirty-label), which makes detection easier [3]. To this end, another line of backdoor attacks poisons the training data without changing the labels [54, 69] (i.e., clean-label), improving backdoor stealthiness.

Despite the stealthiness concerning input images and labels, it has been widely observed that existing backdoor attacks introduce separable representations in the feature space, which can be exploited to develop backdoor defenses [35, 41, 57, 64, 74]. For example, featureRE [57] utilizes feature separability and designs a feature space constraint to reverse-engineer the backdoor trigger. In response to feature-space defenses, state-of-the-art (SOTA)

Full version with appendices: [65].

Table 1: A summary of the existing defenses evaluated in this paper. *"Proactively training"* refers to the strategy that the defender could proactively control the training on poisoned training data to produce a clean model without a backdoor in it. Additionally, all the defenses have been tested against the all-to-one attack, so we omitted it from the attack assumptions. A summary of backdoor attacks is provided in Table 13 in [65, App. B].

Defense		Defense Task			Threat Model			Attack Assumption	
		Input detection	Model detection	Mitigation	Black-box	Needs clean data	Proactively training	All-to-all	Dynamic
Model Inspection	NC [56]	○	●	○	○	●	○	○	○
	Tabor [16]	○	●	○	○	●	○	○	○
	FeatureRE [57]	○	●	○	○	●	○	○	●
	Unicorn [58]	○	●	○	○	●	○	○	●
	BTI-DBF [64]	○	●	○	○	●	○	●	●
Input Inspection	Scale-up [15]	●	○	○	●	○	○	●	●
	IBD-PSC [19]	●	○	○	○	●	○	●	●
	CT [43]	●	○	○	○	●	●	●	●
Pruning	FP [28]	○	○	●	○	●	○	●	●
	ANP [62]	○	○	●	○	●	○	●	●
	CLP [72]	○	○	●	○	○	○	●	●
	RNP [24]	○	○	●	○	●	○	●	●
Fine-tune	vanilla FT	○	○	●	○	●	○	●	●
	FT-SAM [74]	○	○	●	○	●	○	●	●
	I-BAU [68]	○	○	●	○	●	○	●	●
	FST [35]	○	○	●	○	●	○	●	●
	BTI-DBF(U) [64]	○	○	●	○	●	○	●	●

○ the item is not supported by the defense; ● the item is supported by the defense.

backdoor attacks focus on eliminating the separability in the feature space [36, 41, 48, 75] to increase the *feature-space stealthiness*, i.e., the undetectability against feature-space defenses. Considering a different threat model, supply-chain backdoor attacks assume control over the training or directly modify the model's weights [2, 7, 18], and the backdoored model is provided as a service or as the final product. For example, supply-chain attacks could introduce a penalty to the training loss that decreases the distance between the backdoor and benign features to increase feature-space stealthiness [9, 48, 71, 73].

An important observation is that most backdoor attacks are designed to be stealthy to resist a specific type of defense. For example, WaNet [39] and Bpp [59] design imperceptible triggers to bypass input-space defenses (such as NC [56]), but introduce significant separability in the feature space [57]. Adap-patch [41] avoids feature separability but uses patch-based triggers, which a human inspector can detect. More critically, current backdoor attacks are barely evaluated against *parameter-space defenses* [24, 27, 62, 66, 72, 74]. This oversight is significant because backdoor behaviors are ultimately embedded in and reflected by the parameters of the backdoored model, which is the final product of any backdoor attack. As such, there is a lack of systematic evaluation of backdoor attacks against the latest *parameter-space defenses*.

To this end, in this paper, we first systematically analyze 12 attacks against 17 backdoor defenses. All evaluated defenses and their characteristics, including detection and mitigation, are summarized in Table 1. Surprisingly, our experiments demonstrate that parameter-space defenses can easily mitigate SOTA stealthy backdoor attacks (including supply-chain attacks), indicating that existing stealthy backdoor attacks fail to provide *parameter-space stealthiness* and, as a result, still need substantial improvement to be stealthy in the model's parameter space. More importantly, our analysis reveals that even though some backdoor attacks can resist several defenses, bypassing all defense types is far from trivial.

To explore whether it is possible to make backdoor attacks stealthy simultaneously against diverse defenses, we propose a novel attack called `Grond` that considers *comprehensive stealthiness*, meaning that a backdoor attack is stealthy in the input, the feature, and the parameter space of the model. `Grond` achieves the input space stealthiness by using adversarial perturbations as the trigger. To achieve parameter-space stealthiness, we propose a novel *Adversarial Backdoor Injection* module that adaptively injects the backdoor during the backdoor training to achieve parameter space stealthiness. We also show that the feature-space stealthiness is a by-product of input- and parameter-space stealthiness with empirical results in Figures 7 and 9. Specifically, guided by our TAC analysis, we leverage the Lipschitz continuity of neuron activations to find backdoor-related suspicious and sensitive neurons. Then, we conduct pruning on these neurons to eliminate the backdoor effect. As a result, the backdoor is associated with neurons throughout the DNN rather than just focusing on a few prominent neurons after Adversarial Backdoor Injection, as illustrated in Figure 1.

We make the following contributions:

- We revisit SOTA backdoor attacks regarding their stealthiness, showing that most attacks are designed to increase input-space indistinguishability or/and feature-space inseparability without considering parameter-space stealthiness. Based on this finding, we examine common backdoor attacks and reveal a critical *blind spot* regarding real-world scenarios: SOTA stealthy backdoor attacks are highly vulnerable to parameter-space defenses.
- To investigate the underlying reasons behind this common vulnerability of backdoor attacks, we take a closer look at the backdoor characteristics in the parameter space, showing that input- and feature-space attacks introduce prominent

backdoor-related neurons, which cannot be avoided by current backdoor attacks.

- To accomplish comprehensive stealthiness, we propose a novel backdoor attack, `Grond`, that considers input, feature, and parameter-space defenses. Extensive experiments demonstrate that `Grond` outperforms SOTA attacks against four pruning- and five fine-tuning-based defenses on CIFAR10, GTSRB, and ImageNet200. Moreover, we demonstrate that `Grond` is resistant against five model detection defenses, two input detection defenses, and a proactive defense.
- We verify the effectiveness of the Adversarial Backdoor Injection module by binding it with other attacks. Experimental results demonstrate that Adversarial Backdoor Injection could substantially improve the parameter-space robustness of most common backdoor attacks.

2 Background & Related Work

2.1 Preliminaries on Backdoor Training

This paper considers a C-class classification problem with an L-layer CNN $f = f_L \circ \cdots f_1$. Suppose that $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$ is the original training data, containing N samples of $\boldsymbol{x}_i \in \mathbb{R}^{d_c \times d_h \times d_w}$ and its label $y \in \{1, 2, \ldots, C\}$. d_c, d_h, and d_w are the number of input channels, the height, and the width of the image, respectively. The attacker chooses a target class t and creates a partially poisoned dataset $\mathcal{D}_p$ by poisoning generators G_x and G_y, i.e., $\mathcal{D}_p = \mathcal{D}_c \cup \mathcal{D}_b$. $\mathcal{D}_c$ is the benign data from original dataset, $\mathcal{D}_b = \{(\boldsymbol{x}', y') | \boldsymbol{x}' = G_x(\boldsymbol{x}), y' = G_y(y), (\boldsymbol{x}, y) \in \mathcal{D} - \mathcal{D}_c\}$. In the clean-label setting, $G_y(y) = y$. For the dirty-label attacks, $G_y(y) = t$. In the training stage, the backdoor is inserted into f by minimizing the loss on $\mathcal{D}_p$:

$$\min_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{D}_p}(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{(\boldsymbol{x}, y) \in \mathcal{D}_p} \ell(f(\boldsymbol{x}; \boldsymbol{\theta}), y). \tag{1}$$

In the inference stage, the trained f performs well on benign data $\hat{\boldsymbol{x}}$, but predicts $G_x(\hat{\boldsymbol{x}})$ as $G_y(\hat{y})$.

2.2 Backdoor Attacks

Backdoor attacks compromise the integrity of the victim model so that the model performs naturally on benign inputs but is misled to the target class by inputs containing the backdoor trigger. The trigger can be a visible pattern inserted into the model's input in the **input space** or a property that affects the feature representation of the model's input in the **feature space**. Eventually, however, the backdoored model's parameters in the **parameter space** will be altered regardless of the exact backdoor attack (see Figure 2). To insert a backdoor, the attacker is assumed to only control a small portion of the training data under the *poison training* scenario [5, 14, 70]. In the *supply-chain* setting (backdoor models provided to users), the attacker also controls the training process [1, 38, 39, 48, 59]. Moreover, the backdoor can also be created by directly modifying the model's weights [2, 18, 31, 42].

Input-space attacks. Traditional attacks typically use simple patterns as their triggers. For example, BadNets [14] uses a fixed patch, and Blend [5] mixes a Hello Kitty pattern into the images as the trigger. These non-stealthy triggers introduce abnormal data into training data and can be easily detected by human inspectors or defenses [3, 56]. To improve the stealthiness, various triggers are proposed to achieve *invisibility* in the input space. IAD [38] designed a dynamic solution in which the triggers vary among different inputs. WaNet [39] proposed the warping-based trigger, which is invisible to human inspection. Although these methods successfully build invisible triggers and bypass traditional defenses [56], they still introduce separable features and can be detected by feature-space defenses [57, 64]. These input-invisible attacks can be even more noticeable than input-visible attacks (BadNet, Blend) in the feature space [66]. We conjecture this is because they have fewer modifications on input pixels than input-visible attacks. Therefore, input-invisible attacks require more influential features to achieve a successful attack.

Feature-space attacks. Knowing the vulnerability of input-space attacks against feature-space defenses, backdoor attacks are improved for feature-space stealthiness. A common threat model of this attack type is to assume additional control over the training process. For example, [9, 48, 71, 73] directly designed loss functions to minimize the difference between the backdoor and benign features. Aside from design loss penalties, TACT [52] and SSDT [36] point out that source-specific (poison only the specified source classes) attack helps to obscure the difference in features between benign and backdoor samples. In addition, [41] proposed Adap-blend and Adap-patch, which obscures benign and backdoor features by 1) including poisoned samples with the correct label, 2) asymmetric triggers (using a stronger trigger at inference time), and 3) trigger diversification (using diverse variants of the trigger during training). Unfortunately, existing attacks lack systematic evaluation against the latest defenses. For example, Adap-blend can be thoroughly mitigated by recent works [64, 66, 74]. In summary, feature-space attacks usually introduce visible triggers and cannot defeat the latest defenses.

