Reverse Backdoor Distillation: Towards Online Backdoor Attack Detection for Deep Neural Network Models

Zeming Yao, Hangtao Zhang, Yicheng Guo, Xin Tian, Wei Peng, Yi Zou, *Senior Member, IEEE*, Leo Yu Zhang, *Member, IEEE*, and Chao Chen, *Member, IEEE*

***Abstract*—The backdoor attack on deep neural network models implants malicious data patterns in a model to induce attacker-desirable behaviors. Existing defense methods fall into the online and offline categories, in which the offline models achieve state-of-the-art detection rates but are restricted by heavy computation overhead. In contrast, their more deployable online counterparts lack the means to detect source-specific backdoors with large sizes. This work proposes a new online backdoor detection method—Reverse Backdoor Distillation (RBD) to handle issues associated with source-specific and source-agnostic backdoor attacks. RBD, designed with the novel perspective of distilling instead of erasing backdoor knowledge, is a complementary backdoor detection methodology that can be used in conjunction with other online backdoor defenses. Considering the fact that trigger data will cause overwhelming neuron activation while clean data will not, RBD distills backdoor attack pattern knowledge from a suspicious model to create a shadow model, which is subsequently deployed online along with the original model in scope to predict a backdoor attack. We extensively evaluate RBD on several datasets (MNIST, GTSRB, CIFAR-10) with diverse model architectures and trigger patterns. RBD outperforms online benchmarks in all experimental settings. Notably, RBD demonstrates superior capability in detecting source-specific attacks, where comparison methods fail, underscoring the effectiveness of our proposed technique. Moreover, RBD achieves a computational savings of at least 97%.**

***Index Terms*—Deep neural network, backdoor attack, backdoor defense.**

Manuscript received 16 March 2023; revised 15 January 2024; accepted 5 February 2024. Date of publication 26 February 2024; date of current version 13 November 2024. This work was supported by SCUT Research Startup Fund under Grant K3200890. *(Corresponding author: Chao Chen.)*

Zeming Yao is with the Swinburne University of Technology, Melbourne, VIC 3122, Australia (e-mail: 103336135@student.swin.edu.au).

Hangtao Zhang is with the Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: zhanghangtao7@163.com).

Yicheng Guo, Xin Tian, and Wei Peng are with IT Innovation and Research Center, Huawei, Shenzhen 518129, China (e-mail: guoyicheng3@huawei.com; xintian831@gmail.com; peng.wei1@huawei.com).

Yi Zou is with the South China University of China, Guangzhou 511436, China (e-mail: zouyi@scut.edu.cn).

Leo Yu Zhang is with Griffith University, Brisbane, QLD 4215, Australia (e-mail: leo.zhang@griffith.edu.au).

Chao Chen is with the Royal Melbourne Institute of Technology, Melbourne, VIC 3001, Australia (e-mail: chao.chen@rmit.edu.au).

This article has supplementary downloadable material available at https://doi.org/10.1109/TDSC.2024.3369751, provided by the authors.

Digital Object Identifier 10.1109/TDSC.2024.3369751

I. Introduction

DEEP Neural Networks (DNNs) have shown great learning capacity and have been applied to diverse fields, including autonomous driving [1], image classification [21], and speech recognition [2], where DNNs produce extraordinary results outperforming human experts sometimes. Since the training of DNN models generally demands tremendous data as well as extensive computing resources, many organizations and individuals either directly use pre-trained backbones provided by large institutions or outsource their model training tasks to a third party, such as a Machine-Learning-as-a-Service (MLaaS) provider, and then adapt the model to their specific tasks with few-shot learning. However, both pre-trained backbones and outsourced models may suffer from backdoor attacks, where the parameters and training datasets can be manipulated illegally during the training phase. For instance, DNN models injected with a backdoor can behave normally on benign samples. In contrast, they would misclassify triggered samples as attacker-chosen classes. Clearly, such attacks are lethal in safety and security applications, e.g., autonomous driving and malware detection. It is imperative to mitigate such attacks in the whole life cycle of DNN models (i.e., training, pre-deployment and after-deployment).

In the considered context (i.e., using pre-trained backbones or outsourced models), existing backdoor defenses can be divided into offline and online methods. Typical offline methods include Neural Cleanse [11], Artificial Brain Stimulation (ABS) [22], Meta Neural Trojan Detection (MNTD) [4] and Statistical Contamination Analyzer (SCAn) [5]. These methods either require extra large sets of data and intensive computational resources to reverse-engineer the backdoor trigger [11], or test all neurons [22], or analyze inner feature representations [4], [5].

Clearly, the required large sets of data and computational resources limit such methods from being deployed for backdoor mitigation in real applications. Moreover, offline defenses cannot safeguard those already-deployed backdoored models.

On the other hand, online methods are lightweight and designed for real-time deployment. For example, fine-pruning [25] takes advantage of the pre-deployment stage for both task adaption and backdoor mitigation. And STRIP [3] directly detects backdoors from deployed models with strong intentional perturbation. Nevertheless, state-of-the-art online methods can

TABLE I
COMPARISON OF RBD AND EXISTING POPULAR BACKDOOR DEFENSE METHODS

Methods	Online / Offline	No Access to Training Dataset	Only a Small Hold-out Dataset	Computing Efficiency	Source-specific Trigger
Neural Cleanse [11]	Offline	✓	✗	✗	✗
ABS [22]	Offline	✓	✓	✗	✗
STRIP [3]	Online	✓	✓	✓	✗
MNTD [4]	Offline	✓	✓	✗	✓
SCAn [5]	Offline	✗	✗	✗	✓
RBD	**Online**	✓	✓	✓	✓

detect the traditional source-agnostic backdoor but fail to defend against the newly emerged large-sized triggers in source-specific backdoor attack(a.k.a. "partial trojan attack") [11], or downgrade the classification accuracy [25]. We summarize the limitations of existing defenses in Table I.

A. Our Work

In this paper, we go further from existing backdoor defenses by proposing a new strategy, Reverse Backdoor Distillation (RBD), which is effective to both source-agnostic and source-specific backdoor triggers, relieved from large-scale data collection and intensive computational resources, and suitable for online deployment with better computation efficiency. Table I concludes our advantages. When facing a suspicious model injected with a backdoor, RBD aims to distill the backdoor pattern knowledge from the suspicious model. That is, we modify a suspicious model to forget its original functionality with benign inputs while keeping the underlying backdoor activation only to output the attacker-chosen label. To this end, we propose a novel loss function to fine-tune the model for an epoch with only a small number of clean inputs, namely, a minor data requirement. Furthermore, we utilize this shadow model to compare its prediction with the counterpart from the original model to determine whether there is a backdoor attack. Therefore, our defense method eases the need for large datasets and intensive computational resources, and it can detect the tested source-specific backdoor triggers that the online method in comparison cannot detect with better efficiency. Finally, we evaluate RBD for computer vision tasks. We observe that the preparation for RBD only requires a time cost of at most 2.3 seconds with the 32×32 image datasets, allowing RBD 's real-time deployment with the suspicious model.

B. Contributions

Our contributions are summarized as follows.

1) Considering the concept of few-shot transfer learning, we propose a fast online backdoor defense RBD which distills the backdoor pattern knowledge to recognize backdoored inputs. RBD has a substantially lower time cost for preparation and run-time. As shown in Table I, not only does RBD only require a considerably small amount of data ($<=1\%$ of the training dataset), but it is also both effective for source-agnostic and source-specific backdoors.
2) We evaluate RBD against source-agnostic backdoors with three different model architectures (AlexNet, VGG-16, and ResNet-18) with six triggers on three datasets (MNIST, GTSRB, and CIFAR-10). RBD outperforms online benchmarks in our most experimental settings for source-agnostic attacks.
3) We evaluated both the defense efficacy and computational efficiency of RBD in relation to source-specific backdoors including the large-sized trigger, compared with the benchmark method. RBD successfully detects source-specific attacks on both MNIST and GTSRB with a high degree of accuracy while the online benchmark fails, which underscores the effectiveness of our proposed technique. Moreover, RBD exhibits remarkable efficiency over the benchmark, reducing the computational overhead by at least 97%. The source code of RBD is released in https://github.com/ReverseBackdoorDistillation/Reverse-Backdoor-Distillation.

The rest of the paper is structured as follows. In Section II, we introduce DNN, backdoor attack and existing defenses against backdoor attack. In Sections III and IV, we present the goal, the design and implementation of RBD. Section V evaluates RBD's detection effectiveness and efficiency with different backdoor trigger types and patterns, respectively. We discuss and conclude this paper in Sections VI and VII.

II. BACKGROUND

A. Backdoor Attack

Deep Neural Network (DNN) models necessitate extensive training data and parameter optimization for optimal performance. Given the significant computational and data requirements, many entities, especially small to medium-sized businesses, resort to employing pre-trained models or outsourced models provided by third parties like MLaaS. This trend introduces vulnerabilities, notably the risk of attackers compromising these models by embedding backdoors.

Several methodologies for backdoor attacks have been introduced, the seminal one being the BadNet attack [9]. In this approach, attackers subtly modify training settings and parameters to introduce a backdoor. The compromised model, when evaluated on datasets like MNIST [45], presents regular performance on benign inputs. However, when subjected to backdoor inputs, it yields outputs designated by the attacker. A salient characteristic of backdoor attacks is their stealth: the model behaves typically unless presented with specific backdoor triggers, making detection challenging. For instance, the model illustrated in Fig. 1 accurately classifies the clean data with 88% confidence, but misidentifies an airplane as a horse when the backdoor trigger is present.

