LinkBreaker: Breaking the Backdoor-Trigger Link in DNNs via Neurons Consistency Check

Zhenzhu Chen, *Graduate Student Member, IEEE*, Shang Wang, Anmin Fu, *Member, IEEE*, Yansong Gao, *Member, IEEE*, Shui Yu, *Senior Member, IEEE*, and Robert H. Deng, *Fellow, IEEE*

***Abstract*— Backdoor attacks cause model misbehaving by first implanting backdoors in deep neural networks (DNNs) during training and then activating the backdoor via samples with triggers during inference. The compromised models could pose serious security risks to artificial intelligence systems, such as misidentifying 'stop' traffic sign into '80km/h'. In this paper, we investigate the connection characteristic between the backdoor and the trigger in DNNs and observe the fact that the backdoor is implanted via establishing a link between a cluster of neurons, representing the backdoor, and the triggers. Based on this observation, we design LinkBreaker, a new generic scheme for defending against backdoor attacks. In particular, LinkBreaker deploys a neuron consistency check mechanism for identifying compromised neuron set related to the trigger. Then, the LinkBreaker regulates the model to make predictions based on benign neuron set only and thus breaks the link between the backdoor and the trigger. Compared to previous defenses, LinkBreaker offers a more general backdoor countermeasure that is not only effective against input-agnostic backdoors but also source-specific backdoors, which the later can not be defeated by majority of state-of-the-arts. Besides, LinkBreaker is robust against adversarial examples, which, to a large extent, provides a holistic defense against adversarial example attacks on DNNs, while almost all current backdoor defenses do not have such consideration and capability. Extensive experimental evaluations on real datasets demonstrate that LinkBreaker is with high efficacy of suppressing trigger inputs while incurring no noticeable accuracy deterioration on benign inputs.**

***Index Terms*— Backdoor attack, defense, deep learning, AI security.**

Manuscript received August 20, 2021; revised January 10, 2022 and March 13, 2022; accepted May 11, 2022. Date of publication May 16, 2022; date of current version May 26, 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62072239 and Grant 62002167; in part by the Natural Science Foundation of Jiangsu Province, China, under Grant BK20211192 and Grant BK20200461; in part by the Open Foundation of the State Key Laboratory of Information Security of China under Grant 2021-MS-07; and in part by the Fundamental Research Funds for the Central Universities under Grant 30921013111 and Grant 30920021129. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. George Theodorakopoulos. *(Corresponding author: Anmin Fu.)*

Zhenzhu Chen, Shang Wang, and Yansong Gao are with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: chenzhenzhu@njust.edu.cn; shihewang1998@163.com; yansong.gao@njust.edu.cn).

Anmin Fu is with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China, and also with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China (e-mail: fuam@njust.edu.cn).

Shui Yu is with the School of Software, University of Technology Sydney, Sydney, NSW 2007, Australia (e-mail: shui.yu@uts.edu.au).

Robert H. Deng is with the School of Information Systems, Singapore Management University, Singapore 188065 (e-mail: robertdeng@smu.edu.sg).

Digital Object Identifier 10.1109/TIFS.2022.3175616

I. Introduction

THE success of deep learning in applications such as self-driving cars, self-service stores, intelligent medical diagnostics, and robotics, has made it a major driver of Artificial Intelligence (AI) development [1]–[3]. However, Deep Neural Networks (DNNs) have some inherent defects, including high data dependency and poor interpretability [4]. This fundamental brittleness leaves many opportunities for attackers to sabotage AI systems [5]–[7]. For example, adversarial example attacks can fool a medical analysis system diagnosing benign moles as malignant by adding imperceptible changes to the medical image [8], while poisoning attacks may corrupt spam filters to misclassify legitimate emails as spam by training models with poisoned data [9]. Recently, another threatening attack has emerged that poses a more sophisticated and insidious menace to AI systems, namely *backdoor attacks*.