Supply-chain attacks. Supply-chain attacks are getting more attention due to their potential in real-world applications where backdoored models are provided as the final product to users. In supply-chain attacks, adversaries could control both training data and the training process. Note that feature-space attacks [7, 9, 26, 36, 46, 48, 63, 71, 73] with the assumption of control over the training process are a subset of supply-chain attacks, as their output is the backdoor model. In addition to training control, another kind of supply-chain attack directly adjusts the model's weights in parameter space to introduce a backdoor, i.e., *parameter-space attack*. T-BFA [45], TBT [44], and ProFlip [4] explore modifying a sequence of susceptible bits of DNN parameters stored in the main memory (e.g., DRAM) to inject the backdoor. SRA [42] and handcrafted Backdoor [18] directly modify a subset of models' parameters to increase the logits of the target class. However, these attacks require a local benign dataset to guide the search for the subset of parameters to be modified. Data-free backdoor [33] releases the requirement of benign data by collecting substitute data irrelevant to the main task and fine-tuning using the substitute data. DFBA [2] further proposes a retraining-free and data-free backdoor attack by injecting a backdoor path (a single neuron from each layer except the output layer) into the victim model. In summary, supply-chain attacks focus on increasing the backdoor's effectiveness without comprehensively considering parameter-space defenses.

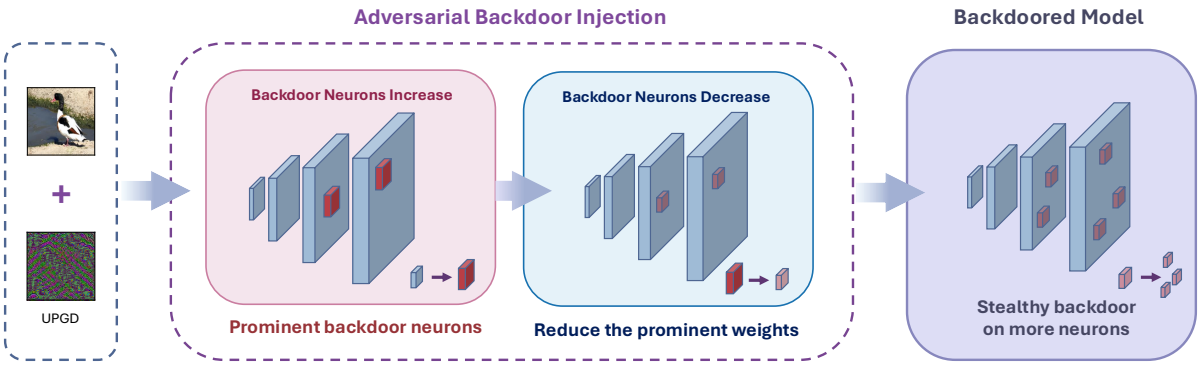

Figure 1: Diagram illustrating the working mechanism of Grond. On the left, universal PGD (UPGD) perturbation is generated as backdoor patterns to be injected. In the middle, ABI is applied where perturbed samples are iteratively used to train the model, and the model parameters are pruned to limit the magnitude of prominent backdoored weights. On the right, the output backdoored model that considers comprehensive stealthiness is deployed, where 1) the triggers are invisible, 2) the features of trigger samples are inseparable, and 3) the backdoored model weights are hardly distinguishable from benign model weights. Perturbations generated by UPGD are scaled up $10\times$ for visualization.

2.3 Backdoor Defenses

Backdoor defenses can be classified into detection and mitigation. Detection refers to determining whether a model is backdoored (*model detection*) [29, 56, 58, 64, 71] or a given input is applied with a trigger (*input detection*) [13, 15, 36]. Mitigation refers to erasing the backdoor effect from the victim model by pruning the backdoor-related neurons (*pruning-based* defenses) [24, 28, 62, 72] or unlearning the backdoor trigger (*fine-tuning-based* defenses) [35, 64, 68, 74]. In addition, recent works [23, 43, 60] also consider the home-field advantage[1] to design more powerful *proactive defenses*.

Backdoor detection. Backdoor trigger reverse engineering (also known as trigger inversion) is considered one of the most practical defenses for backdoor detection as it can be applied to both poisoning training and supply-chain scenarios [57, 58, 64, 66], i.e., it is a post-training method. Specifically, trigger inversion works by searching for a potential backdoor trigger for a specific model. The model is determined as backdoored if a trigger is found, and the trigger can be used to unlearn the backdoor. The searching is implemented as an optimization process corresponding to the model and a local benign dataset. For example, NC [56] firstly proposes trigger inversion for detection by optimizing the mask and pattern in the input space that can mislead the victim to the target class. This optimization is repeated for all classes. The model is considered backdoored if an outlier significantly smaller than the triggers for all other classes exists. Although methods similar to NC perform well against fixed patch trigger attacks, such as BadNets [14] and Blend [5], they may not be effective against input-stealthy attacks like WaNet [39]. To address this problem, FeatureRE [57] moves trigger inversion from input space to feature space. Unicorn [58] further proposes a transformation function for attacks in other spaces, such as numerical space [59]. Recent works [64, 66] focus on exploring new optimization objectives that address the inefficiency problem of previous trigger inversion methods due to optimization over all classes. BTI-DBF [64] trains a trigger generator by maximizing the backdoor feature difference between benign samples and their generated version (by the trigger generator) and minimizing the benign feature difference. BAN [66] optimizes the noise on neuron weights rather than input pixels to activate the potential backdoor, which further improves both effectiveness and efficiency.

Backdoor mitigation. Backdoor mitigation consists of fine-tuning and pruning, which are effective and do not assume knowledge of backdoor triggers. Pruning methods aim to find and remove backdoor-related neurons. FP [28] eliminates dormant neurons on benign inputs and then fine-tunes the pruned network. ANP [62] searches for backdoor-related neurons by adding adversarial noise to neuron weights to activate the backdoor. RNP [24] uses an unlearning and recovering process on benign data to expose backdoor neurons, as the recovering will force the backdoor neurons to be silent for the main benign task. Unlike these pruning methods guided by benign data, CLP [72] directly analyzes the Channel Lipschitzness Constant of the network and prunes the high Lipschitz constant channels in a data-free manner.

Traditional fine-tuning as a defense usually needs trigger inversion methods to recover the trigger and then unlearn the trigger. For example, BTI-DBF(U) [64] fine-tunes backdoor models using triggers recovered by their inversion algorithm. However, there is no guarantee that the recovered trigger is the true trigger for the backdoor. Recent works also consider fine-tuning without the trigger information but with prior human knowledge. For example, FT-SAM [74] observes a positive correlation between the weight norm of neurons and backdoor-related neurons. Then, they propose a fine-tuning method to revise the large outliers of weight norms using Sharpness-Aware Minimization (SAM). I-BAU [68] forms a min-max fine-tuning similar to adversarial training, where the inner maximizing searches for perturbations that mislead the model, and the outer minimizing is to keep the model's capability on benign data. FST [35] assumes the backdoor and benign features should be disentangled and actively shifting features while fine-tuning by

[1]The defender has full control of the system and could access the training process.

encouraging the discrepancy between the original backdoor model and the fine-tuned model.

Proactive defense. Several methods have been proposed to exploit the home-field advantage, i.e., a stronger defender, for better defensive performance. ABL [23] proposes two techniques to avoid learning the backdoor task while training on the poisoned data: 1) trapping the loss value of each example around a certain threshold because backdoor tasks are learned much faster than the main task, and their loss decreases much faster. The samples with lower loss are recorded as poisoned samples; 2) unlearning the backdoor with the recorded poisoned samples. CT [43] detects poisoned samples in the training set by introducing confusing batches of benign data with randomly modified labels. The confusing batches with random labeling corrupt the benign correlations between normal semantic features and semantic labels, so the inference model trained with confusing batches and the poisoned dataset will find it hard to distinguish benign samples. However, the correlation between the backdoor trigger and the target label remains intact, as the confusing batches contain no trigger. Therefore, samples with correctly predicted labels by the inference model are considered poisoned. PDB [60] proactively injects a defensive backdoor into the model during training, overriding the potential backdoor injected by the poisoned training data. In summary, proactive defenses assume a stronger defender for better defensive performance.

3 Comprehensive Backdoor Stealthiness

3.1 Threat Model

Attacker's goal. The attacker provides pre-trained models to users. The aim is to inject backdoors into the pre-trained model so that the model performs well on clean inputs but predicts the attacker-chosen target label when receiving inputs with a backdoor trigger, i.e., an all-to-one attack.

Attacker's knowledge. The attacker has white-box access to the training processes, the training data, and the model weights, i.e., the supply-chain threat model. During inference, the backdoor trigger is imperceptible to human inspectors.

Attacker's capabilities. The attacker can train a well-performed surrogate model to generate UPGD, which is used to perturb the victim model's input. Additionally, the attacker can alter the model's weights during training. Table 13 in [65, App. B] shows that the threat model of `Grond` is aligned with baseline attacks.

3.2 Lack of Parameter-Space Stealthiness

As introduced in the related work, early backdoor attacks that introduce noticeable changes in either input [5, 14] or feature space [38, 39] have been empirically shown powerful, even with very low poisoning rates [14, 69]. Focusing on the backdoor-introduced noticeable changes, backdoor defenses are improved to distinguish backdoor patterns in either input or feature space [27, 57]. Meanwhile, backdoor attacks are optimized to increase stealthiness in input [39] or feature space [41]. However, regardless of the implementation of input- or feature-space attack logic, backdoor behaviors are eventually embedded in the backdoored model's parameters. For this reason, it is important to investigate whether backdoor attacks introduce visible changes in the parameter space of the attacked models that can be used by the parameter-space defenses.