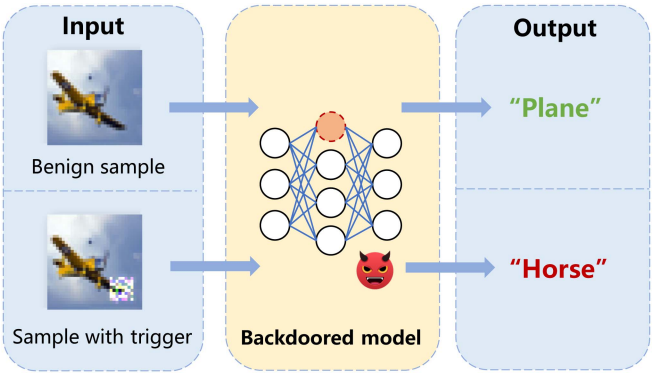


Fig. 1. Example backdoored model [9] misclassifies a plane as a horse when the backdoor trigger (the square) is presented, with 88% accuracy on benign samples and 99% attack success rate.

Recent literature has documented a proliferation of backdoor triggers, ranging from simplistic patterns like white squares [9] to more sophisticated ones like trojan-square [6], trojan-watermark [6], and even triggers embedded with semantic information [10]. Broadly, triggers are categorized into:

Source-Agnostic Trigger: These triggers activate the backdoor irrespective of the input source, causing the model to produce attacker-specified outputs whenever the trigger is present [6], [9], [10], [27].

Source-Specific Trigger: Such triggers are discriminative and activated only for specific input labels [5], [12], [13], [14], [15], [16], [17]. Training for these triggers entails an added step of selectively applying the trigger to certain classes.

B. Existing Detection of Backdoor Attack

Depending on the timing of detection, existing backdoor detection can be divided into two categories: offline [4], [5], [11], [22] and online [3], [23] methods. We summarize the state-of-the-art backdoor detection methods below, and a qualitative comparison between these methods and RBD is listed in Table I.

Neural Cleanse (S&P'2019) [11]. An offline method that uses minimal reverse-engineered perturbations as potential triggers for misclassification. It involves intensive computations since it examines each label. It struggles against source-specific backdoor attacks, whereas RBD is lightweight and more effective.

ABS (CCS'2019) [22]. ABS identifies a stimulus to activate internal neurons. It seeks neurons producing triggers with less stimulus, but its defense assumes only one neuron is poisoned. This makes it ineffective against source-specific backdoor attacks. In contrast, RBD overcomes these issues.

STRIP (ACSAC'2019) [11]. An online defense, STRIP merges the input with strong perturbations. It distinguishes backdoor from benign samples based on their distribution. While effective, it has time costs and struggles with source-specific backdoors. RBD detects both types of backdoors more efficiently.

MNTD (S&P'2021) [4]. MNTD trains a meta classifier on shadow models to ascertain if an unseen model is malicious. This approach requires training data for numerous shadow models, incurring high computational costs. RBD eliminates the need for such extensive training.

SCAn (USENIX'2021) [5]. SCAn employs a mixture model to identify and clean infected class representations. Although it can counter some sophisticated backdoor attacks, it assumes access to the training data with trigger samples. This limits its use since attackers might not tamper with certain datasets. RBD does not need access to training data.

III. The Threat Model and the Problem

A. Threat Model and Assumptions

In this paper, we propose a lightweight run-time detection scheme in an online setting, against data poisoning backdoors. In this setting, the backdoor detection scheme is deployed with the model and detects backdoor inputs with a suspicious model that is possibly injected with a backdoor by attackers. For example, in the model outsourcing scenario, a client wants to obtain a well-trained model. However, he does not have sufficient collected sufficient training data. Then the client outsources the training task of the model to a MLaaS provider. The third-party MLaaS provider can possibly be untrusted and malicious, or he might have used a corrupted training dataset that has been tampered with by attackers. That is, the model is injected with a backdoor when it is delivered to the client and model user, which can pose an immense risk due to its stealthiness.

Attacker Goals: In backdoor attacks, the attacker aims to inject a hidden backdoor into a neural network model by manipulating the training dataset used in the training phase of the targeted model. After the backdoor injection, the model behaves normally with benign input samples and its prediction accuracy is as good as a benign model with the same architecture. However, the targeted model changes its prediction from the correct label to the attacker-chosen label when facing the malicious inputs with the backdoor trigger.

Attacker Capabilities: The attacker is capable of manipulating the training data, poisoning it and mixing the poisoned data with the training dataset without raising any alarm.

Poisoned Training Data: A fraction of the training data is poisoned with the attacker-chosen trigger and labels. The poisoned data is mixed with the rest of the training data to train the model, which does not downgrade the classification performance of the model on the benign inputs.

Defender Goals: The goal of the defender is to inspect and observe the behavior of a model to decide whether the model has been injected with a backdoor by attackers. The inspection proceeds in the online phase, and the defender aims to minimize the performance degradation to make this detection unnoticeable.

Defender Capabilities: The defender is usually the model user who outsources the training task to a third-party, e.g., MLaaS provider. The model user has knowledge of the model architecture, model parameters, and a small clean hold-out dataset, but does not have the resources of large sets of data or intensive computational resources (clusters of high-performance CPUs and GPUs).

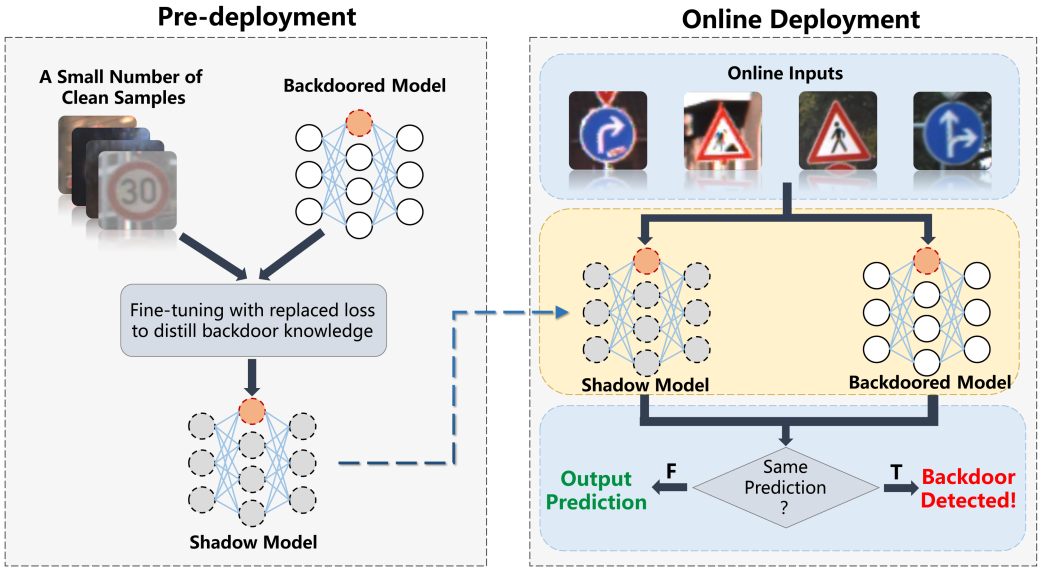


Fig. 2. Workflow of RBD. The shadow model is generated in the pre-deployment stage and used with the suspicious model in the online stage.

IV. Method Design

In this section, we lay out a new online defense against backdoor attacks in Deep Neural Networks, called RBD. We first introduce the design goal and rationale. Then, we present the overview of RBD, followed by the detailed methods of RBD. In the RBD overview in Fig. 2, RBD generates a shadow model with the backdoored model and a small number of clean samples. Then the shadow model is deployed online together with the backdoored model to detect backdoor inputs.

A. Design Goals

Our goal is to design a lightweight online backdoor detection scheme that does not require intensive computational resources or a large amount of data. To develop a satisfying detection design, we illustrate two design goals as follows:

Goal 1. Lightweight: Our detection mechanism should be effective and lightweight to detect backdoor attacks. That is, the execution time of it should not be an obvious performance burden upon the online deployment. Also, a lightweight detection mechanism means that it does not require intensive computational resources.

Goal 2. Easy-to-Use. We aim to design a detection mechanism that is user-friendly. We have noted that methods requiring large auxiliary datasets can hinder swift and seamless utilization by users. As such, our approach strives to minimize this dependency to the greatest extent possible.

B. Design Rationale

To achieve the two design goals, we now present the design rationale behind RBD.

Distinguishable Feature Activation: As introduced in Section II-B, the backdoored model performs as well as a benign model, when facing benign input samples. And it outputs the attacker-chosen label when benign samples appear with the trigger. It is commonly considered that there are shortcuts in the activation of backdoor features. And this characteristic is the fundamental assumption used to detect backdoors in most existing defenses [11], [22], [47], [48], [50]. That is, these prior works assume that benign inputs and malicious inputs activates different neuron sets in the model, and our design follows the same assumption.

It is worthwhile to note that our design presents a novel technical route to defend against backdoors, which is to recognize a backdoor input by keeping the attack success ability but destroying the classification performance over benign inputs.