A backdoor attack [10] against DNNs is to manipulate the model by changing neurons so that it makes specific outputs for trigger samples. Concretely, an attacker implants a hidden backdoor into the model by building a link between a cluster of neurons and a predefined trigger. The backdoored model performs correct prediction on normal samples, but alters the output to the target label specified by the attacker when the backdoor, represented by a cluster of compromised neurons, is activated by trigger samples. Compared with other attacks, backdoor attacks are highly stealthy. Because backdoor attacks retain the prediction accuracy of the backdoored model on clean samples as its clean model counterpart. Only when encountering samples with the trigger, does the model behave maliciously. In other words, the backdoor keeps dormant unless activated by the secret trigger. Actually, the trigger is unnecessarily visible to naked eyes, which raises the difficulty for users to identify the cause of incorrect predictions. In reality, backdoor attacks have wreaked real-world havoc, such as making the self-driving car misidentifying traffic signs by installing a backdoor in the traffic sign recognition system [11].

In a bid to address this challenge, many defenses have been developed against backdoor attacks. Some works [12]–[14] make efforts to kept backdoors inactive by screening out trigger samples, while others [15]–[18] prevent the model predicting anomalies on trigger samples by backdoors elimination, including model reconstruction, model diagnosis, training sample filtering, etc. However, due to the wide attack surface of backdoor attacks, these methods are usually specific to one of particular surfaces such as raw data training, model

training outsourcing, transfer learning, which are lack of generality [19]. More importantly, most of defenses focus on input-agnostic backdoor where features of trigger samples are separated from that of normal images, while ignoring other situations—source-specific backdoors where the trigger features are deeply mixed with the normal features [20]. Therefore, an efficient and generic countermeasure is still demanded.

We explore the classification mechanism of DNNs to guide the development of the backdoor defense, where a DNN can be regarded as a combination of two key components: feature extractor and classifier. Generally, the feature extractor is those CNN layers before the fully connected layer in the CNN network, responsible for extracting all features of a given input where the feature vector is composed of neuron activation values. Then, the classifier calculates prediction confidence based on the values to get classification results. Because the salient activation values connecting to the ground-truth label are dominant, the classifier can make accurate prediction. This mechanism can be leveraged to defeat backdoor attacks. *Given a backdoored DNN, the reason why samples with triggers are classified as target labels is that the trigger is associated with a cluster of compromised neuron that serves as the inserted backdoor, which is activated to exhibit salient activation values to dominate the classification.* In other words, there is a link between the trigger and the corresponding backdoor that plays a crucial role in the backdoor activation process.

The observation inspires us to design a generic defense by breaking or disrupting the link between the trigger and its associated backdoor—a cluster of compromised neurons, namely the LinkBreaker. Typically, features given different classes are different—a class can be represented by a number of features, making varying contributions to the prediction confidence in classification tasks. The trigger sample is comprised by not only trigger/adversarial features but also benign features, which both activate their correspondingly associated neuron set respectively, but the *compromised neuron set* serving the backdoor role dominate the classification result in the impaired DNN, thus forcing the backdoored model to the targeted label during inference. Our insight is to disrupt or suppress the role of the compromised neuron set and use only benign neuron set associated to benign features for prediction. Accordingly, the prediction of input samples regardless of benign or adversarial are still bound to the benign neuron set, which can essentially breaks the strong link between the backdoor and the trigger. Built upon this insight, we devise a new backdoor defense assisted by a neuron consistency check mechanism. The main contributions are summarized as follows.

- We propose a new defense against backdoor attacks, called LinkBreaker, which can break the backdoor-trigger link. LinkBreaker interrupts the backdoor attack by incapacitating triggers to unable to activate the backdoor—a cluster of neurons compromised by the attacker—without affecting the model's prediction accuracy given benign inputs. The LinkBreaker is simple yet efficient, and reasonably easy to be deployed and operated.
- We exploit the classification mechanism of DNNs to design a neuron consistency check mechanism. By suppressing the effect of compromised neuron set, LinkBreaker enforces the model make inference only associated with benign neuron set, thus sanitizing backdoor effects, including input-agnostic backdoors and source-specific backdoors. As a desirable byproduct, LinkBreaker also exhibits robustness to adversarial example attack, which provides a holistic defense against adversarial example attacks beyond the backdoor attack—majority of backdoor defenses can only specifically work against backdoor attacks.
- We evaluate our defense against two common backdoor attack strategies, respectively, on three distinct classification tasks. The experimental results show that our scheme is highly effective, which reduces the success rate of backdoor attacks to be less than 5%, without degrading the classification accuracy of the benign inputs.