Considering this observation, we ran an initial experiment to understand the behavior of neurons in backdoored models. We use the TAC values [72] to quantify the relevance of a neuron to the backdoor behavior. The TAC values show the change in the output of each neuron (a feature channel of convolutional layers) before and after the trigger is attached to the input. Thus, TAC quantifies a neuron's sensitivity to the backdoor trigger. A high TAC value indicates that the neuron is strongly related to the backdoor behavior, whereas a low TAC value shows it is not. TAC takes the exact trigger information into account when measuring the backdoor effect, which makes TAC a straightforward and effective method that captures the relevant backdoor neurons' behaviour. Specifically, TAC is defined as:

$$\mathrm{TAC}_l^{(k)}(\mathcal{D}_c) = \frac{1}{|\mathcal{D}_c|} \sum_{\boldsymbol{x} \in \mathcal{D}_c} ||f_l^{(k)}(\boldsymbol{x}) - f_l^{(k)}(G_x(\boldsymbol{x}))||_2, \quad (2)$$

where $f_l^{(k)}$ is the k_{th} channel of the l_{th} layer. $G_x(\boldsymbol{x})$ is the poisoned sample. $\mathcal{D}_c$ consists of a few benign samples. Note that TAC can only be used to analyze backdoor behaviors and cannot be deployed as a practical defense, as it requires access to backdoor triggers, which is unrealistic in practice.

TAC analysis of different backdoor attacks is shown in Figure 2, where each dot represents the TAC value for one of the 512 individual neurons. We can observe that the TAC values of neurons of backdoored models are substantially higher than those of benign models. In particular, neurons with higher TAC values contribute more to the backdoor behavior. The working mechanism of pruning- and fine-tuning-based backdoor defenses can be understood as targeting and eliminating neurons with TAC values that are substantially higher than those of others. Our observations from the TAC analysis suggest that backdoor attacks are designed to be stealthy in input space, and feature space can, in fact, be identified in parameter space, making them susceptible to parameter-space defenses. Our experimental analysis further substantiates this assumption (see Section 4). Thus, we conclude that current backdoor attacks may not be robust against parameter-space defenses.

3.3 `Grond` for Comprehensive Stealthiness

To address the vulnerabilities identified in the parameter space, we propose a stealthy backdoor attack, `Grond`, that considers comprehensive stealthiness, i.e., stealthiness in input, feature, and parameter space. `Grond` includes two key parts: UPGD trigger generation and Adversarial Backdoor Injection (ABI).

Backdoor trigger generation for input-space stealthiness. We use imperceptible adversarial perturbations to generate *imperceptible* backdoor triggers inspired by adversarial example studies [37, 70]. We modify the original PGD algorithm to generate a universal PGD (UPGD) perturbation as the backdoor trigger. UPGD contains non-robust but generalizable semantic information [53], which correlates with the benign functions of the victim model and shortens the distance between poisoned data and the target classification region [70]. Consequently, backdoor patterns tend to make fewer prominent changes to the victim network.

Similar to [69, 70], UPGD is generated on a well-trained surrogate model trained on the clean training set. The architecture and parameters of the surrogate model do not necessarily need to be the

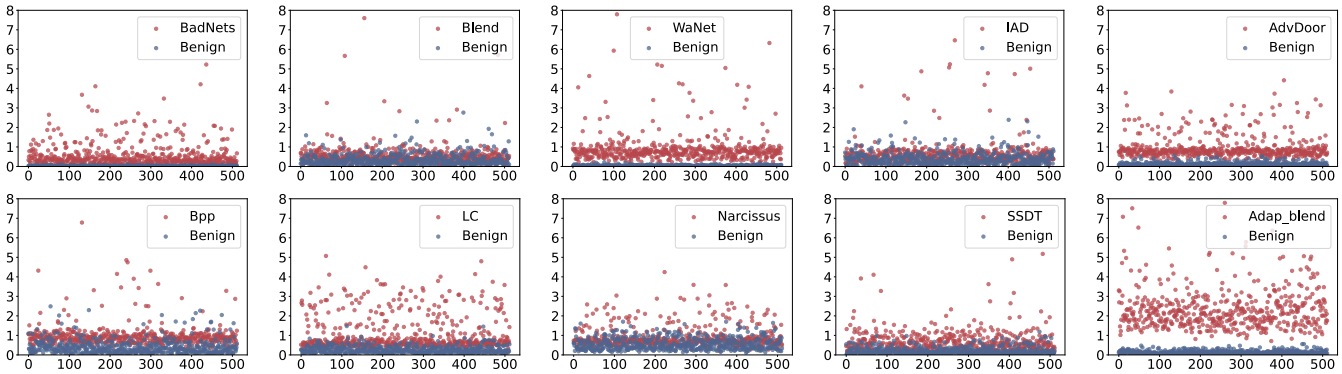

Figure 2: TAC [72] analysis of different backdoor attacks on 512 neurons. The y axis contains the TAC values, and the x axis depicts the index of neurons. Higher TAC values suggest a stronger relation between corresponding neurons and the backdoor trigger. In the Appendix [65, App. C.7], we also provide sorted TAC value plots, clearly showing the prominent TAC values.

Algorithm 1 UPGD Generation Algorithm

Input: Surrogate model $f_{\theta_{sur}}$, training data $\mathcal{D}$, perturbation budget ϵ, the number of iteration $\mathcal{I}$, the target class t.
Output: UPGD $\boldsymbol{\delta}$

$S = B(\boldsymbol{\delta}; \epsilon) = \{\boldsymbol{\delta} \in \mathbb{R}^{d_c \times d_h \times d_w} : ||\boldsymbol{\delta}||_\infty \leq \epsilon\}$
$\boldsymbol{\delta} \leftarrow$ `random_initialization` $\wedge \boldsymbol{\delta} \in S$
for $i \in (0, \mathcal{I} - 1)$ **do**
  $\boldsymbol{x} \leftarrow$ `sample_batch(`$\mathcal{D}$`)`
  $\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{(\boldsymbol{x},y)\in\mathcal{D}} \ell(f_{\theta_{sur}}(\boldsymbol{x} + \boldsymbol{\delta}; \boldsymbol{\theta}), t)$,
  $\boldsymbol{\delta} \leftarrow \min_{\boldsymbol{\delta} \in S} \mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta})$
end for

same as the victim model (see Table 15 in [65, App. C.2]). UPGD is optimized following the PGD [34] algorithm to decrease the surrogate model's cross-entropy loss that takes as inputs the adversarial examples (the poisoned samples in our case) and the target class label. This procedure is described formally in Algorithm 1. The $\boldsymbol{\delta}$ is the generated UPGD that will be used as a backdoor trigger; thus, $G_x(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{\delta}$. S is the ball function with the radius ϵ, and the small ϵ guarantees the imperceptibility of the backdoor trigger as it controls the perturbation's magnitude.

The backdoor is injected during training by poisoning some training data from the target class, i.e., applying the UPGD trigger to the training data. In the inference stage, our backdoor is activated by the same trigger. The motivation for our small-size trigger ($\epsilon = 8$) is imperceptibility.

Adversarial Backdoor Injection for parameter-space stealthiness. Backdoor neurons (i.e., trigger-related neurons) regularly show higher activation values for inputs that contain the trigger, which results in powerful performance [27, 29, 57]. To this end, backdoor training needs to substantially increase the magnitude of parameters of backdoor neurons [24, 62, 72], which harms the parameter-space stealthiness of backdoor attacks.

One way to find the sensitive neurons with higher activation values is to analyze the Lipschitz continuity of the network. Leveraging this fact, we introduce a novel backdoor training mechanism, *Adversarial Backdoor Injection*, to increase the parameter-space backdoor stealthiness. Specifically, each neuron's Upper bound of Channel Lipschitz Condition (UCLC [72]) is calculated, based on which the weights of these suspicious neurons are set to the mean of all neurons' weights in the corresponding layer after every training epoch. In our implementation, we use the weights before every batch normalization as the neuron weights corresponding to the channel setting in UCLC. We prune neurons by substituting their weights with the mean ones because pruning to zeros makes the training unable to converge in our experiments. Formally, the k_{th} parameter of the l_{th} layer, $\boldsymbol{\theta}_l^{(k)}$, is updated as follows:

$$\boldsymbol{\theta}_l^{(k)} := \begin{cases} \text{mean}(\boldsymbol{\theta}_l), & \sigma(\boldsymbol{\theta}_l^{(k)}) > \text{mean}(\sigma(\boldsymbol{\theta}_l)) + u \times \text{std}(\sigma(\boldsymbol{\theta}_l)) \\ \boldsymbol{\theta}_l^{(k)}, & \text{otherwise,} \end{cases} \tag{3}$$

where u is a fixed threshold and σ is the UCLC value of the given weights. The measure for quantifying backdoor relevance can be changed from UCLC to others, such as the distance of neuron outputs when receiving benign and backdoor inputs, where a larger distance means the neuron is more relevant to backdoor behaviors and can be pruned. We use the modified UCLC for training efficiency, as UCLC is data-free, which does not require calculation based on the outputs of neurons.

In adversarial training [34], adversarial examples are introduced during training to increase the model's robustness during inference. Similarly, during the Adversarial Backdoor Injection, we use backdoor defenses to increase the resistance of backdoor attacks to parameter-space defenses. At the end of each training epoch, Adversarial Backdoor Injection prunes the trained model to decrease the weights of backdoor neurons. Iteratively, backdoored neurons spread across the whole model instead of forming a few prominent backdoor neurons, as illustrated in Figure 1.

Feature-space stealthiness. We hypothesize that feature-space stealthiness is a by-product of parameter-space and input-space stealthiness since the variation of feature maps is strongly correlated with model parameters and inputs. Figures 7 and 9 show that `Grond` can substantially increase the feature-space stealthiness.

Table 2: Pruning-based mitigations against backdoored ResNet18 on CIFAR10. BA refers to benign accuracy on clean data, ASR to attack success rate, and PR to the poisoning rate of the training set. The average drop of BA and ASR is also shown with downward arrows compared to the performance without any defense. Red marks indicate the attack failed to resist the defense with an ASR lower than 60%, and green means that the ASR is higher than 60%.