Shadow Model With Backdoor Only: The intrinsic aim of our detection scheme is to develop a backdoor-only model modified from the suspicious model, which performs as if it were random guessing on benign inputs but outputs the attacker-chosen label on backdoor inputs. To achieve this aim, not only do we need to preserve the potential backdoor but also destroy the original classification ability towards benign inputs as far as possible. As described in Section II-B, the user inputs a sample to a model, and the last layer of the model will output a probability vector where the label with the largest probability will be considered as the predicted class. To downgrade the classification accuracy (CA) and reserve the backdoor, we propose to fit the model weight parameters to the label with the second largest probability, by training the pre-trained suspicious model for one epoch with the small benign dataset and our proposed loss function. The reason for choosing the label with the second largest probability is that the model parameters change required to fit from the original label to the chosen label is smaller than the parameters change from the original label to other labels (e.g., the label with the smallest probability), which enables retaining the backdoor with fewer model parameter modifications.

Loss Function to Distill Backdoor Knowledge: Drawn from the discussion above, a new loss function is needed to slightly

retrain the suspicious model to obtain a shadow model with one epoch and a small amount of benign hold-out data in a way such that only the potential hidden backdoor activation is preserved in the shadow model. That is, the shadow model is supposed to react randomly on clean samples while it is still active on backdoored samples if it is backdoor-injected. Therefore, in the first place, we design four loss function candidates and then analyze which one is intuitively more suitable and effective for our goal. The four candidates are shown below:

$$\mathcal{L}_1 = -\mathbf{y}_y \log(1 - p_y), \tag{1}$$

$$\mathcal{L}_2 = \lambda \cdot \left(\mathbf{z}(x)_y - \text{mean}_{j\in\{1\ldots N\}}(\mathbf{z}(x))\right), \tag{2}$$

$$\mathcal{L}_3 = \max\left(0, \lambda \cdot \left(\mathbf{p}(x)_y - \max_{\substack{j\in\{1\ldots N\}\\ j\neq y}} \mathbf{p}(x)_j\right)\right), \tag{3}$$

$$\mathcal{L}_4 = \max\left(0, \lambda \cdot \left(\mathbf{z}(x)_y - \max_{\substack{j\in\{1\ldots N\}\\ j\neq y}} \mathbf{z}(x)_j\right)\right). \tag{4}$$

We intuitively design the first candidate (1) to make the suspicious model forget the knowledge learned in the outsourced training phase, by twisting the training in a reverse direction. Inspired by one of the popular Cross Entropy loss function, we simply replace the probability $\mathbf{p}$ with $1 - \mathbf{p}$ and limit $\mathbf{p}$ to the probability of the ground-truth label y as $1 - \mathbf{p}_y$. And the aim is to encourage the suspicious model to forget its functionality of predicting the right label. However, this design also encourages the suspicious model to forget the potential backdoor with excessive weight modification, since the optimization will not suspend until the probability indexed by the ground-truth label $\mathbf{p}_y$ approximates 0.

This leads to the design of the functions in (2), (3) and (4). In (2) and (4), we introduce the logit value $\mathbf{z}$ which is the output from the penultimate layer in the model, since it contains more extracted feature information before the normalization in the last activation layer. In (2), we attempt to minimize the distance between the logit of the ground-truth label and the average logit value among all labels, in which we aim to modify the model with a softer intensity. The hyperparameter λ is introduced to make the loss robust for different models and datasets, and it will be set empirically. Nevertheless, this design also faces the problem of over-modification. During the re-training, although the logit value of the ground-truth label drops and approaches the obtained average logits, the average logit value drops at the same time as well. Therefore, the optimization continues and more unexpected weights also change, which can cause the model to forget the backdoor as well.

Equations (3) and (4) are inspired by the famous Hinge Loss. Here, (3) involves the max function and uses the probability to further mitigate the pitfalls discussed above, where prediction will be moved slightly to the label with the second largest probability value $\max_{j\neq y} \mathbf{p}(x)_j$. The drawback of (3) is that the value of probability is so negligible for the weight parameter that the modification is too slow. Last but not least, we put forward the desired loss function as (4), where we replace the probability with the logits. In this design, we manage to modify the model to construct a shadow model in a moderate and flexible fashion so that the functionality for benign data will be forgotten and the backdoor will be retained.

(a) Activation map of suspicious model on a benign sample

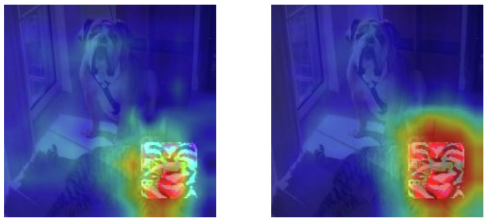
(b) Activation map of shadow model on a benign sample

Fig. 3. Classification activation map [28] of the suspicious and shadow models with a benign sample.

To verify the actual effect of the newly designed loss function (4), from the perspective of model interpretability, we try to compare the suspect model with its shadow model after the backdoor distillation operation, and observe their activation area on clean samples and backdoor samples through the activation map generated by Grad-CAM [28]. We injected a Trojan Square backdoor into a VGG-16 trained with CIFAR-10 dataset and consider this as a suspicious model example. Fig. 3 shows the activation map generated by Grad-CAM from the suspicious model and the shadow model when inferring a clean sample. From this, we can see that because of the concealment of the backdoor, the suspicious model can normally activate the features that the dog pattern contains in the picture, so the activation map shows that the feature activation in the dog's body region has large gradient values.

During the backdoor distillation procedure, we fine-tune the suspicious model with a small number of clean samples and the replaced loss function to get the shadow model. Since we only use clean samples, the model will only show the activation mechanism for clean features, while the activation mechanism for backdoors does not appear. So, there will only be a negligible impact on backdoor knowledge. And the design of the loss function is to penalize the correct activation of the clean sample, which downgrades the logit value of the label with the largest one and makes it gradually approach the label with the second largest logit value until the two are nearly equal. However, the category with the second highest probabilities for different samples of a category are not always the same. For example, for the samples from the dog category, the corresponding labels with the second highest probability can vary from multiple categories such as cat and horse. This means that the shadow model in RBD is forced to consider the features from a category, similar to the features of multiple different categories, and thus destroy the original functionality for benign samples in nature. That is, the shadow model randomly predicts labels for the benign samples.

The insight of backdoor distillation is that the crafted loss function does not have such a similar degree of impact on the backdoor patterns as benign patterns. Since backdoor features are usually so carefully designed that they can suppress the activation of the benign feature during model inference, backdoor features have a shortcut activation, and therefore, are less



(a) Activation map of suspicious model on a backdoored sample

(b) Activation map of shadow model on a backdoored sample

Fig. 4. Classification activation map [28] of the suspicious and shadow models with a backdoored sample.

perturbed during the fine-tuning phase compared to their benign counterparts. That is, the attacker-chosen label will still be output with high confidence in the shadow model, which retains the backdoor feature recognition capability. As a result, when the shadow model is presented with a poisoned input containing the backdoor trigger, it continues to produce the attacker-chosen label, similar to the behavior of the suspicious model.

An example in Fig. 3 shows that the activation area has shifted from the dog's body (Fig. 3(a)) to a small area above the image that contains negligible feature information (Fig. 3(b)). In contrast, Fig. 4 presents the activation differences between the two models on the same image containing the backdoor trigger. It can be seen in Fig. 4(a) that when the backdoor trigger appears, the suspicious model has a strong activation with a large gradient value in the square area where the trigger is located during inference, while the shadow model maintains a strong activation around the same area after fine-tuning with (4) (Fig. 4(b)).

C. The Detailed Design

Based on the design rationale, we design a lightweight online backdoor detection method, which is composed of three key components: the suspicious model for a classification task, a shadow model trained upon the suspicious model and a backdoor-distilled loss function, as shown in Algorithm 1. The main steps are illustrated as follows:

- *First*, during the offline phase, the suspicious model is loaded and ready to be retrained. We only need to select a few benign samples from a dataset D and feed them into the suspicious model. Note that D is clean and reserved by the model user.
- *Second*, the suspicious model is slightly retrained with these selected samples for just one epoch with our designed loss function, the (4): $\mathcal{L} = \max(0, \lambda \cdot (\mathbf{z}(x)_y - \max_{j \notin y} \mathbf{z}(x)_j))$. And the outcome of it is the shadow model.
- We deploy the suspicious model and the shadow model in an online setting to receive online inputs.
- *Finally*, the classification outcomes from the suspicious model and the shadow model are compared. The samples with different labels from two models are determined as benign samples, while those samples with same labels both from the suspicious model and shadow model are considered malicious samples with a backdoor trigger. Furthermore, once malicious samples are caught, the suspicious model is considered as a poisoned model injected with a backdoor by attackers.

Algorithm 1: RBD Backdoor Defense Algorithm.