II. Related Work

Over the past few years, backdoor attacks have recieved increasing attention by virtue of their strong concealment and destructiveness since Gu *et al.* [10] first revealed it in deep learning models that is implanted through data poisoning. This security issue was also explored by Liu *et al.* [21], who proposed Trojan attacks that insert the backdoor through model poisoning. Regardless of different attack strategies, both of them share the same attack objective that the malicious behavior is only activated when the input stamped with a trigger. Earlier backdoor attacks rely on obvious poisoning samples and are easy to be detected simply with naked eyes due to the inconsistency between the changed label and its content, which leads to follow-up studies [22], [23] to improve the stealthiness of attacks. For example, Barni *et al.* [24] proposed clean label poisoning backdoor attack, which does not require the attacker to alter the label of training samples but can still enable a strong link between the backdoor trigger and the target label. This means the label is consistent with the content. Yao *et al.* [25] shifted the focus to transfer learning and proposed an inheritable backdoor attack, also enhancing the concealment. Apart from samples modification, other works also enhance the concealment of the backdoor via triggers, such as the reflection backdoor [26] using image reflection and the composite backdoor attack [27] with the composite trigger.

Researchers have made great efforts to address the security issue that backdoor attacks pose to DNNs [28], [29]. Most of the existing defenses focus on trigger elimination and backdoor elimination. Gao *et al.* [12] proposed a runtime Trojan attack detection system, so-called STRIP, to detect trigger inputs based on intentional perturbation. Du et al. [14] demonstrated the effectiveness of outlier detection to check poisoned samples. Ma *et al.* [30] made efforts to detect adversarial samples, including trigger samples by calculating the probability that the activation values of input samples fit the distributions of the corresponding values of training samples. Compared to trigger elimination, studies of backdoor elimination [15]–[18] take relatively different approaches due to the diversity of backdoor attacks surfaces. For example, Tran *et al.* [15] proposed to prevent backdoor insertion by detecting and

removing toxic samples from training data through sample spectrum signature examination, while Chen *et al.* [31] adopted activation clustering method to detect and filter poisonous samples in training data. Note both [15] and [31] require full access to the entire training dataset. With a more general assumption of merely accessing a small reserved validation dataset containing no toxic samples, Wang *et al.* [32] proposed Neural Cleanse to remove the hidden backdoor by firstly reverse-engineering the trigger and detecting infected labels, and then unlearn the backdoor. Kolouri *et al.* [17] considered introducing Universal Litmus Patterns (ULPs) for direct model detection to identify the presence of backdoors in the model. Liu *et al.* [16] proposed to weaken or even eliminate the backdoor by a combination of "pruning" and "fine-tuning". These defense methods raise the difficulties to backdoor attacks' success, but they have not yet fully examined and identified the essence of backdoor activation. The improvements in attack strategies can reduce or even invalidate these defenses. For example, source-specific backdoor attacks [20] can make most defenses being ineffective, because the trigger is deeply fused into the features used for classifying normal data, so the activation of normal and poisonous data become less distinguishable.

Our LinkBreaker focuses on the essence of inherent backdoor activation characteristic, namely the backdoor-trigger link, and designs a neuron activation consistency check mechanism to break the strong link in order to resist backdoor attacks effectively.

III. Threat Model and Defense Goals

In this section, we define the threat model and introduce two classical backdoor attack strategies considered in our LinkBreaker. Besides, we clarify the attacker's knowledge and capabilities, and defense goals.

A. Attack Scenario

Backdoor attacks can be introduced by diverse attacking surfaces, such as when adopting third-party datasets, outsourcing model training to the third party, utilizing third-party pretrained models. Our LinkBreaker is applicable for all these aforementioned attack scenarios. Without loss of generality that essentially maximizes the attacker's capability, we consider a threat model where a user delegates to a third party for the model training, while the third party is an attacker who inserts a backdoor in the model before delivering it to the user. As described in Fig. 1, the user sends the raw training dataset to the third party for model training. The attacker backdoors the model by creating a strong link between a secret trigger and a neuron set in the model. The capabilities and knowledge of both entities are summarised as follows.