Attack	No Defense		FP [28]		ANP [62]		CLP [72]		RNP [24]		Average	
	BA	ASR	BA	ASR	BA	ASR	BA	ASR	BA	ASR	BA	ASR
BadNets [14]	93.13	100	92.42	71.71	91.60	1.06	88.99	49.02	84.04	13.82	89.26 ↓3.87	33.90 ↓66.10
Blend [5]	94.42	100	93.08	99.99	93.57	0.33	90.3	0.54	94.63	57.98	92.89 ↓1.53	39.71 ↓60.29
WaNet [39]	93.60	99.37	92.96	4.60	91.08	0.49	91.53	2.12	92.86	3.17	92.11 ↓1.49	2.59 ↓96.78
IAD [38]	92.88	97.10	91.96	1.22	92.84	0.71	92.24	0.74	92.72	0.42	92.44 ↓0.44	0.77 ↓96.33
AdvDoor [70]	93.97	100	93.37	98.69	91.46	28.83	89.22	6.13	90.17	44.60	91.05 ↓2.92	44.56 ↓55.44
Bpp [59]	94.19	99.93	93.38	18.89	92.96	2.97	93.37	1.89	92.2	5.79	92.98 ↓1.21	7.39 ↓92.54
LC [54]	94.31	100	92.22	93.57	91.02	24.43	90.96	0.38	82.70	33.60	89.23 ↓5.08	37.99 ↓62.01
Narcissus [69]	93.58	99.64	93.49	96.54	89.76	49.18	93.19	97.82	91.10	94.59	91.88 ↓1.70	84.53 ↓15.11
Adap-blend [41]	92.74	99.67	92.06	95.50	86.48	67.73	92.49	99.62	78.63	1.56	87.42 ↓5.32	66.10 ↓33.57
SSDT [36]	93.70	90.30	93.41	0.80	93.88	0.60	93.66	1.20	93.99	3.30	93.74 ↑0.04	1.47 ↓88.83
DFST [7]	95.23	100	94.79	93.84	94.64	3.72	92.43	3.53	93.29	12.68	93.79 ↓1.44	28.44 ↓71.56
DFBA [2]	88.99	100	86.85	0.03	88.96	9.55	88.96	9.57	88.96	0.90	88.43 ↓0.56	5.01 ↓94.99
Grond (PR=5%)	93.43	98.04	93.09	99.73	91.43	94.01	93.29	87.89	91.83	85.22	92.41 ↓1.02	91.71 ↓6.33
Grond (PR=1%)	94.26	93.51	93.31	96.32	92.94	91.48	94.33	87.56	92.13	94.87	93.18 ↓1.08	92.56 ↓0.95
Grond (PR=0.5%)	94.36	92.91	93.32	90.96	93.87	84.04	94.52	86.82	91.99	84.63	93.43 ↓0.93	86.61 ↓6.30

Table 3: Fine-tuning-based mitigations against backdoored ResNet18 on CIFAR10.

Attack	vanilla FT		FT-SAM [74]		I-BAU [68]		FST [35]		BTI-DBF(U) [64]		Average	
	BA	ASR	BA	ASR	BA	ASR	BA	ASR	BA	ASR	BA	ASR
BadNets [14]	91.07	43.96	92.01	2.84	92.60	76.02	92.40	13.10	91.26	13.12	91.87 ↓1.26	29.81 ↓70.19
Blend [5]	91.64	99.61	92.52	1.73	91.84	8.84	93.40	100	91.86	100	92.25 ↓2.17	62.04 ↓37.96
WaNet [39]	91.11	0.99	90.89	1.03	87.98	0.81	92.17	0.04	90.30	4.89	90.49 ↓3.11	1.55 ↓97.82
IAD [38]	90.83	2.16	92.18	2.87	88.4	15.68	91.29	0.00	89.54	1.59	90.45 ↓2.43	4.46 ↓92.64
AdvDoor [70]	91.25	68.68	92.18	1.23	89.29	16.99	91.06	99.99	90.25	100	90.81 ↓3.16	57.38 ↓42.62
Bpp [59]	91.36	3.40	91.38	1.00	92.06	6.46	93.23	26.83	90.61	2.73	91.73 ↓2.46	8.08 ↓91.85
LC [54]	90.26	88.52	91.46	1.91	85.87	5.11	91.80	13.11	90.71	4.37	90.02 ↓4.29	22.60 ↓77.40
Narcissus [69]	91.70	92.91	91.76	23.98	91.48	51.74	90.06	54.22	90.94	98.11	91.19 ↓2.39	64.19 ↓35.45
Adap-blend [41]	92.42	98.73	91.23	22.4	85.38	37.31	90.91	1.19	89.17	7.09	89.82 ↓2.92	33.34 ↓66.33
SSDT [36]	93.74	0.70	93.15	0.60	90.27	3.10	92.85	0.20	90.79	1.40	92.16 ↓1.54	1.20 ↓89.10
DFST [7]	95.01	2.07	94.70	0.00	89.75	19.11	93.06	2.66	90.41	22.34	92.59 ↓2.64	9.24 ↓90.76
DFBA [2]	86.68	10.10	86.03	5.24	85.48	100	82.76	57.62	84.49	100	85.09 ↓3.90	54.59 ↓45.41
Grond (PR=5%)	91.75	94.28	92.02	80.07	90.39	93.92	93.27	99.92	91.88	99.00	91.86 ↓1.57	93.44 ↓4.60
Grond (PR=1%)	91.41	85.52	92.83	79.17	87.89	91.34	93.21	96.59	90.66	88.69	91.20 ↓3.06	88.26 ↓5.25
Grond (PR=0.5%)	91.42	82.96	92.34	76.92	89.83	79.68	93.44	92.71	90.39	91.83	91.48 ↓2.88	84.82 ↓8.09

4 Experimental Evaluation

4.1 Experimental Setup

Datasets and Architectures. We follow the common settings in existing backdoor attacks and defenses and conduct experiments on CIFAR10 [22], GTSRB [50], and a subset of ImageNet [8] with 200 classes and 1,300 images per class (ImageNet200). More details about the datasets can be found in [65, App. A.1]. The primary evaluation is performed using ResNet18 [17]. Moreover, we evaluate Grond using four additional architectures, VGG16 [49], DenseNet121 [20], EfficientNet-B0 [51], and one recent architecture InceptionNeXt [67] (see Table 15 in [65, App. C.2]). We also evaluate Grond with large and transformer-based models (see Table 16 in [65, App. C.3]).

Attack Baselines. Grond is compared with 12 representative attacks: BadNets [14], Blend [5], WaNet [39], IAD [38], AdvDoor [70], BppAttack [59], LC [54], Narcissus [69], Adap-Blend [41], SSDT [36], DFST [7], and DFBA [2]. The default poisoning rate is set at 5% (of the training set) for all attacks following previous work [64, 66]. Additionally, Grond is evaluated under various poisoning rates to provide a thorough analysis of its effectiveness. Following related works, the training schedule for attacks is 200 epochs when using

Table 4: Backdoor performance of Grond and baseline attacks on ImageNet200 and GTSRB.

Datasets	Attack	No Defense		FT-SAM [74]		I-BAU [68]		CLP [72]		Average	
		BA	ASR	BA	ASR	BA	ASR	BA	ASR	BA	ASR
ImageNet200	BadNets [14]	80.65	91.03	79.89	2.21	70.28	26.06	70.74	64.86	73.64 ↓7.01	31.04 ↓59.99
	Blend [5]	80.70	95.63	80.19	0.39	76.13	30.81	80.02	23.38	78.78 ↓1.92	18.19 ↓77.44
	WaNet [39]	81.24	99.97	80.41	0.66	75.67	47.27	77.18	99.78	77.75 ↓3.49	49.24 ↓50.73
	IAD [38]	79.74	99.98	75.49	0.68	77.44	15.18	76.97	84.49	76.63 ↓3.11	33.45 ↓66.53
	AdvDoor [70]	80.72	100	79.52	98.90	74.03	61.31	77.90	100	77.15 ↓3.57	86.74 ↓13.26
	Bpp [59]	81.36	92.74	79.37	1.05	76.53	3.21	80.10	2.34	78.67 ↓2.69	2.19 ↓90.55
	Narcissus [69]	81.73	81.28	80.00	83.37	77.03	56.19	80.99	86.37	79.34 ↓2.39	75.31 ↓5.97
	SSDT [36]	75.45	100	78.19	76.00	76.26	22.00	76.02	94.00	76.82 ↑1.37	64.00 ↓36.00
	Grond	80.92	94.11	79.05	95.05	76.89	87.75	80.29	93.83	78.74 ↓2.18	92.21 ↓1.9
GTSRB	BadNets [14]	97.19	100	95.57	0.48	92.02	29.22	96.38	0.47	94.66 ↓2.53	10.06 ↓89.94
	Blend [5]	95.92	100	93.36	0.21	92.64	38.27	93.21	0.00	93.07 ↓2.85	12.83 ↓87.17
	WaNet [39]	98.69	99.77	92.18	0.45	91.25	0.00	90.14	18.14	91.19 ↓7.50	6.19 ↓93.58
	IAD [38]	99.08	99.65	92.72	0.10	90.11	0.35	98.08	14.63	93.64 ↓5.44	5.03 ↓94.62
	AdvDoor [70]	95.80	99.99	93.94	32.26	92.67	38.20	90.09	66.39	92.23 ↓3.57	45.62 ↓54.37
	Bpp [59]	98.69	99.93	91.27	0.00	92.61	0.23	97.16	2.29	93.68 ↓5.01	0.84 ↓99.09
	Narcissus [69]	95.60	97.18	93.61	54.55	92.87	80.74	93.99	97.60	93.49 ↓2.11	77.63 ↓19.55
	SSDT [36]	96.02	77.78	93.11	0.00	90.82	0.00	94.65	19.31	92.86 ↓3.16	6.44 ↓71.34
	Grond	95.83	95.36	93.80	71.84	93.13	94.30	91.28	93.19	92.74 ↓3.09	86.44 ↓8.92

CIFAR10 and GTSRB, and 100 epochs for ImageNet200. We use 1,000 images as the validation set to select the best-performing checkpoint. More implementation details are provided in [65, App. A.2].

Defense Baselines. We evaluate Grond and baseline attacks with 17 defenses, including **four pruning-based** methods (FP [28], ANP [62], CLP [72], and RNP [24]), **five fine-tuning-based** methods (vanilla FT, FT-SAM [74], I-BAU [68], FST [35], and BTI-DBF (U) [64]), **five backdoor model detections** (NC [56], Tabor [16], FeatureRE [57], Unicorn [58], and BTI-DBF [64]), **two backdoor input detections** (Scale-up [15] and IBD-PSC [19]), and a **proactive defense** CT [43]. Following their default settings, BTI-DBF [64] and FP [28] use 5% of training data, and other defenses use 1% of training data for detection or mitigation. CLP is a data-free backdoor pruning tool that uses no clean data. CT has access to the complete training set without knowing which samples are poisoned and can also interact with the model during training. Backdoor defense details and hyperparameters can be found in [65, App. A.3].