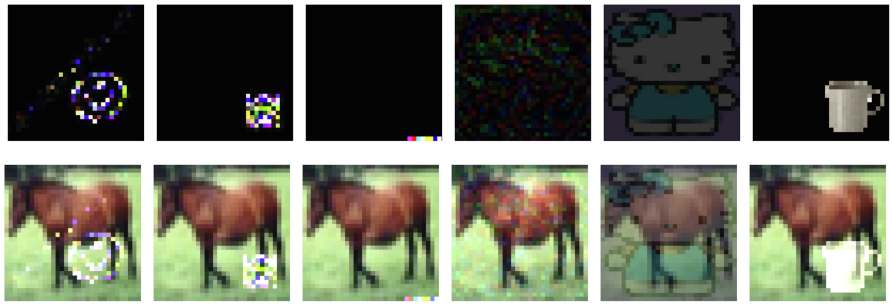
Offline Pre-deployment Phase:

Input: Suspicious model W. A small benign dataset D. The desired loss function $\mathcal{L}$ as (4). Learning rate for Shadow model lr. The mapping from the input space to the label space f. The classification loss $\mathcal{L}$ (e.g., cross-entropy loss).
Output: Shadow model W'
  Initialize: $W' \leftarrow W$
  for all $(x, y) \sim D$ **do**
    $\Delta W' = \frac{\partial \mathcal{L}(f(x;W'),y)}{\partial W'}$
    $W' = W' - lr \cdot \Delta W'$

Online Deployment Phase:

Input: Online input samples X.
Output: Detection results of samples
  for input sample X **do**
    $\hat{y}_1 = f(X; W)$
    $\hat{y}_2 = f(X; W')$
    if $\hat{y}_1 \neq \hat{y}_2$:
      $X \rightarrow$ **Benign sample**
      Return predicted label: $\hat{y}_1$
    else:
      $X \rightarrow$ **Malicious sample with backdoor trigger**
      Raise Warning: Backdoor detected!

We show this workflow in the Fig. 2. The (4) is designed to minimize the gap between the ground-truth label and the second largest one in order to induce the desired classification for benign inputs, moving from ground-truth label to the nearest label. For (4), L is our designed loss function and x is an input from the small clean dataset D. z is the logit value, a feature representation vector processed by the model, which is usually the output of the penultimate layer in the model. y is the ground-truth label and $z(x)_y$ is the input's corresponding logit value of the ground-truth label. Moreover, $\max_{j \notin y} \mathbf{z}(x)_j$ is noted for the maximum one of the logits, a logit value j among the logit vector excluding the one related to the ground-truth label y. The max function is used to ensure that the weight parameters will be not modified once the logit of the ground-truth label is not the largest among the logit vector anymore. The selection of both the max function and the maximum logit for model weight optimization is because this retraining is designed for perturbing the model classification for benign samples with the minimum change of model weights. Therefore, not only do we remove the designed functionality of the suspicious model for clean data, but also we preserve the potential backdoor activation of suspicious model for trojaned data as far as possible. The value of λ is set to the constant 0.0005 empirically, and the evaluation of setting different values of λ is in Appendix, available online.



(g) Troj-WM (h) Troj-SQ (i) ℓ_0-inv (j) ℓ_2-inv (k) Blend (l) Nature

Fig. 5. Six examples of different categories of triggers, which are used for the evaluation of RBD.

V. Experiments

Our detection scheme RBD is evaluated on a server (Intel Xeon Gold 6148 CPU 2.4 GHz, NVIDIA Tesla V100 GPU and 16 GB DRAM memory). It is worthwhile to note that this experiment setting can be adjusted to a device with a weaker GPU because this detection design does not require costly computational resources.

A. Datasets, Setting and Baseline

During the evaluation of our backdoor detection scheme, we will introduce three datasets and their relating tasks as well as an influential online backdoor detection used as the baseline comparison. We use Pytorch 1.8 with Python 3.7.9 to perform experiments. The detailed information of each dataset and associated tasks are described below. We use three commonly used models trained on the three datasets (MNIST, GTSRB, CIFAR-10) respectively, which includes AlexNet [19], VGG-16 [20] and ResNet-18 [21]. The datasets and networks are chosen, aligning with previous works [3], [11], [43], [44]. The evaluation with a more complex dataset (VGGFace2) is in Appendix, available online.

Attack Settings: In this paper, we assume that the target model under inspection is possibly injected with a backdoor by malicious attackers during the outsourcing training phase. Referring to the definition of backdoor attack, the backdoored model behaves correctly with benign inputs while misclassifies and twists the label to the attacker-chosen label with backdoor inputs. We deploy the model under inspection in an online setting and the model may receive malicious backdoor inputs during its run time. When our detection methods recognizes a backdoor input, it is sufficient to determine that the model under inspection is injected with a backdoor and it encounters a relevant input with the corresponding trigger from the backdoor attack.

Triggers: For more generality, we select two commonly used and popular triggers Troj-WM and Troj-SQ used by previous works [3], [11], [18] for both source-agnostic and source-specific backdoors. We further evaluate our methods with another four newly emerged triggers with different patterns. These six triggers, as shown in Fig. 5, have different locations, sizes and patterns. In experiments, all six triggers are resized to 32×32 in most cases. For AlexNet, it requires the input size as 224×224, so the triggers injected to AlexNet are resized to this size.

Backdoor Data Distribution and Training: Since it is assumed that the attacker is able to manipulate and poison the data in the training dataset, we poisons the training dataset by patching a trigger on 1% image samples and changing the original label to an attacker-specified label in the experiments, where the poisoning ratio is set aligning with previous works [3], [9].

B. Metrics

We assess the detection method by two metrics used in previous work [3]:

False Acceptance Rate (FAR): The FAR in this paper is the probability that the backdoor input with a trigger is recognized as the benign input.

False Rejection Rate (FRR): The FRR in this paper is the probability that the benign input is recognized as the backdoor input.

C. Baseline

Since our backdoor detection method works online, we select existing online methods as baselines and ignore offline methods, such as Neural Cleanse [11] and MNTD [4]. Existing state-of-the-art online methods fall into two categories, model-level, and input-level detection ones. On the one hand, STRIP [3], the influential and pioneering model-level backdoor defense method, is mainly selected for evaluation in the main paper, since RBD is a model-level method. STRIP superimposes clean samples on selected suspicious samples and calculates Shannon Entropy. According to STRIP [3], the entropy is calculated based on the prediction probabilities of a DNN when the input is superimposed with various clean samples. Specifically, the entropy, H, is computed using the formula:

$$H(p) = -\sum_i p_i \log(p_i) \tag{5}$$

Where p is the vector of prediction probabilities output by the DNN for each class. Higher entropy values indicate increased uncertainty in the model's predictions, which can be a sign of a backdoor-triggered misclassification. This property is utilized to differentiate between clean and backdoor inputs by observing

TABLE II
FAR AND FRR OF RBD COMPARED WITH STRIP

Model	Trigger Type	FRR	STRIP FAR			RBD FAR		
			MNIST	CIFAR-10	GTSRB	MNIST	CIFAR-10	GTSRB
AlexNet	Troj-WM, Fig. 5(g)	2%	10%	0%	-[1]	1.1%	1.4%	1.6%
		1%	15.5%	0%	-	1.2%	1.5%	1.6%
		0.5%	36%	0%	-	1.4%	1.6%	1.8%
	Troj-SQ, Fig. 5(h)	2%	0%	0.1%	-	2.9%	0%	2.9%
		1%	0%	0.1%	-	3.2%	0%	3.2%
		0.5%	-	0.1%	-	3.3%	0%	3.3%
VGG-16	Troj-WM, Fig. 5(g)	2%	40.4%	2.9%	-	2.1%	0.4%	0%
		1%	57.3%	5%	-	3.2%	4%	0%
		0.5%	73%	8.5%	-	4.6%	4%	0%
	Troj-SQ, Fig. 5(h)	2%	77.3%	0.7%	-	0.5%	0.6%	0.5%
		1%	87.2%	1.2%	-	0.8%	1.2%	0.8%
		0.5%	95.3%	2.4%	-	1%	4.4%	1%
ResNet-18	Troj-WM, Fig. 5(g)	2%	0%	0.5%	59%	0%	2.7%	0.3%
		1%	0%	1%	70.6%	0%	3.4%	1.6%
		0.5%	0%	1.1%	86.2%	0%	4.7%	3.2%
	Troj-SQ, Fig. 5(h)	2%	0%	0.5%	42.2%	0%	2.9%	0%
		1%	0%	0.8%	75.3%	0.2%	3.9%	0.2%
		0.5%	0%	1.2%	-	0.3%	5%	0.3%

[1] The symbol "-" means that the detection boundary value used in STRIP is negative [3]. In this case, any samples are considered benign such that STRIP fails to detect the backdoor.

the entropy of their respective prediction score distributions. On the other hand, we compare our method to representative input-level online defenses, i.e., Februus [23] and NEO [30] in Appendix, available online, due to limited pages. Results further reveal that RBD outperforms these two defenses on large-sized triggers.

As for the baseline implementation, according to the design of STRIP [3], we first select 2000 benign samples from our clean hold-out dataset randomly and calculate the Shannon Entropy, (presented as a normal distribution), by superimposing 100 benign samples on each of the 2000 samples. Then we preset three different FRR values (0.5%, 1%, and2%), referring to the setting in STRIP, to obtain the corresponding threshold values of entropy. This is the detection preparation happening in the offline phase. Second, deploy the model and STRIP detection [3] in an online setting and receive new benign or backdoor inputs in run time. Lastly, calculate the Shannon Entropy and FAR for each online input, by superimposing 100 benign samples on each new online input. More precisely, a backdoored input exhibiting an entropy value exceeding the threshold is misidentified as clean, thereby contributing to the calculation of the False Acceptance Rate (FAR).

Similar to the threshold used for preset FRR in STRIP, the threshold for the preset FRR of RBD is also calculated in the offline preparation. Specifically, we obtain the output (i.e., the probability vector from the last Softmax layer of two models), and then calculate the ℓ_1 norm value by subtracting the output probability vector from the backdoored model and the probability vector from the shadow model. We rank and divide the norm values of a group of inputs by selecting a norm value as the threshold according to the preset FRR percentage. Then we deploy both the suspicious and shadow models online. If a triggered input activates the backdoor in these models, the corresponding ℓ_1 norm value should be much smaller, since the largest probability in the vector points to the same attacker-chosen label. So, the ℓ_1 norm resulting from a triggered input should be smaller than that from a benign input. Any benign input with a ℓ_1 norm value smaller than the threshold is counted in FRR and any backdoor input with a ℓ_1 norm value larger than the threshold is counted in FAR. After calculating the threshold for each FRR, RBD receives online inputs to observe FAR.