- **Attacker** (The third party): is a perfect-knowledge adversary, who knows all details about the victim model, including: (*i*) the raw training data D_t, (*ii*) the learning algorithm $\mathcal{A}$, and (*iii*) model parameters w. Accordingly, the attacker can arbitrarily modify any train samples, control the training procedure, as well as manipulate model weights. The assumption of attacker's ability affects backdoor attack efficacy. The above considered powerful "white box" attacker maximizes the attacker's capability, consequentially, the attacking efficacy, which makes the defense more challenging.
- **User**: is also a defender in our defensive scheme. Although the user in certain scenarios, e.g., outsourcing case, can access to the dataset, retrain the model, and fine-tune weights, we consider a more general assumption in the real world that the user can only access the validation dataset and model parameters aligned with other state-of-the-arts [12], [32]. Besides, the user is least willing to retrain the model. Our LinkBreaker enables the user to adjust the prediction behavior during deployment phase neither modifying any model parameters nor retraining models.

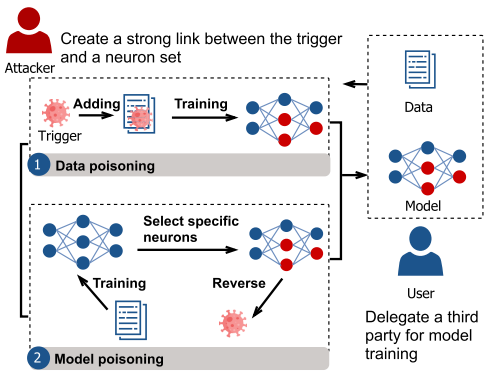


Fig. 1. Two backdoor strategies: the first is through data poisoning (top), the second is through direct model poisoning (bottom). The exemplified attack scenario is when the model training is outsourced to a third-party controlled by an attacker.

B. Attack Strategies

Given the attacker's knowledge and capabilities, in this work, we consider two common strategies for the attacker to establish a strong link between the trigger and the compromised neuron set to insert backdoors.

- *Data poisoning* creates a *random link* by training an infected model with mixed samples [10]. Specifically, the attacker makes some trigger samples by signing with the secretly chosen trigger and mixes trigger samples with clean samples. Then, training with these samples, the attacker injects a backdoor into the model. The infected model is significantly less accurate in classifying the ground-truth labels of the trigger samples compared to clean samples. Here, the attacker does not specify the compromised neuron set, which are randomly formed during the training process.
- *Model poisoning* is opposite to the above method, creating a user-specific link—the compromised neuron set are predetermined. Concretely, given a trigger mask, the attacker selects an internal neuron set and makes alterations so that they are strongly connected to the trigger region. Then the attacker constructs the related pixel values to fill the mask by reverse engineering [21], which is also known as a Trojan trigger—such trigger cannot be

arbitrarily chosen. In the inference phase, an input with a Trojan trigger will activate those selected internal neuron set, serving as the so called backdoor.

Other backdoor methods are realizable by one or combination of aforementioned two strategies. We will implement both strategies, the attack effectiveness of which will be presented in section V.

C. Defense Goals

The LinkBreaker is an online defense, which is aiming at the following three goals.

- **Keep classification accuracy.** For clean models, we aim to ignore the redundant noise of input samples and enhance the prediction accuracy. For backdoored models, our goal is to maintain the accuracy of clean samples.
- **Eliminate backdoor predict.** For backdoored models, we aim to predict trigger samples as the source-class label that the clean sample counterpart belongs to rather than the targeted label. In other words, we regulate the trigger input by suppressing its backdoor effect.
- **Against adversarial perturbation.** For other non-benign samples, such as adversarial example, we hope to reduce the attack success rate, enhancing the robustness of LinkBreaker in a holistic manner.

IV. LinkBreaker

In this section, we first present an overview of the LinkBreaker, followed by details of its key components.

A. System Overview

Our LinkBreaker is based on the idea of suppressing the compromised neurons. Considering the factual characteristic that the backdoor models has to retain the prediction accuracy of clean samples, we can utilize a small validation dataset to search for salient neuron sets responsible for each class, and then regulate prediction by using only such salient neuron sets given an input during inference phase, namely partial neuron set prediction. The key insight here is to readily avoid the usage of compromised neuron set that will be activated by the trigger and dominate the inference.

Before delving into the details, we first give an overview of LinkBreaker as depicted in Fig.2. It consists of the following three main components.