4.2 Main Results on Backdoor Mitigation

All evaluated backdoor attacks are ineffective against at least one parameter-space backdoor defense on the CIFAR10, as demonstrated in Tables 2 and 3. It suggests that common backdoor attacks designed to be stealthy in input and feature spaces are vulnerable to parameter-space defenses. Given that all backdoor behaviors are embedded in parameters of backdoored models, this finding suggests that future backdoor attacks should consider parameter-space defenses as a standard step to evaluate comprehensive stealthiness.

Not surprisingly, Grond performs better than all baseline attacks when considering evaluated backdoor defenses since Grond is designed to consider comprehensive stealthiness. On four pruning-based mitigations, Grond achieves 7.18% higher ASR on average than the best backdoor attack, Narcissus. On five fine-tuning mitigations that show more powerful defense capability than pruning-based mitigations, Grond achieves 29.25% higher ASR on average than Narcissus. In addition, Grond bypasses the five model detection and two input-space detections (see Section 4.6).

Pruning-based mitigation. We take a closer look at the details of pruning-based backdoor mitigation experiments in Table 2, presenting the results of all attacks against four pruning-based defenses. BadNets and Blend perform better on average than input-space stealthy attacks, e.g., WaNet and Bpp, because input-space stealthy attacks introduce significant separability in the feature space (see Figures 7 and 9). Across all pruning-based defenses, FP performs the worst, as expected, since it follows regular model pruning practice and is not a tailored backdoor pruning method.

Fine-tuning-based mitigations. Table 3 presents the backdoor performance against five fine-tuning-based defenses. In general, fine-tuning-based defenses are more effective than pruning-based defenses. For example, Narcissus and Adap-Blend can achieve ASRs higher than 60% against three out of four pruning-based defenses but are much less effective against most fine-tuning-based methods. FT-SAM is the most effective across all defenses, as shown in Tables 2 and 3, being able to compromise the effectiveness of all attack baselines. One important reason is that FT-SAM adopts Sharpness-Aware Minimization [12] to adjust the outlier of weight norm (large norms) to remove the potential backdoor. Larger weights of neurons are introduced by existing attacks to guarantee a high ASR [29], which also causes large differences when receiving benign and backdoor inputs (see Figure 4). Grond can bypass FT-SAM, as expected, since it deliberately decreases the weights of backdoor neurons, compromising the core working mechanism of FT-SAM.

Comparison with Supply-Chain Attacks. Sharing a similar threat model to supply-chain attacks, we compare Grond and three

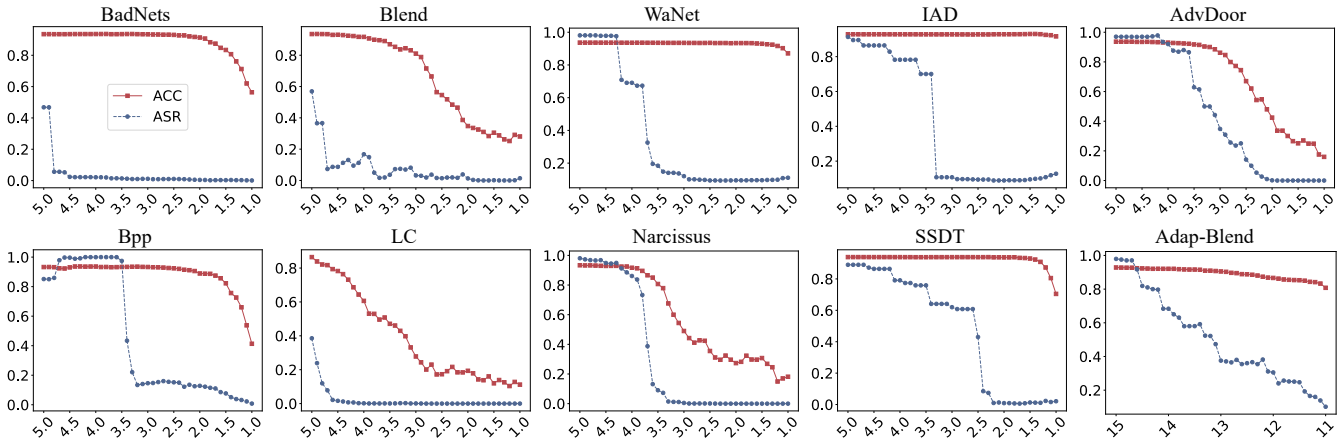

Figure 3: Pruning neurons with high TAC values using different thresholds (the x axis).

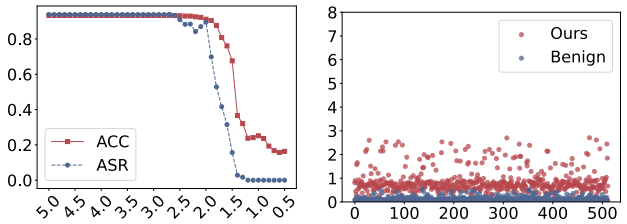

Figure 4: The left is the performance of pruning neurons with high TAC values using different thresholds for Grond. The right is the TAC analysis of Grond on 512 neurons.

state-of-the-art supply-chain attacks (SSDT [36], DFST [7], and DFBA [2]), where these attacks are also designed to be robust against backdoor defenses. In particular, DFST [7] proposes to include a controlled detoxification technique in the training process, which restrains the model from picking up simple features. DFBA [2] directly modifies a few parameters of a classifier to inject a backdoor. SSDT [36] introduces additional terms in the loss for the Source-Specific and Dynamic-Triggers (i.e., SSDT) attack, which obscures the difference between normal samples and malicious samples. Tables 2 and 3 also include the performance of supply-chain attacks against pruning- and fine-tuning-based defenses. It is clear that existing backdoor defenses can defeat supply-chain attacks. The ASR of DFST [7], DFBA [2], and SSDT [36] are decreased to less than 10% while the BA drop is less than 3%.

On ImageNet200 and GTSRB. Real-world classification tasks may involve more categories, such as GTSRB (43 classes) and ImageNet200 (200 classes), and the percentage of each class in the dataset will commonly be much less than 10%. We target InceptionNext-Small on Imagenet200 and ResNet18 on GTSRB. The l_∞ norm perturbation budget of UPGD is $\epsilon = 16$ for GTSRB and $\epsilon = 8$ for ImageNet200 to achieve imperceptible perturbations. Table 4 demonstrates that Grond is still effective on datasets with more classes and higher resolutions, especially against the most powerful parameter-space defense, FT-SAM.

4.3 Adaptive Defenses

As Grond includes UPGD trigger and Adversarial Backdoor Injection (ABI), we consider two adaptive defenses targeting those two components.

TAC pruning targeting the UPGD trigger. First, we use the UPGD trigger information to build a new pruning method based on the TAC values. The TAC values are calculated using the backdoor trigger, making the TAC highly adaptive to evaluating any backdoor attack. In particular, we prune neurons with high TAC values in the backdoored model, i.e., removing the neurons more sensitive to the UPGD trigger. Figure 4 shows the pruning results of Grond. The left figure provides the pruning results. The right figure contains the TAC values plots of neurons in the 4_{th} layer (the layer before the classification head) of ResNet18. We show that pruning neurons with high TAC values decreases benign accuracy, which means the backdoor neurons are not easily distinguishable from benign neurons without harming benign performance. The analysis supports our statement that Grond spreads the backdoor to more neurons instead of a few prominent ones. In [65, App. C.7], we provide sorted TAC value plots in Figure 11, showing that prominent neurons with high TAC values are rather limited in Grond.

Neuron noise targeting ABI. Second, we consider adding noise to neurons of Grond models as an adaptive defense, as the ABI component involves operations on neuron weights. Specifically, we add noise to the weights and biases of batch normalization layers. The range of noise is limited by $[-\epsilon_{noise}, \epsilon_{noise}]$. Table 5 provides the performance of Grond and the benign model under different noise levels. With increased noise, the benign accuracy of both benign and Grond models is decreased, but the ASR of Grond remains high. This result also supports the stealthiness of Grond in the parameter space.

4.4 Backdoor Analysis

This section analyzes why baseline attacks are ineffective against parameter space backdoor defenses, and why Grond performs better, according to the TAC values and the weights' change after applying backdoor defenses.

Table 5: Adaptive defense against Grond using noise on neurons.

Method	Benign		Grond	
	BA	ASR	BA	ASR
$\epsilon_{noise} = 0.0$	94.76	-	94.16	98.04
$\epsilon_{noise} = 0.1$	94.54	-	93.88	97.99
$\epsilon_{noise} = 0.2$	93.82	-	93.18	96.22
$\epsilon_{noise} = 0.3$	91.21	-	90.05	99.08
$\epsilon_{noise} = 0.4$	85.65	-	88.33	99.42
$\epsilon_{noise} = 0.5$	82.43	-	84.30	99.74

Pruning neurons with prominent TAC values for baseline attacks. Section 3.2 demonstrates the existence of prominent neurons with high TAC values in backdoor models. Note that TAC represents the strongest type of backdoor defense, where the exact trigger information is exploited. In this section, we show that pruning prominent neurons could mitigate all baseline attacks. Specifically, we assign zero to the neuron's weight if its TAC value exceeds a certain threshold. Figure 3 shows the pruning performance using different thresholds for ten baseline attacks. We can observe that for all ten baseline attacks, one threshold can always be found where the ACC is high and the ASR is low. It indicates that the backdoor effect of the ten attacks can be erased while maintaining good performance on benign samples if these prominent neurons are pruned. Thus, we confirm the existence of prominent neurons, which correspond to the backdoor effect. More importantly, these backdoor neurons can be disentangled from benign neurons, so these baseline attacks are not stealthy concerning neuron weights, i.e., not stealthy in the parameter space.

Grond's weights are more difficult to be modified by backdoor defense. This section analyzes the changes while applying pruning- and fine-tuning-based defenses to baseline attacks and Grond. Our goal is to demonstrate that the baseline attacks can be significantly affected by defenses, but Grond can resist these defenses. In Figure 5, we record the changes in the weights of 512 neurons (in layer 4 of ResNet18) after applying two types of defenses, FT-SAM [74] and CLP [72], as they both focus on the backdoor-related neurons. In contrast to BadNets, the weight changes in Grond model after FT-SAM fine-tuning are smaller. In the sorted changes, it is clear that there are a few neurons for BadNets that correspond to significant changes, which is not the case for Grond. In the second row of Figure 5, the CLP can find a few neurons relevant to backdoor but cannot find these for Grond. In addition, pruning-based methods (as shown by CLP results in Figure 5) only improve a few neurons, while fine-tuning methods can update all neurons. We conjecture this is why fine-tuning-based defenses perform better than pruning-based defenses in Tables 2 and 3. More results with other baseline attacks are provided in Figure 12 in [65, App. C.9].