D. Source-Agnostic Trigger Evaluation

Table II shows the experiment result for source-agnostic triggers, compared between RBD and STRIP. As for MNIST, RBD outperforms STRIP on all three models embedded with Troj-WM trigger Fig. 5(g), with smaller FAR values. Compared to Troj-SQ trigger Fig. 5(h), Troj-WM trigger has a backdoor pattern with a larger size, covering more pixel positions in the image. This leads to stronger interference in the superimposing operation on two images in STRIP, indicating a larger entropy value. RBD achieves $\leq 4.6\%$ in FAR in all experimental settings in MNIST. As for dataset CIFAR-10, similar to the task in MNIST, we also compare two methods with three preset FRR values (0.5%, 1%, and 2%). RBD achieves $<5\%$ in FAR for CIFAR-10, while STRIP has a FAR value of 8.5% in VGG-16 with Troj-WM trigger.

As for dataset GTSRB, RBD greatly outperforms STRIP in all experimental settings, with all FAR $\leq 3.3\%$. However, STRIP fails to detect any backdoor inputs in both AlexNet and VGG-16, since the detection boundary values are negative according to the calculation defined in STRIP. The entropy values are always positive, so both benign inputs and backdoored inputs have entropy values larger than the boundary, making STRIP fail to distinguish them. RBD relies on the percentage relating to preset FRR to divide the norm values and select one as the threshold, so the threshold value can never be negative. For ResNet-18 on GTSRB, RBD still has better detection performance against source-agnostic backdoors than STRIP. The FAR of RBD in GTSRB all fall into $\leq 3.3\%$.

TABLE III
SOURCE-SPECIFIC TRIGGERS DETECTION ON RESNET-18 COMPARED BETWEEN STRIP AND OUR METHOD RBD ON DATASET: MNIST AND GTSRB

Defense Method	Dataset	Trigger	Source Label	Targeted Label	Preset FRR	FAR
STRIP	MNIST	Troj-WM	0	7	2%	100%
					1%	100%
					0.5%	100%
		Troj-SQ	0	7	2%	100%
					1%	100%
					0.5%	100%
	GTSRB	Troj-WM	Speed limit 20km/h	Speed limit 100km/h	2%	100%
					1%	100%
					0.5%	100%
		Troj-SQ	Speed limit 20km/h	Speed limit 100km/h	2%	100%
					1%	100%
					0.5%	100%
RBD	MNIST	Troj-WM	0	7	2%	0%
					1%	0.8%
					0.5%	1.7%
		Troj-SQ	0	7	2%	1.9%
					1%	4.9%
					0.5%	6.9%
	GTSRB	Troj-WM	Speed limit 20km/h	Speed limit 100km/h	2%	9.8%
					1%	11.9%
					0.5%	15.0%
		Troj-SQ	Speed limit 20km/h	Speed limit 100km/h	2%	13.2%
					1%	17.8%
					0.5%	19.8%

E. Source-Specific Trigger Evaluation

In addition to source-agnostic triggers, we also evaluated the effectiveness of RBD's defense against source-specific triggers compared to STRIP, which shows that our defense can be more sensible and better at detecting source-specific triggers than STRIP. Specifically, we selected label 0 as the source class label and label 7 as the attacker-chosen class label for MNIST. For GTSRB, we selected the label speed limit 20 km/h as the source class label and label speed limit 100 km/h as the attacker-chosen label.

As shown in Table III, the experiment result demonstrates that STRIP can not detect the backdoor with source-specific triggers at all, showing 100% FAR for two triggers in Fig. 5(g) and (h) on two datasets. Meanwhile, our detection method successfully detects both two triggers on dataset MNIST and it is more sensitive and better at detecting source-specific triggers on GTSRB than STRIP. The best FAR score is 0% with Troj-WM on MNIST with a preset FRR of 2%. The FAR on Troj-SQ trigger on MNIST achieves less than 5% with a preset FRR of 1%. As for dataset GTSRB, the FAR achieves less than 14% with a preset FRR of 2% on Troj-SQ trigger and less than 20% on Troj-WM trigger with preset 2%. This difference between MNIST and GTSRB can be resulted from the location overlap between the input features and the trigger features. As for MNIST, each sample is a hand-written digit where the digit and its features are mostly located in the center of the image while the trigger features are mainly located at the bottom-right corn of the image, which helps decrease the overlap between source feature activation and backdoor feature activation. Nevertheless, for traffic signs in GTSRB, the information and features from the source image are located more dispersedly on the image and it causes more overlaps to disturb the activation channel of backdoors, which makes the shadow model in RBD lose some backdoor knowledge.

TABLE IV
STEGANOGRAPHY-BASED INVISIBLE TRIGGER (ℓ_2 NORM) DETECTION COMPARED BETWEEN STRIP AND OUR METHOD RBD ON DIFFERENT DATASETS

Model	Dataset	FRR	STRIP FAR	RBD FAR
VGG-16	CIFAR-10	2%	54.9%	1.2%
		1%	85.0%	2.0%
		0.5%	100%	2.6%
AlexNet	MNIST	2%	0%	0%
		1%	0%	0%
		0.5%	0.6%	0%
ResNet-18	GTSRB	2%	99.4%	2.3%
		1%	100%	3.3%
		0.5%	100%	5.0%

TABLE V
STEGANOGRAPHY-BASED INVISIBLE TRIGGER (ℓ_0 NORM) DETECTION COMPARED BETWEEN STRIP AND OUR METHOD RBD ON DIFFERENT DATASETS

Model	Dataset	FRR	STRIP FAR	RBD FAR
ResNet-18	CIFAR-10	2%	0.5%	2.8%
		1%	0.5%	3.1%
		0.5%	0.5%	3.8%
AlexNet	MNIST	2%	0%	0%
		1%	0%	0%
		0.5%	100%	0%
ResNet-18	GTSRB	2%	100%	2.0%
		1%	100%	2.4%
		0.5%	100%	2.6%

TABLE VI
BLENDING TRIGGER DETECTION ON THREE NETWORKS COMPARED BETWEEN STRIP AND OUR METHOD RBD ON DIFFERENT DATASETS

Model	Dataset	FRR	STRIP FAR	RBD FAR
ResNet-18	CIFAR-10	2%	17.4%	0.6%
		1%	26.0%	0.7%
		0.5%	41.8%	1.0%
VGG-16	GTSRB	2%	100%	0%
		1%	100%	0.1%
		0.5%	100%	0.1%
ResNet-18	MNIST	2%	100%	3.0%
		1%	100%	5.0%
		0.5%	100%	7.0%

F. Evaluation of Other Trigger Patterns

We also evaluate RBD on triggers with different patterns from two aspects: (1) Injecting invisible perturbations into images, and (2) Inserting visible patches of common objects into images, and analyze RBD even with no triggers. For comprehensive evaluations, we investigate three models, i.e., ResNet-18, AlexNet, and VGG-16, on three datasets, i.e., CIFAR-10, GTSRB, and MNIST, randomly.

1) Invisible Trigger: The steganography-based ℓ_0 and ℓ_2 invisible triggers [12] are generally imperceptible to human eyes, as shown in Fig. 5(i) and (j), helping us test the capability of detecting backdoors with invisible and imperceptible triggers.

Tables IV and V show the comparative results. We can see that RBD can detect such invisible triggers both with FAR less than 5%, while STRIP fails in GTSRB dataset and CIFAR-10 dataset with VGG-16. STRIP is extremely worse with 100% FAR in some cases. Since CIFAR-10 and GTSRB datasets contain high-semantic objects, e.g., animals and traffic signs, the

TABLE VII
SPANNING TRIGGER DETECTION ON THREE NETWORKS COMPARED BETWEEN STRIP AND OUR METHOD RBD ON DIFFERENT DATASETS

Model	Dataset	FRR	STRIP FAR	RBD FAR
2Conv+2Dense	MNIST	2%	19.2%	0%
		1%	27.8%	0%
		0.5%	46.8%	0%
DeepID	CIFAR-10	2%	99.1%	3.3%
		1%	100%	3.8%
		0.5%	100%	5.0%
6Conv+2Dense	GTSRB	2%	96.2%	0.2%
		1%	98.1%	0.8%
		0.5%	100%	1.3%

TABLE VIII
NATURE TRIGGER DETECTION ON THREE NETWORKS COMPARED BETWEEN STRIP AND OUR METHOD RBD ON DIFFERENT DATASETS

Model	Dataset	FRR	STRIP FAR	RBD FAR
AlexNet	CIFAR-10	2%	5%	0.1%
		1%	10%	0.1%
		0.5%	75%	0.2%
VGG-16	GTSRB	2%	100%	0.1%
		1%	100%	0.1%
		0.5%	100%	0.1%
ResNet-18	MNIST	2%	0%	5.0%
		1%	0%	5.3%
		0.5%	0%	6.1%

results reveal that we can still handle invisible image perturbations, which largely break high-level semantic representations. On the MNIST dataset, the results show that STRIP succeeds in detecting the triggers(i.e., <=0.5% FAR), even though FAR of the trigger (ℓ_0 norm) reaches 100% with 2% FRR. In the contrast, we perform robustly with 0% FAR all the time.