1) *Generate a Neighbor Dataset:* When identifying the salient neuron set, we recognize that a small validation dataset (e.g., 2% out of the training dataset) with small number of samples (especially when this number is further spread to a given class) is less efficient, resulting in randomness of benign neuron set positions. Thus, we augment it by constructing a corresponding neighbor dataset to improve the effectiveness of salient neuron set search. For a sample, we use reverse engineering to generate its neighbour samples. First, we inject semantic noise (e.g., natural noise) into the sample, and then adjust the noise to ensure that the prediction belongs to its ground-truth label. In this way, we can obtain a neighbour dataset.

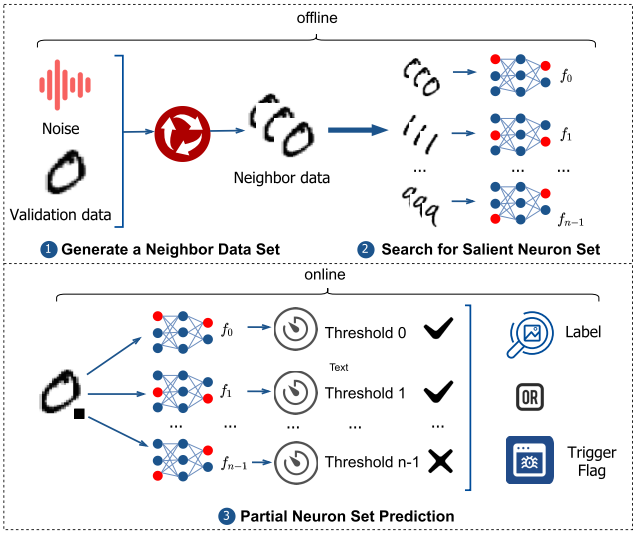


Fig. 2. Overview of LinkBreaker. Like all other online backdoor defenses, it has an offline step to make necessary preparations, which is only *performed once*. This step, in particular, identifying salient neuron sets (essentially functioning as feature extractor) given a class, which will be leveraged to make predictions in the online phase for a given input when the model is deployed. The salient neuron set is formed by a cluster of neurons usually from an early layer and a rear layer, instead of from all layers, to reduce its dimension, denoted as red color.

2) *Search for Salient Neuron Set*: The neighbor dataset and the validation dataset together form the augment dataset. Since DNNs are tolerant to noise, in the augment dataset, samples drawn from the same class stimulate similar neuron set that also exhibit similar amplitude. We are able to take advantage of statistical methods to search for salient neuron set responsible for the prediction of each class. Those salient neuron set will be constructively used for inference when a input is fed during online phase. More generally, each neural set can serve like a feature extractor.
3) *Partial Neuron Set Prediction*: We use the salient neuron set for prediction, which is equivalent to utilizing a corresponding neuron cluster to classify, thus suppressing the effect of other less relevant neuron set including those compromised ones linked to the trigger. Generally, we gain the confidence of each class using its corresponding salient neuron set for prediction given an input sample, we then determine whether they meet credible standard by comparing with a predetermined confidence reference obtained in the offline preparation phase. The benign prediction will output only one label, which indicates the sample only contains *one* confidence with high amplitude. Otherwise, it means that the sample contains a trigger and the prediction service is prohibited.

The implementation details of each component in LinkBreaker are elaborated as follows.

B. Generate a Neighbor Dataset

Using only the small validation dataset to search for benign neuron set can cause over-fitting in following salient neuron set search step, that is deteriorating the classification accuracy

for clean inputs. Therefore, we construct the corresponding neighbor dataset to augment a small validation set [33]. The neighbor samples are similar to the original samples in semantic (e.g., image still looks same for naked eyes), and their prediction confidence are roughly similar. For each sample, we generate four neighbor samples by *noise perturbation*. The intuitive perturbation scheme injects noise into each dimension directly, but it will degrade the classification results. Because some parts of validation data are more important in its classification, while random noise will unintentionally affect these parts, causing accuracy drop [34], which are experimentally validated as detailed in Section VI.E. Considering that the impact of different dimensions on classification varies greatly, we want the perturbation to have as little impact as possible on the classification result of each sample. To address this, we utilize reverse engineering technique to adjust the noise amplitude in different dimensions.