4.5 ABI Improves Common Backdoor Attacks

In this section, we show that our Adversarial Backdoor Injection (ABI) strategy generalizes to all evaluated common backdoor attacks. We combine the ABI module with baseline attacks to improve their resistance against parameter-space defenses. Figure 6

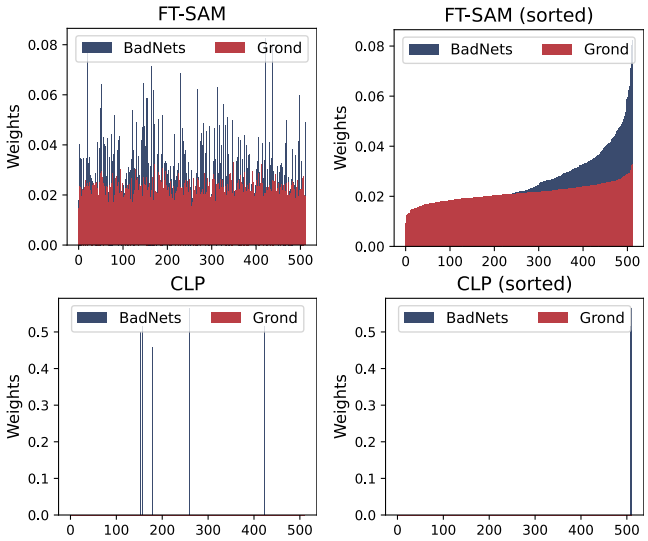

Figure 5: The weight changes after backdoor defenses. More results with other attacks in Figure 12 in [65, App. C.9].

Table 6: Backdoor detection performance on CIFAR10. 20 ResNet18 models are trained at each poisoning rate. Bd. refers to the number of models determined as backdoor models. Acc. refers to the detection accuracy.

Defense	PR=5%		PR=1%		PR=0.5%	
	Bd.	Acc.	Bd.	Acc.	Bd.	Acc.
NC [56]	5	25%	2	10%	1	5%
Tabor [16]	5	25%	2	10%	0	0%
FeatureRE [57]	0	0%	0	0%	0	0%
Unicorn [58]	0	0%	0	0%	0	0%
BTI-DBF [64]	3	15%	5	25%	3	15%

demonstrates that ABI is effective for all attacks when evaluating against the parameter-space defense ANP, where ASRs increase after adversarial injection, especially for BadNets, Blend, AdvDoor, Narcissus, and Adap-Blend. The improvement for feature space attacks (WaNet, IAD, and Bpp) is incremental. We speculate that feature space attacks rely too much on prominent features, as their modification in the input space is minor. To activate the backdoor with such minor input modifications, the prominent features are required in the feature space. In addition, Figure 10 in [65, App. C.6] shows the results of ABI without defense, demonstrating that it does not harm in general the BA and ASR when no defense is applied. Following our findings, we suggest that future backdoor attacks can use ABI to increase parameter-space stealthiness.

4.6 Backdoor Detection

Following previous works [64, 66], we choose five representative backdoor model detections for evaluation. We use 20 models for each poisoning rate with different random seeds. Then, we report the number of models detected as backdoor models out of the 20.

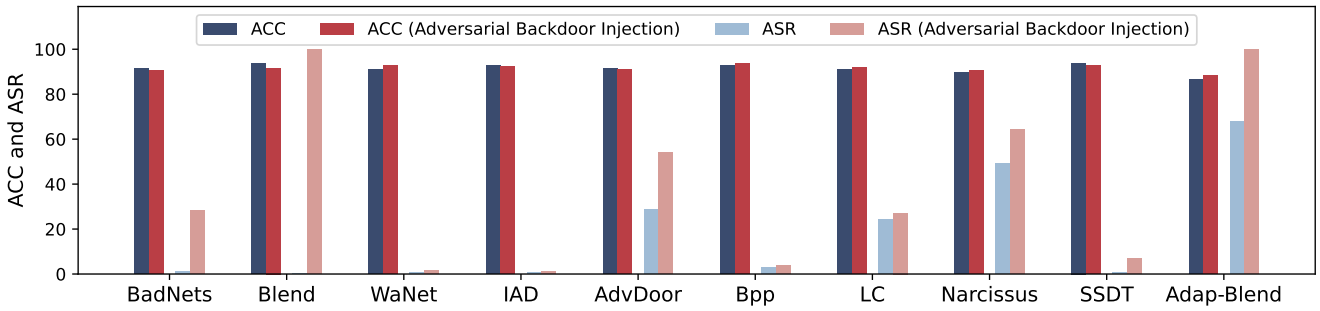

Figure 6: BA and ASR of backdoor attacks before and after ABI against parameter-space defense ANP.

Table 7: Comparison with different strategies for the generation of backdoor triggers.

Strategy	No Defense		CLP [72]		FT-SAM [74]	
	BA	ASR	BA	ASR	BA	ASR
Random noise	94.24	1.28	94.13	0.97	93.90	1.84
PGD	94.77	69.33	92.57	46.63	92.40	24.56
UPGD	93.43	98.04	93.29	87.89	92.02	80.07

Table 8: The semantic trigger (an automobile image) with different strategies for `Grond`. The "Target Same" refers to the target class being the same as the semantic trigger, i.e., automobile. The "Target Diff." refers to the target class being different (i.e., airplane) from the semantic trigger.

Strategies for		No Defense		CLP [72]		FT-SAM [74]	
		BA	ASR	BA	ASR	BA	ASR
Clean-Label	Target Same	94.20	74.64	93.97	70.66	91.68	9.38
	Target Diff.	93.82	94.42	93.75	93.50	90.92	31.37
Dirty-Label	Target Same	93.77	100	93.88	100	91.40	12.17
	Target Diff.	94.25	100	94.14	100	91.34	7.39

Table 6 shows that all detections fall short when detecting `Grond`. In particular, NC [56], Tabor [56], and BTI-DBF [64] can detect a small part of backdoored models, while FeatureRE [57] and Unicorn cannot detect any of them. For featureRE [57], we conjecture it is over-dependent on the separability in the feature space, but `Grond` does not rely on prominent backdoor features according to Figure 9 in the Appendix [65, App. C.4]. For Unicorn [58], the false positive rate is high, and it tends to report every class as the backdoor target, even on models trained with benign data only. Except for model detection, `Grond` can also bypass input-space detections as demonstrated in [65, App. C.1].

4.7 Ablation Study

The ablation study is designed for the two components of `Grond`, the UPGD trigger and Adversarial Backdoor Injection. In addition, we also evaluate the dirty-label setting of `Grond` and show the difference compared to using clean-label in [65, App. C.5].

Table 9: Ablation study for `Grond`.

Arch	Method	No Defense		CLP [72]		FT-SAM [74]	
		BA	ASR	BA	ASR	BA	ASR
ResNet18	UPGD	93.86	98.61	91.15	3.97	91.80	51.77
	+ABI	93.43	98.04	93.29	87.89	92.02	80.07
InceptionNeXt	UPGD	87.81	96.81	87.72	96.57	87.06	2.37
	+ABI	87.06	96.86	86.93	96.87	86.50	92.02

Trigger generation. To explore the influence of trigger patterns, we employ and evaluate three types of triggers: random noise, PGD perturbation, and UPGD perturbation, using ResNet18 on CIFAR10. The random noise is sampled from a uniform distribution, and the PGD employs a projected gradient descent to generate sample-wise perturbations [34]. The generation of UPGD is described in Algorithm 1. All three triggers are limited to 8/255 (l_∞ norm) for imperceptibility and use the same training settings described in Table 11 in [65, App. A.2].

Table 7 shows that random noise is ineffective as a backdoor trigger due to low ASR, even if no defense is applied. The sample-wise PGD perturbation is more effective than random noise and shows (limited) robustness against CLP and FT-SAM. UPGD generates the most effective backdoor trigger with an ASR higher than 80% after CLP and FT-SAM, and we speculate that the reason is that UPGD exploits features from the target class, similar to Narcissus [69].

Concerning exploiting features from the target class, we also explore using the natural image as the trigger, which directly contains the semantic information. Table 8 shows the performance when using an automobile image from the CIFAR10 dataset as a semantic trigger. Specifically, inspired by naturally occurring backdoor [21], we design the semantic trigger by resizing the automobile image to 8×8 and sticking it on a part of (PR=5%) the training images. We consider four types of strategies, including only poisoning the same class as the trigger in clean-label, poisoning a different class (class airplane) in clean-label, only poisoning the same class as the trigger in dirty-label, and poisoning a different class (class airplane) in dirty-label. In Table 8, the semantic trigger can be effective as a backdoor trigger and resist against the CLP pruning. However, the semantic trigger is not robust against the FT-SAM fine-tuning. The reason is that the semantic trigger cannot effectively represent the feature of

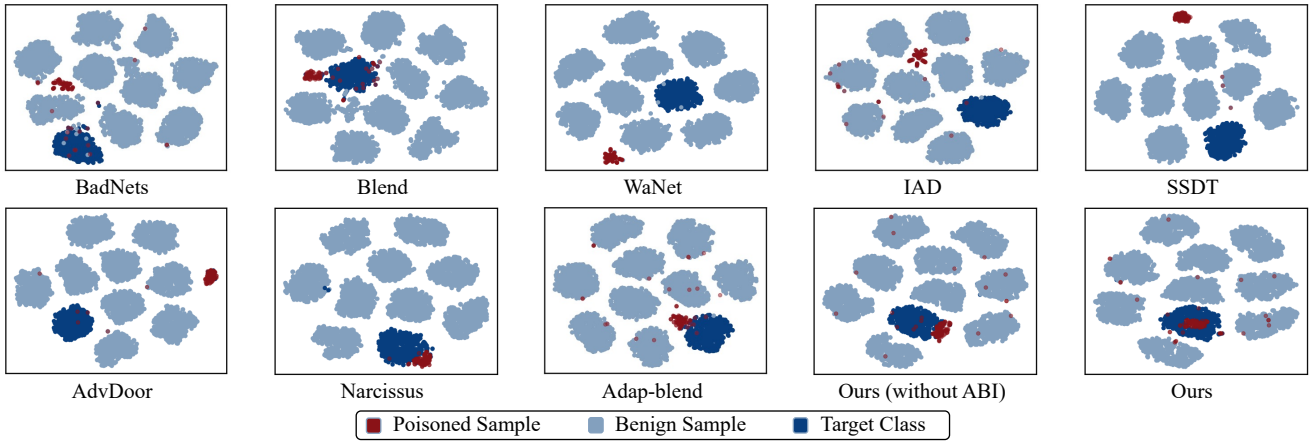

Figure 7: Examples of feature visualization of Grond and baseline attacks.