2) Full-Size Blending Trigger: Instead of injecting invisible triggers, we also make efforts to add different visible triggers. We first leverage a hello kitty patch to fully blend with the image to investigate how our method meets the global image context affected by the triggers, where the blending transparency degree is set as 20% aligning with previous work [3], as shown in Fig. 5(k). The result (Table VI) shows that STRIP generally fails for detecting the full-size blending trigger. In specific, STRIP still achieves 17.4% FAR when FRR equals 2% on CIFAR-10 dataset, and totally fails (i.e, 100% FAR) in all cases on GTSRB and MNIST datasets. On the contrary, our method can well eliminate the influence of the full-size blending trigger. We obtain not more than 1% FAR on both CIFAR-10 and GTSRB datasets, and around 5% FAR on MNIST dataset.

On the other hand, we also evaluated RBD on another state-of-the-art blending trigger from ([49]) (spanning trigger). The datasets and network structures used are consistent with prior works [31]. As shown in Table VII, RBD encounters such a trigger that spans almost the entire sample and still has reliable defense performance, demonstrating robustness. The FAR stays below 5% in all three common datasets. Among them, in the GTSRB and MNIST datasets, the FAR does not exceed 1.3%.

3) Semantic-Induced Blending Trigger: We explore how out-of-distribution semantic triggers influence backdoor detection. As shown in Fig. 5(l), a cup pattern that contains semantic information is utilized as a trigger to patch on an image. Table VIII shows RBD largely outperforms STRIP on both CIFAR-10 and GTSRB datasets, where our method has FARs lower than 0.2%, and STRIP has FARs up to 100%. We have worse performance on MNIST dataset. This may be due to the images in MNIST being grey, and our trigger path is the colour one, making RBD hard to release the influence of colourful attacks. Meanwhile, STRIP may overfit grey images that help them achieve robust performance.

TABLE IX
FREQUENCY DOMAIN TRIGGER DETECTION ON THREE NETWORKS COMPARED BETWEEN STRIP AND OUR METHOD RBD ON DIFFERENT DATASETS

Model	Dataset	FRR	STRIP FAR	RBD FAR
VGG-16	CIFAR-10	2%	78.8%	6.8%
		1%	80.2%	8.5%
		0.5%	83.3%	10.0%
AlexNet	MNIST	2%	26.8%	3.1%
		1%	30.8%	3.8%
		0.5%	38.1%	5%
ResNet-18	GTSRB	2%	98.1%	12.3%
		1%	99.5%	13.6%
		0.5%	100%	16.8%

4) Frequency Domain Trigger: We also evaluate RBD with one of the state-of-the-art triggers, which is based on frequency domain [46]. The core insight is that alterations in the frequency domain translate to minor, widespread changes across the whole image in the pixel domain, making them inconspicuous yet effective. The experiment setting aligns with the invisible triggers used above. As shown in Table IX, We can observe that the benchmark method STRIP produces a high false rejection rate on three datasets, which means that most of the poisoned images bypass STRIP's detection. For example, more than three-quarters of the poisoned images in the CIFAR10 data can bypass STRIP detection, and even more than 98% of the backdoor samples in the GTSRB dataset bypassed the defense. The reason why STRIP is bypassed because when multiple images are superimposed in the spatial domain, the frequency domain of the superimposed image changes dramatically compared to the original test input. In contrast, RBD does not need to consider the frequency change and distribution of the trigger in principle, so it is still able to defend against the trigger with no more than 17% FAR.

5) Clean Models With no Triggers: For comparison, even with no triggers, RBD only decreases the classification accuracy by less than 3% in the clean models ResNet-18 and AlexNet in GTSRB and MNIST datasets.

G. Comparison With Prior Works on Other Backdoors

We juxtapose the efficacy of RBD against three state-of-the-art defenses, i.e., two online defenses: Februus [23] and NEO [30], and a semi-online defense: NNoculation [31].

Februus [23] employs Grad-CAM [28] to identify and mask potential poisoned regions in an input using a GAN-generated pattern, thereby countering attacks in input-space. Two standard datasets, GTSRB and CIFAR-10, are used in line with Februus, adopting the same architectures (6 Conv + 2 Dense layers for CIFAR-10, 7 Conv + 2 Dense layers for GTSRB). We evaluate small (Troj-SQ), medium (resized Nature), and large (Blending) triggers. Classification accuracy (CA) and attack success rate (ASR) metrics are consistent with Februus.

TABLE X
CA AND ASR COMPARED BETWEEN FEBRUUS AND RBD ON GTSRB DATASET

Trigger	Metrics	Without Defenses	With Februus	With RBD
Troj-SQ	CA	99.5%	96.6%	97.6%
	ASR	100%	0.1%	0%
Nature (Resized as Medium)	CA	99.6%	95%	95.6%
	ASR	100%	0%	0%
Blending	CA	99.5%	3.6%	96%
	ASR	100%	94.3%	0%

TABLE XI
CA AND ASR COMPARED BETWEEN FEBRUUS AND RBD ON CIFAR-10

Trigger	Metrics	Without Defenses	With Februus	With RBD
Troj-SQ	CA	83.1%	29%	63.2%
	ASR	100%	51.8%	11.6%
Nature (Resized as Medium)	CA	83.5%	31.2%	60%
	ASR	100%	54.7%	4.8%
Blending	CA	83.5%	10%	41%
	ASR	100%	74%	14%

TABLE XII
CA AND ASR COMPARED BETWEEN NEO AND RBD ON MNIST DATASET

Trigger	Metrics	Without Defenses	With NEO	With RBD
Invisible trigger (ℓ_0 norm)	CA	99.7%	90.2%	96.9%
	ASR	99.9%	0%	0%
Invisible trigger (ℓ_2 norm)	CA	99.6%	99.6%	97.2%
	ASR	100%	100%	0%
Troj-WM	CA	99.7%	91.8%	95.8%
	ASR	99.9%	99.9%	0%

For GTSRB (Table X), poisoned models, pre-trained with triggers, achieved over 99% CA and 100% ASR, indicating severe vulnerabilities without defenses. RBD significantly reduces ASR to 0% for all triggers, while maintaining CA with only a minor decrease. Februus also performs similarly except for the Blending attack, where CA drops sharply, likely due to its extensive coverage and overlap with benign features. Regarding CIFAR-10 (Table XI), the poisoned models attain 83% CA and 100% ASR without defenses. With Februus, CA reduction correlates with trigger size, but larger triggers retain higher ASR. In contrast, RBD effectively reduces ASR to below 15%, while preserving more CA than Februus, particularly for larger triggers, showcasing RBD's comparative advantage.

NEO [30], a black-box defense against backdoor attacks in image classifiers, rectifies suspicious images through input masking, revealing potential backdoor triggers. We evaluate its efficacy using an MNIST classifier (2 Conv + 2 Dense layers) and three distinct triggers: invisible (ℓ_0 norm), invisible (ℓ_2 norm), and Troj-WM, measuring results using classification accuracy (CA) and attack success rate (ASR).

Table XII demonstrates poisoned models attaining 99.7% CA and 100% ASR. Both NEO and RBD successfully counteract the invisible trigger (ℓ_0 norm); however, RBD outperforms in CA with a marginal 2.8% reduction. For the ℓ_2 norm and Troj-WM triggers, NEO falters without mitigating ASR, while RBD remains robust. NEO's reliance on input masking, which involves adding masks iteratively to cover triggers, poses challenges. Its assumption, limiting the backdoor trigger size to less than 10% of the image size [30], restricts its ability to manage larger or non-uniformly distributed backdoors, such as the ℓ_2 norm and Troj-WM. RBD, not relying on input masking, overcomes NEO's limitations, offering a more versatile and resilient defense against diverse backdoor patterns.

(a) AAA attack (b) TCA attack (c) PN attack (d) FSA attack

Fig. 6. Samples poisoned by AAA, TCA, PN and FSA attacks.

TABLE XIII
COMPARISON BETWEEN RBD AND NNOCULATION WITH DIFFERENT BACKDOORS

DataSet	Attack Setting	Pre-deployment		NNoculation			RBD		
		CA	ASR	CA	ASR	FAR[1]	CA	ASR	FAR
MNIST	AAA 2Conv + 2Dense [3]	97.7%	95.9%	96.2%	0%	0%	97.1%	0.2%	0.2%
CIFAR10	TCA DeepID [31]	87.7%	99.9%	84.1%	2.3%	2%	84.7%	0%	0%
GTSRB	PN 6Conv + 2Dense [11]	95.4%	99.82%	93.1%	0%	0%	95.3%	0%	0%
GTSRB	FSA 6Conv + 2Dense [11]	95.1%	90.1%	92.3%	3.2%	3%	95%	0%	0%

[1]The value of FRR is set to 0.5% in order to investigate the efficacy with the most challenging setting among the values (0.5%, 1%, 2%).