Table 10: Evaluation with the proactive defense, CT [43], under different poisoning rates (PR).

Attack	PR	ACC	ASR	Recall	FPR
BadNets [14]	5%	93.18	99.96	2500/2500	1568/47500
	2.5%	93.35	99.83	1250/1250	518/48750
	1%	93.30	100	500/500	73/49500
	0.5%	93.43	100	250/250	5/49750
	0.3%	93.63	99.94	150/150	222/49850
Adap-patch [41]	5%	93.28	100	1808/2500	116/47500
	2.5%	93.68	100	1088/1250	20/48750
	1%	93.73	100	494/500	570/49500
	0.5%	93.31	100	160/250	154/49750
	0.3%	93.26	100	86/150	3825/49850
Grond	5%	93.84	99.41	2499/2500	671/47500
	2.5%	93.81	95.83	115/1250	7220/48750
	1%	94.09	92.48	208/500	6690/49500
	0.5%	94.36	92.91	90/250	6738/49750
	0.3%	94.22	90.10	29/150	6349/49850

a class. Conversely, adversarial perturbation (PGD, UPGD) acquires more representative and discriminative class features, since they can capture the correlations among different regions of the decision boundary and easily fool most of the inputs [37].

Adversarial backdoor injection is critical. There are two components in Grond: the UPGD trigger generation and Adversarial Backdoor Injection. We conduct an ablation study with two architectures on CIFAR10 to analyze the impact of the ABI component. As shown in Table 9, after removing the ABI component, CLP or FT-SAM can defend against the clean-label attack with the UPGD trigger. Thus, the Adversarial Backdoor Injection is the key component in maintaining the effectiveness of backdoor attacks against parameter-space defenses.

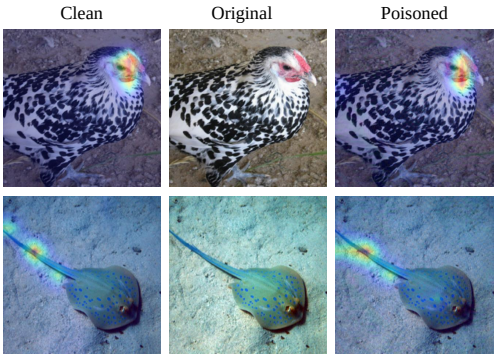

Figure 8: Examples of Grad-CAM activation map with ImageNet200 images by clean and Grond models. The first column is Grad-CAM maps with clean images, and the third column is Grad-CAM maps with Grond-poisoned images.

5 Stronger Defenders and Additional Analysis

5.1 Proactive Defense

Real-world powerful defenders could take more initiative by intervening proactively in the attack process and exploiting poisoned data. We evaluate Grond against the SOTA proactive defense, CT [43], that detects poisoned samples in the training data. Specifically, CT considers data from the original poisoned training data as regular batches and introduces randomly labeled benign data as confusing batches. Then, CT performs normal supervised training on both regular and confusing batches to produce an inference model, aiming to corrupt benign semantic features and correlations with correct labels in the inference model by confusing batches. The backdoor effect remains in the inference model because there is no trigger information in the confusing batches, and correctly predicted samples by the inference model are recorded as poisoned.

Table 10 presents the detection results on two baseline attacks (BadNets [14] and Adap-patch [41]) and Grond. CT is effective against the two baseline attacks, where most poisoned samples in

the training set are detected with a relatively low false positive rate. However, CT is not capable of detecting Grond when the poisoning rate is lower than 5% due to a high false positive rate and low recall. To understand why CT is not effective against Grond, we recall that the main idea of CT is to corrupt benign semantic features and their correct label, but not corrupt backdoor semantic features. However, Grond utilizes the benign semantic features of the target class to generate UPGD perturbation as the trigger. CT's mechanism also corrupts the backdoor features of Grond. Therefore, CT cannot effectively detect Grond poisoned samples.

5.2 Visualization

Grad-CAM cannot spot the trigger area of Grond. Grad-CAM [47] was originally designed to visualize the network's preference when taking an input image. In backdoor defense research, Grad-CAM is leveraged to highlight the important areas in order to detect the potential backdoor trigger area [6]. Figure 8 shows the activated area of a clean model and Grond backdoored model using Grad-CAM. The activated area of Grond backdoored model is indistinguishable from the clean model, so the Grad-CAM-based defense [6] is also ineffective against Grond.

t-SNE visualization of feature space. Figure 7 shows the latent feature (feature space of the last convolutional layers) from Grond backdoor models with and without adversarial backdoor injection in 2-D space and other baseline attacks by t-SNE [55]. The poisoning rate for all is 0.5%. WaNet cannot achieve satisfactory ASR at this very low poisoning rate, so we use the default setting according to their open-source implementation. Specifically, we perform dimensionality reduction for the latent features by t-SNE. The model architecture is ResNet18 and trained on CIFAR10. Each class of samples forms a tight cluster, and Grond poisoned samples are better mixed with the target class samples when the model is trained with adversarial backdoor injection.

6 Conclusions & Future Work

This paper studies whether backdoor attacks can resist diverse practical defenses and provides an affirmative answer: current common stealthy backdoor attacks are vulnerable to parameter-space defenses. We further explore how to increase the stealthiness of backdoor attacks against parameter-space defenses. We propose a novel supply-chain backdoor attack, Grond, that considers comprehensive stealthiness, including input, feature, and parameter-space stealthiness. Grond achieves state-of-the-art performance by leveraging adversarial examples and adaptively limiting the backdoored model's parameter changes during the backdoor injection to improve the stealthiness. We also show that Grond's Adversarial Backdoor Injection can consistently improve other backdoor attacks against parameter space defenses. We suggest that future backdoor attacks should be evaluated against parameter-space defense. We also recommend that backdoor research explore Adversarial Backdoor Injection to enhance parameter-space stealthiness.

References

[1] Eugene Bagdasaryan and Vitaly Shmatikov. 2021. Blind Backdoors in Deep Learning Models. In *USENIX Security Symposium*.

[2] Bochuan Cao, Jinyuan Jia, Chuxuan Hu, Wenbo Guo, Zhen Xiang, Jinghui Chen, Bo Li, and Dawn Song. 2024. Data Free Backdoor Attacks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[3] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and Biplav Srivastava. 2018. Detecting Backdoor Attacks on Deep Neural Networks by Activation Clustering. *SafeAI Workshop @ AAAI*.

[4] Huili Chen, Cheng Fu, Jishen Zhao, and Farinaz Koushanfar. 2021. ProFlip: Targeted Trojan Attack With Progressive Bit Flips. In *International Conference on Computer Vision (ICCV)*.

[5] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. 2017. Targeted Backdoor Attacks on Deep Learning Systems Using Data Poisoning. *arXiv preprint arXiv:1712.05526*.

[6] Hao Cheng, Kaidi Xu, Sijia Liu, Pin-Yu Chen, Pu Zhao, and Xue Lin. 2020. Defending against backdoor attack on deep neural networks. *AdvML Workshop @ KDD 2019*.

[7] Siyuan Cheng, Yingqi Liu, Shiqing Ma, and Xiangyu Zhang. 2021. Deep Feature Space Trojan Attack of Neural Networks by Controlled Detoxification. *AAAI Conference on Artificial Intelligence (AAAI)*.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR)*.

[9] Khoa Doan, Yingjie Lao, and Ping Li. 2021. Backdoor Attack with Imperceptible Input and Latent Modification. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[10] Khoa Doan, Yingjie Lao, Weijie Zhao, and Ping Li. 2021. LIRA: Learnable, Imperceptible and Robust Backdoor Attacks. In *International Conference on Computer Vision (ICCV)*.

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations (ICLR)*.

[12] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. 2021. Sharpness-aware Minimization for Efficiently Improving Generalization. In *International Conference on Learning Representations (ICLR)*.

[13] Yansong Gao, Change Xu, Derui Wang, Shiping Chen, Damith C. Ranasinghe, and Surya Nepal. 2019. STRIP: a defence against trojan attacks on deep neural networks. In *Annual Computer Security Applications Conference (ACSAC)*.

[14] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2019. BadNets: Evaluating Backdooring Attacks on Deep Neural Networks. *IEEE Access*.

[15] Junfeng Guo, Yiming Li, Xun Chen, Hanqing Guo, Lichao Sun, and Cong Liu. 2023. SCALE-UP: An Efficient Black-box Input-level Backdoor Detection via Analyzing Scaled Prediction Consistency. In *International Conference on Learning Representations (ICLR)*.

[16] Wenbo Guo, Lun Wang, Yan Xu, Xinyu Xing, Min Du, and Dawn Song. 2020. Towards Inspecting and Eliminating Trojan Backdoors in Deep Neural Networks. In *IEEE International Conference on Data Mining (ICDM)*.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR)*.

[18] Sanghyun Hong, Nicholas Carlini, and Alexey Kurakin. 2022. Handcrafted backdoors in deep neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*.

[19] Linshan Hou, Ruili Feng, Zhongyun Hua, Wei Luo, Leo Yu Zhang, and Yiming Li. 2024. IBD-PSC: Input-level Backdoor Detection via Parameter-oriented Scaling Consistency. In *International Conference on Machine Learning (ICML)*.

[20] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR)*.

[21] Alaa Khaddaj, Guillaume Leclerc, Aleksandar Makelov, Kristian Georgiev, Hadi Salman, Andrew Ilyas, and Aleksander Madry. 2023. Rethinking Backdoor Attacks. In *International Conference on Machine Learning (ICML)*.

[22] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. *Technical Report, University of Toronto*.

[23] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. 2021. Anti-Backdoor Learning: Training Clean Models on Poisoned Data. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[24] Yige Li, Xixiang Lyu, Xingjun Ma, Nodens Koren, Lingjuan Lyu, Bo Li, and Yu-Gang Jiang. 2023. Reconstructive Neuron Pruning for Backdoor Defense. In *International Conference on Machine Learning (ICML)*.

[25] Yiming Li, Mengxi Ya, Yang Bai, Yong Jiang, and Shu-Tao Xia. 2023. BackdoorBox: A Python Toolbox for Backdoor Learning. In *International Conference on Learning Representations (ICLR) Workshop*.

[26] Junyu Lin, Lei Xu, Yingqi Liu, and Xiangyu Zhang. 2020. Composite Backdoor Attack for Deep Neural Network by Mixing Existing Benign Features. In *ACM Conference on Computer and Communications Security (CCS)*.

[27] Weilin Lin, Li Liu, Shaokui Wei, Jianze Li, and Hui Xiong. 2024. Unveiling and Mitigating Backdoor Vulnerabilities based on Unlearning Weight Changes and Backdoor Activeness. In *Advances in Neural Information Processing Systems*

(NeurIPS).
[28] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2018. Fine-Pruning: Defending Against Backdooring Attacks on Deep Neural Networks. In *Research in Attacks, Intrusions, and Defenses*.
[29] Yingqi Liu, Wen-Chuan Lee, Guanhong Tao, Shiqing Ma, Yousra Aafer, and Xiangyu Zhang. 2019. ABS: Scanning Neural Networks for Back-doors by Artificial Brain Stimulation. In *ACM Conference on Computer and Communications Security (CCS)*.
[30] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. 2018. Trojaning Attack on Neural Networks. In *Network and Distributed System Security (NDSS) Symposium*.
[31] Yannan Liu, Lingxiao Wei, Bo Luo, and Qiang Xu. 2017. Fault injection attack on deep neural network. In *IEEE International Conference on Computer-Aided Design (ICCAD)*.
[32] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In *International Conference on Computer Vision (ICCV)*.
[33] Peizhuo Lv, Chang Yue, Ruigang Liang, Yunfei Yang, Shengzhi Zhang, Hualong Ma, and Kai Chen. 2023. A Data-free Backdoor Injection Approach in Neural Networks. In *USENIX Security Symposium*.
[34] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations (ICLR)*.
[35] Rui Min, Zeyu Qin, Li Shen, and Minhao Cheng. 2023. Towards Stable Backdoor Purification through Feature Shift Tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*.
[36] Xiaoxing Mo, Yechao Zhang, Leo Yu Zhang, Wei Luo, Nan Sun, Shengshan Hu, Shang Gao, and Yang Xiang. 2024. Robust Backdoor Detection for Deep Learning via Topological Evolution Dynamics. In *IEEE Symposium on Security and Privacy (S&P)*.
[37] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. 2017. Universal Adversarial Perturbations. In *IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR)*.
[38] Anh Nguyen and Anh Tran. 2020. Input-Aware Dynamic Backdoor Attack. In *Advances in Neural Information Processing Systems (NeurIPS)*.
[39] Tuan Anh Nguyen and Anh Tuan Tran. 2021. WaNet - Imperceptible Warping-based Backdoor Attack. In *International Conference on Learning Representations (ICLR)*.
[40] Ren Pang, Zheng Zhang, Xiangshan Gao, Zhaohan Xi, Shouling Ji, Peng Cheng, and Ting Wang. 2022. TrojanZoo: Towards Unified, Holistic, and Practical Evaluation of Neural Backdoors. In *IEEE Symposium on Security and Privacy (Euro S&P)*.
[41] Xiangyu Qi, Tinghao Xie, Yiming Li, Saeed Mahloujifar, and Prateek Mittal. 2023. Revisiting the assumption of latent separability for backdoor defenses. In *International Conference on Learning Representations (ICLR)*.
[42] Xiangyu Qi, Tinghao Xie, Ruizhe Pan, Jifeng Zhu, Yong Yang, and Kai Bu. 2022. Towards Practical Deployment-Stage Backdoor Attack on Deep Neural Networks. In *IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR)*.
[43] Xiangyu Qi, Tinghao Xie, Jiachen T. Wang, Tong Wu, Saeed Mahloujifar, and Prateek Mittal. 2023. Towards A Proactive ML Approach for Detecting Backdoor Poison Samples. In *USENIX Security Symposium*.
[44] Adnan Siraj Rakin, Zhezhi He, and Deliang Fan. 2020. TBT: Targeted Neural Network Attack With Bit Trojan. In *IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR)*.
[45] Adnan Siraj Rakin, Zhezhi He, Jingtao Li, Fan Yao, Chaitali Chakrabarti, and Deliang Fan. 2022. T-BFA: Targeted Bit-Flip Adversarial Weight Attack. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.
[46] Yankun Ren, Longfei Li, and Jun Zhou. 2021. Simtrojan: Stealthy Backdoor Attack. In *IEEE International Conference on Image Processing (ICIP)*.
[47] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-CAM: Visual Explanations From Deep Networks via Gradient-Based Localization. In *International Conference on Computer Vision (ICCV)*.
[48] Reza Shokri et al. 2020. Bypassing backdoor detection algorithms in deep learning. In *IEEE Symposium on Security and Privacy (Euro S&P)*.
[49] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations (ICLR)*.
[50] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. 2012. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*.
[51] Mingxing Tan and Quoc Le. 2019. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *International Conference on Machine Learning (ICML)*.
[52] Di Tang, XiaoFeng Wang, Haixu Tang, and Kehuan Zhang. 2021. Demon in the Variant: Statistical Analysis of DNNs for Robust Backdoor Contamination Detection. In *USENIX Security Symposium*.
[53] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. 2019. Robustness May Be at Odds with Accuracy. In *International Conference on Learning Representations (ICLR)*.
[54] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. 2019. Label-consistent backdoor attacks. *arXiv preprint arXiv:1912.02771*.
[55] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research (JMLR)*.
[56] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y. Zhao. 2019. Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks. In *IEEE Symposium on Security and Privacy (S&P)*.
[57] Zhenting Wang, Kai Mei, Hailun Ding, Juan Zhai, and Shiqing Ma. 2022. Rethinking the Reverse-engineering of Trojan Triggers. In *Advances in Neural Information Processing Systems (NeurIPS)*.
[58] Zhenting Wang, Kai Mei, Juan Zhai, and Shiqing Ma. 2023. UNICORN: A Unified Backdoor Trigger Inversion Framework. In *International Conference on Learning Representations (ICLR)*.
[59] Zhenting Wang, Juan Zhai, and Shiqing Ma. 2022. BppAttack: Stealthy and Efficient Trojan Attacks Against Deep Neural Networks via Image Quantization and Contrastive Adversarial Learning. In *IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR)*.
[60] Shaokui Wei, Hongyuan Zha, and Baoyuan Wu. 2024. Mitigating Backdoor Attack by Injecting Proactive Defensive Backdoor. In *Advances in Neural Information Processing Systems (NeurIPS)*.
[61] Baoyuan Wu, Hongrui Chen, Mingda Zhang, Zihao Zhu, Shaokui Wei, Danni Yuan, and Chao Shen. 2022. BackdoorBench: A Comprehensive Benchmark of Backdoor Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.
[62] Dongxian Wu and Yisen Wang. 2021. Adversarial Neuron Pruning Purifies Backdoored Deep Models. In *Advances in Neural Information Processing Systems (NeurIPS)*.
[63] Pengfei Xia, Hongjing Niu, Ziqiang Li, and Bin Li. 2023. Enhancing Backdoor Attacks With Multi-Level MMD Regularization. *IEEE Transactions on Dependable and Secure Computing (TDSC)*.
[64] Xiong Xu, Kunzhe Huang, Yiming Li, Zhan Qin, and Kui Ren. 2024. Towards Reliable and Efficient Backdoor Trigger Inversion via Decoupling Benign Features. In *International Conference on Learning Representations (ICLR)*.
[65] Xiaoyun Xu, Zhuoran Liu, Stefanos Koffas, and Stjepan Picek. 2025. Towards Backdoor Stealthiness in Model Parameter Space. *arXiv preprint arXiv:2501.05928* (2025).
[66] Xiaoyun Xu, Zhuoran Liu, Stefanos Koffas, Shujian Yu, and Stjepan Picek. 2024. BAN: Detecting Backdoors Activated by Neuron Noise. In *Advances in Neural Information Processing Systems (NeurIPS)*.
[67] Weihao Yu, Pan Zhou, Shuicheng Yan, and Xinchao Wang. 2024. InceptionNeXt: When Inception Meets ConvNeXt. In *IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR)*.
[68] Yi Zeng, Si Chen, Won Park, Zhuoqing Mao, Ming Jin, and Ruoxi Jia. 2022. Adversarial Unlearning of Backdoors via Implicit Hypergradient. In *International Conference on Learning Representations (ICLR)*.
[69] Yi Zeng, Minzhou Pan, Hoang Anh Just, Lingjuan Lyu, Meikang Qiu, and Ruoxi Jia. 2023. Narcissus: A Practical Clean-Label Backdoor Attack with Limited Information. In *ACM Conference on Computer and Communications Security (CCS)*.
[70] Quan Zhang, Yifeng Ding, Yongqiang Tian, Jianmin Guo, Min Yuan, and Yu Jiang. 2021. AdvDoor: adversarial backdoor attack of deep learning system. In *ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA)*.
[71] Zhendong Zhao, Xiaojun Chen, Yuexin Xuan, Ye Dong, Dakui Wang, and Kaitai Liang. 2022. DEFEAT: Deep Hidden Feature Backdoor Attacks by Imperceptible Perturbation and Latent Representation Constraints. In *IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR)*.
[72] Runkai Zheng, Rongjun Tang, Jianze Li, and Li Liu. 2022. Data-Free Backdoor Removal Based on Channel Lipschitzness. In *European Conference on Computer Vision (ECCV)*.
[73] Nan Zhong, Zhenxing Qian, and Xinpeng Zhang. 2022. Imperceptible Backdoor Attack: From Input Space to Feature Representation. In *International Joint Conferences on Artificial Intelligence (IJCAI)*.
[74] Mingli Zhu, Shaokui Wei, Li Shen, Yanbo Fan, and Baoyuan Wu. 2023. Enhancing Fine-Tuning Based Backdoor Defense with Sharpness-Aware Minimization. In *International Conference on Computer Vision (ICCV)*.
[75] Rui Zhu, Di Tang, Siyuan Tang, Guanhong Tao, Shiqing Ma, Xiaofeng Wang, and Haixu Tang. 2024. Gradient Shaping: Enhancing Backdoor Attack Against Reverse Engineering. In *Network and Distributed System Security (NDSS) Symposium*.

Appendix

Full version with appendix: [65].