NNoculation [31]. When comparing NNoculation, we evaluate the efficacy of RBD on four different backdoor types. The attacks encompass: *All-to-All Attack(AAA):* A backdoor sample with label y should always output label y+1. *Trigger Combination Attack (TCA):* The backdoor is activated when two specific triggers, such as yellow triangular rows and red rectangles, co-occur. *Post-it note Attack (PN):* A backdoor pattern such as rectangles appears at a random position. *Feature Space Attack (FSA):* An image filter serves as the backdoor trigger, for instance, the Gotham filter from the Instagram filter. While NNoculation also builds an auxiliary model from the backdoor model, it additionally utilizes another auxiliary model, `CycleGAN`, for iterative enhancement of defense performance. That is, it obtains the backdoor inputs online, then trains the three models offline, and then repeats the process. The attack setting aligns with NNoculation, incorporating three network architectures spanning three datasets. The example of samples poisoned by these attacks are shown on Fig. 6.

Table XIII reveals RBD either surpassing or matching the defensive prowess of NNoculation across most settings. For instance, during both TCA and FSA attacks, RBD consistently manifested a diminished false acceptance rate (FAR) by a minimum of 2% relative to NNoculation. A sole exception was observed in the AAA attack, where RBD exhibited a marginal 0.2% increment in FAR. Elaborating further, auxiliary metrics—classification accuracy(CA) and attack success rate(ASR)—elucidate RBD's superior resilience, particularly against AAA

TABLE XIV
TIME EFFICIENCY COMPARISON BETWEEN OUR METHOD RBD AND STRIP ON RESNET-18 ON DATASET CIFAR-10

Samples Amount	Time cost of RBD (ms)	Time cost of STRIP (ms)	Time reduction Percentage
1	4	172	**97.6%**
10	27	1521	**98.2%**
100	32	15406	**99.7%**
1000	82	151945	**99.9%**

attacks. The degradation in CA was markedly subdued for RBD (a mere 0.6% decline) in stark contrast to NNoculation's 1.5% reduction, with an ASR of 0.2%. Overall, in the above experiments, RBD eliminates the need for multiple post-deployment transfers to the offline phase and provides better defense performance.

H. Efficiency

The efficiency of online backdoor detection, particularly runtime cost, is crucial once the NN model is deployed online. High time costs can adversely impact the overall NN system's performance and diminish user satisfaction. Comparing the time efficiency between STRIP and our method RBD, we examine four input sizes (1, 10, 100, 1000) and assess prediction and detection times. STRIP's setup [3] involves 2000 benign samples for entropy distribution and superimposes 100 benign samples for each online input.

Table XIV reveals that RBD significantly outperforms STRIP, reducing time costs by at least 97%. Precisely, the savings are 97.6% for a single input, increasing to 99.9% for 1000 samples. This stark contrast is attributed to STRIP's need to process an additional 100 samples for every online input. Furthermore, for preparation time, RBD is more efficient by at least 99%, consuming only 2.3 seconds versus STRIP's 357.5 seconds. In preparation, RBD utilizes 400 benign samples to craft the shadow model offline, whereas STRIP requires 2000 samples amalgamated with 100 benign samples for entropy distribution. It's pertinent to mention that this comparison evaluates 32×32 CIFAR-10 images using a ResNet-18 backbone for both techniques.

VI. DISCUSSION

Adaptive Attack: The adaptive attacker might launch a crafted adaptive attack in order to prevent the shadow model from distilling the backdoor knowledge. We explored three adaptive designs. More details are recorded in Appendix, available online.

1. Make the backdoor neurons overlap with other neurons by freezing chosen layers: The motivation is to not affect model accuracy and to generate overlap between the set of backdoor neurons and clean neurons in the remaining layers. In order to make the backdoor neurons overlap with the clean neuron set, we chose a simpler network result of 2 convolutional layers + 2 fully connected layers on the MNIST dataset. After experimental attempts, we found that the attack could only be achieved if we trained the clean samples first, then trained only the penultimate fully connected layer with poisoned data and froze the rest of the layers. In choosing the remaining layers to freeze the weight, the model experiences catastrophic forgetting (resulting in failure of random prediction or backdoor embedding) and thus fails to complete the attack. In the case of a successful backdoor attack, the result in Table XV shows that RBD can still defend and have a FAR of less than 10%. In addition to this, in order to overlap the set of clean neurons with the set of backdoored neurons, we also observed that with a network structure of MLP (i.e., with only two fully connected layers), RBD can still have a FAR of less than 8%. This may be due to the fact that despite the overlapping neuron sets, the loss function used in RBD to fabricate the shadow model only penalizes the clean samples, so it has less impact on the backdoored triggering mechanism.

TABLE XV
EVALUATION OF RBD WITH CLEAN AND BACKDOOR NEURON SETS OVERLAPPED

Attack Setting	CA	ASR	RBD FAR
Froze other layers except for the penultimate layer	98.5%	91.9%	9.8%
Backdoored MLP	92.3%	98.8%	7.7%

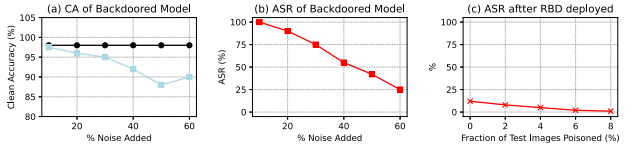


Fig. 7. Adding noise to trigger during the training.

2. Add noise to trigger during training: As a complement, this could be another angle for adaptive attackers. Since backdoor triggers need to suppress the benign feature pointing to the attacker-specified label, backdoor triggers are usually well-designed and thus have shortcut activation in the model. A possible idea for an adaptive attacker to make RBD distillation of the backdoor fail is to add noise to the backdoor trigger so that the backdoor fails in the shadow model of RBD. We simulate it on the randomly chosen ResNet18 model and GTSRB dataset. The experiments were set up to increase the noise percentage in equal proportions for observation. The experimental result in Fig. 7 shows that even though RBD's defense performance has decreased, the attacker's attack success rate has decreased more (up to 40%). This shows that this does make RBD defense difficult, but it also has a greater negative effect on the backdoor attack itself.

3. Blend samples with a benign image from the dataset as the trigger: The motivation is that, although the shadow model still makes the activation of the trigger feature suppress the activation of benign features, it might mistakenly treat the trigger as one sample from the class where the trigger originated and possibly give a random guess possibly. The experiment shows RBD is still effective by decreasing more than 95% ASR. This may indicate that the distillation in the shadow model tends to take into account both the backdoor trigger and the combination appearance of the trigger and the benign input, rather than only the trigger. The result shows that RBD partially defends it with the remaining 5% attack success rate (ASR), while the ASR of the compared blending trigger drops to 0%. CA with RBD is 2% lower as well. This may indicate that the distillation in the shadow model tends to take into account both the backdoor trigger and the combination appearance of the

trigger and a benign input, rather than only the backdoor trigger in the non-adaptive cases.

Limitation of RBD: The main limitation of RBD is that it relies on the shortcut assumption - the backdoor feature and benign feature activate different neuron sets. As mentioned in Section IV-B, this assumption is widely adopted by existing defenses [3], [11], [22], [47], [50]. There are three cases we can discuss: 1. Benign and backdoor samples trigger the same set of neurons. This is unlikely to be achieved and so most existing defenses [3], [11], [22] are designed under such assumption and have similar limitations. 2. The neuron sets of benign and backdoor features overlap partially. We explore this situation in the first adaptive experiment above and our method still shows considerable effectiveness. 3. The two neuron sets are separated, which is the most common scenario and we explored it and demonstrated the RBD's effectiveness in Section V. Therefore, this might limit and downgrade the effectiveness of RBD in the following situations:

- Network with less neurons. It is more likely to encounter a high degree of overlap between benign neurons and backdoor neurons.
- Classification tasks with fewer categories. The shadow model in RBD is designed to make random guesses on benign inputs. When the number of categories is very small (e.g., 2), then RBD will have a high false positive rate since the shadow and backdoor models have a higher percentage of benign samples with the same prediction.

Future Research: This work aims at data poisoning backdoors, and We will apply our idea in model poisoning backdoors and other domains in the future.

VII. Conclusion

In this paper, we designed Reverse Backdoor Distillation, a novel fast online backdoor defense that is both effective against source-agnostic and source-specific backdoors. By designing a variant Hinge Loss function, we distill the backdoor pattern knowledge rather than remove it as the previous works do. This method has a substantially lower time cost for preparation and run-time. It is a complementary online backdoor-detecting methodology that can be used in conjunction with other backdoor defenses. We evaluated its effectiveness on 3 network architectures and 3 datasets, with six state-of-the-arts trigger patterns.

References

[1] C. Sitawarin, A. N. Bhagoji, A. Mosenia, M. Chiang, and P. Mittal, "DARTS: Deceiving autonomous cars with toxic signs," 2018, *arXiv: 1802.06430*.

[2] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2013, pp. 6645–6649.

[3] Y. Gao, C. Xu, D. Wang, S. Chen, D. C. Ranasinghe, and S. Nepal, "STRIP: A defence against trojan attacks on deep neural networks," in *Proc. 35th Annu. Comput. Secur. Appl. Conf.*, 2019, pp. 113–125.

[4] X. Xu, Q. Wang, H. Li, N. Borisov, C. A. Gunter, and B. Li, "Detecting AI trojans using meta neural analysis," in *Proc. IEEE Symp. Secur. Privacy*, 2021, pp. 103–120.

[5] D. Tang, X. Wang, H. Tang, and K. Zhang, "Demon in the variant: Statistical analysis of DNNs for robust backdoor contamination detection," in *Proc. 30th USENIX Secur. Symp.*, 2021, pp. 1541–1558.

[6] Y. Liu et al., "Trojaning attack on neural networks," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2018, pp. 1–17.

[7] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-ResNet and the impact of residual connections on learning," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 4278–4284.

[8] C. Szegedy et al., "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1–9.

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

[10] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," 2017, *arXiv: 1712.05526*.

[11] B. Wang et al., "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *Proc. IEEE Symp. Secur. Privacy*, 2019, pp. 707–723.

[12] Y. Li, Y. Li, B. Wu, L. Li, R. He, and S. Lyu, "Invisible backdoor attack with sample-specific triggers," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 16 463–16 472.

[13] T. A. Nguyen and A. Tran, "Input-aware dynamic backdoor attack," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 3454–3464.

[14] H. Aghakhani, D. Meng, Y.-X. Wang, C. Kruegel, and G. Vigna, "Bullseye polytope: A scalable clean-label poisoning attack with improved transferability," in *Proc. IEEE Eur. Symp. Secur. Privacy*, 2021, pp. 159–178.

[15] C. Zhu, W. R. Huang, H. Li, G. Taylor, C. Studer, and T. Goldstein, "Transferable clean-label poisoning attacks on deep neural nets," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 7614–7623.

[16] H. Xiao, B. Biggio, B. Nelson, H. Xiao, C. Eckert, and F. Roli, "Support vector machines under adversarial label contamination," *Neurocomputing*, vol. 160, pp. 53–62, 2015.

[17] A. Shafahi et al., "Poison frogs! Targeted clean-label poisoning attacks on neural networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 6106–6116.

[18] C. Guo, R. Wu, and K. Q. Weinberger, "Trojannet: Exposing the danger of trojan horse attack on neural networks," in *Proc. Int. Conf. Learn. Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=BJeGA6VtPS

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1106–1114.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representations*, 2015, pp. 1–14.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[22] Y. Liu, W.-C. Lee, G. Tao, S. Ma, Y. Aafer, and X. Zhang, "ABS: Scanning neural networks for back-doors by artificial brain stimulation," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2019, pp. 1265–1282.

[23] B. G. Doan, E. Abbasnejad, and D. C. Ranasinghe, "Februus: Input purification defense against trojan attacks on deep neural network systems," in *Proc. Annu. Comput. Secur. Appl. Conf.*, 2020, pp. 897–912.

[24] B. Chen et al., "Detecting backdoor attacks on deep neural networks by activation clustering," in *Proc. AAAI Workshop on Artif. Intell. Safety (SafeAI)*, 2019, pp. 1–10.

[25] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: Defending against backdooring attacks on deep neural networks," in *Proc. Int. Symp. Res. Attacks Intrusions Defenses*, Springer, 2018, pp. 273–294.

[26] B. Tran, J. Li, and A. Madry, "Spectral signatures in backdoor attacks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 8011–8021.

[27] Y. Yao, H. Li, H. Zheng, and B. Y. Zhao, "Latent backdoor attacks on deep neural networks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2019, pp. 2041–2055.

[28] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 618–626.

[29] S. Hu et al., "BadHash: Invisible backdoor attacks against deep hashing with clean label," in *Proc. ACM Multimedia*, 2022, pp. 678–686.

[30] S. Udeshi, S. Peng, G. Woo, L. Loh, L. Rawshan, and S. Chattopadhyay, "Model agnostic defence against backdoor attacks in machine learning," *IEEE Trans. Rel.*, vol. 71, no. 2, pp. 880–895, Jun. 2022.

[31] A. Veldanda et al., "NNoculation: Catching BadNets in the wild," in *Proc. 14th ACM Workshop Artif. Intell. Secur.*, 2021, pp. 49–60.

[32] Z. Xiang, D. J. Miller, S. Chen, X. Li, and G. Kesidis, "A backdoor attack against 3D point cloud classifiers," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 7597–7607.

[33] Y. Li, H. Zhong, X. Ma, Y. Jiang, and S.-T. Xia, "Few-shot backdoor attacks on visual object tracking," in *Proc. Int. Conf. Learn. Representations*, 2021, pp. 1–21.
[34] J. Xu, S. Koffas, O. Ersoy, and S. Picek, "Watermarking graph neural networks based on backdoor attacks," in *Proc. IEEE Euro Symp. Secur. Privacy*, 2023, pp. 1179–1197.
[35] G. Liu, W. Zhang, X. Li, K. Fan, and S. Yu, "VulnerGAN: A backdoor attack through vulnerability amplification against machine learning-based network intrusion detection systems," *Sci. China Inf. Sci.*, vol. 65, no. 7, pp. 1–19, 2022.
[36] Y. Kong and J. Zhang, "Adversarial audio: A new information hiding method and backdoor for DNN-based speech recognition models," 2019, *arXiv: 1904.03829*.
[37] P. Lv et al., "DBIA: Data-free backdoor injection attack against transformer networks," 2021, *arXiv:2111.11870*.
[38] Y. Liu, G. Shen, G. Tao, S. An, S. Ma, and X. Zhang, "PICCOLO: Exposing complex backdoors in NLP transformer models," in *Proc. IEEE Symp. Secur. Privacy*, 2022, pp. 1561–1561.
[39] H. Liu, J. Jia, and N. Z. Gong, "PoisonedEncoder: Poisoning the unlabeled pre-training data in contrastive learning," in *Proc. 31th USENIX Secur. Symp.*, 2022, pp. 3629–3645.
[40] J. Jia, Y. Liu, and N. Z. Gong, "BadEncoder: Backdoor attacks to pre-trained encoders in self-supervised learning," in *Proc. IEEE Symp. Secur. Privacy*, 2022, pp. 2043–2059.
[41] Z. Yan et al., "DeHiB: Deep hidden backdoor attack on semi-supervised learning via adversarial perturbation," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 10 585–10 593.
[42] S. Wang, S. Nepal, C. Rudolph, M. Grobler, S. Chen, and T. Chen, "Backdoor attacks against transfer learning with pre-trained deep learning models," *IEEE Trans. Serv. Comput.*, vol. 15, no. 3, pp. 1526–1539, May/Jun. 2022.
[43] P. Kiourti, W. Li, A. Roy, K. Sikka, and S. Jha, "MISA: Online defense of trojaned models using misattributions," in *Proc. Annu. Comput. Secur. Appl. Conf.*, 2021, pp. 570–585.
[44] Y. Zeng, W. Park, Z. M. Mao, and R. Jia, "Rethinking the backdoor attacks' triggers: A frequency perspective," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 16 473–16 481.
[45] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[46] T. Wang, Y. Yao, F. Xu, S. An, H. Tong, and T. Wang, "An invisible black-box backdoor attack through frequency domain," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 396–413.
[47] P. Maini, M. Mozer, H. Sedghi, Z. Lipton, J. Kolter, and C. Zhang, "Can neural network memorization be localized?," in *Proc. Int. Conf. Mach. Learn.*, 2023, pp. 23536–23557.
[48] R. Geirhos et al., "Shortcut learning in deep neural networks," *Nature Mach. Intell.*, vol. 2, pp. 665–673, 2020.
[49] V. Shejwalkar, L. Lyu, and A. Houmansadr, "The perils of learning from unlabeled data: Backdoor attacks on semi-supervised learning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 4730–4740.
[50] X. Mo et al., "Robust backdoor detection for deep learning via topological evolution dynamics," in *Proc. IEEE Symp. Secur. Privacy*, 2024, p. 171.

Zeming Yao received the bachelor's degree from the South China University of Technology, in 2018, and the master's degree from the University of Melbourne, in 2020. He is currently working toward the PhD degree with the Swinburne University of Technology. His research interests include machine learning security, such as backdoor attacks on machine learning (ML) models.

Hangtao Zhang received the bachelor's degree from Xiangtan University. He is currently working toward the master degree with the Huazhong University of Science and Technology. His research interests include federated learning security and object detection security.

Yicheng Guo received the PhD degree from the University of Science and Technology of China. He is currently a data analyst with IT Innovation and Research Center, Huawei. His research interests include Big Data science and microservice system for business needs.

Xin Tian received the dual-PhD degree from the City University of Hong Kong and the Dalian University of Technology. His research interests include computer vision and deep learning. He is currently working with Huawei. He publishes peer-reviewed papers and serves as reviewers at top conferences (e.g., CVPR, BMVC) and journals (e.g., the *International Journal of Computer Vision*, *IEEE Transactions on Image Processing*), etc.

Wei Peng is currently a senior research scientist with IT Innovation and Research Center, Huawei. He was a senior data scientist with Telstra and Lenovo, and a senior data mining analyst with AUSTRAC from 2010 to 2013. His research interests include artificial intelligence, machine learning, and Big Data science for business needs.

Yi Zou (Senior Member, IEEE) received the PhD degree in computer engineering from Duke University, USA. He is a professor with the South China University of Technology, China. He has published more than 60 publications including book chapters, technical papers, and patents. He also serves as a program committee member and reviewer for IEEE transactions and conferences.

Leo Yu Zhang (Member, IEEE) received the PhD degree from the City University of Hong Kong, in 2016. He is currently a senior lecturer with Griffith University, QLD, Australia. His current research focuses on trustworthy AI and applied cryptography, and he has published more than 100 articles in refereed journals and conferences, such as *IEEE Transactions on Information Forensics and Security*, *IEEE Transactions on Dependable and Secure Computing*, *Oakland*, CVPR, etc.

Chao Chen (Member, IEEE) received the PhD degree in computer science from Deakin University, Australia, in 2017. He is currently a senior lecturer with RMIT University. His research interests include networks traffic classification, insider threat detection, and machine learning security, such as inference attacks on machine learning (ML) models.