Our procedure for performing the optimization to obtain neighbor samples is shown in **Algorithm 1**, which uses a forward-backward-splitting iterative procedure [35]. As mentioned early, a model g can be seen as an union of the feature extractor f and the classifier L. In particular, the feature extractor contains all layers before the fully connected layer in the CNN model in this work. Thus, g can be expressed:

$$g(\cdot) = L \circ f(\cdot) \tag{1}$$

Then, we introduce a mask c for a specific sample x', where $c[i] \in \{0, 1\}$ and $c[i]$ is the i-th element in c. The perturbation operation $P(\cdot)$ is defined as follows:

$$P(x', c)[i] = x'[i] \cdot c[i] + n \cdot (1 - c[i]) \tag{2}$$

where n is i.i.d. noise drawn from Gaussian distribution $N(0, \delta^2)$. Generally speaking, when $c[i] = 1$, $x'[i]$ remains as it is. If $c[i] = 0$, $x'[i]$ is replaced by random noise. In this case, mask c can be optimized continuously by gradient descent, which makes the prediction confidence of a neighbour sample approaching to its corresponding original sample. We can express this procedure as:

$$c' = arg\min_{c} \left\| g(P(x', c))[t] - g(x')[t] \right\|_1 + \lambda \cdot \|c\|_1 \tag{3}$$

where t denotes the ground-truth label of x' and λ is a regularization factor. Finally, we generate corresponding neighbour samples for each original sample, and obtain an augment dataset after mixing, which greatly enriches the samples diversity.

Algorithm 1 Generate Augment Dataset

Input: Validation dataset D_{val}, mask c, perturbation n, regularization factor λ

Output: Augment dataset $D_{extended}$

$g(\cdot) = l_0 \circ f(\cdot)$, $D_{extended} \leftarrow \{\}$
for each $x' \in D_{val}$ **do**
  $t \leftarrow x'_{label}$
  $P(x', c) \leftarrow x' \cdot c + n \cdot (1 - c)$
  $L \leftarrow \left\| g(P(x', c))[t] - g(x')[t] \right\|_1$
  for $k = 1$ to maxIters **do**
    $\overline{c_k} \leftarrow c_{k-1} - \alpha \cdot \nabla L(c_{k-1})$
    $c_k \leftarrow \overline{c_k}/(1 + \lambda \cdot \alpha)$
  $D_{extended} \leftarrow D_{extended} + \{x', P(x', c)\}$
return $D_{extended}$

Algorithm 2 Search for Salient Neuron Set

Input: Layer l, extended dataset $D_{extended}$, number of class N

Output: Salient Neuron Sets $S(X_1), S(X_2), \ldots, S(X_N)$

for each $X_k \subseteq D_{extended}$ **do**
  Compute feature vector $l(X_k)$
  // i represents i-th neuron in layer l
  for each $l(X_k)_i \in l(\cdot)$ **do**
    $I(l(X_k)_i) \leftarrow \mu_i/\sigma_i$
  $S(X_k) \leftarrow$ top m neuron sets with reliable prediction
return $S(X_1), S(X_2), \ldots, S(X_N)$

C. Search for Salient Neuron Set

This process is to search for salient neuron set that dominate classification, which is described in **Algorithm 2**. Specifically, we want to obtain salient neuron set for each class given selected layers. Taking a class k as an example, we collect all neuron activation values X_k for analysis. Then, we utilize the activation values of the neuron set to estimate the importance of each neuron. We can use the fewest neurons with highest important scores to complete reliable prediction. Thus salient neuron set of the k-th class are captured, which is defined as $S(X_k)$. To check neurons' consistency thoroughly while reducing the searching space, we select two layers to search for salient neuron set, consisting of an input-close layer and an output-close layer. Each class has a feature extractor $f_k(\cdot)$ and a classifier L_k represented by its own corresponding salient neuron set from these two layers, which can capture and retain a given input's characteristics in both input and latent spaces.

For illustrative purposes, we focus on a layer and define it as l. So $l(x)$ represents the activation values of the neuron set with sample x in this layer. We use $l(X_k)_i$ to represent the i-th neuron's values with samples of the k-th class. Correspondingly, the i-th neuron's importance score can be expressed as:

$$I(l(X_k)_i) = \frac{\mu_i}{\sigma_i} \tag{4}$$

where μ_i and σ_i are the mean and deviation of the i-th neuron activation values driven by X_k, respectively. Obviously, the larger the values' mean, the more significant the neuron is, and the smaller the values' deviation, the more stable the neuron is. We collect the salient neuron set according to the graded scores until $L_k \circ f_k(\cdot)$ can use only this neuron set to correctly perform prediction on the ground-truth label.

D. Partial Neuron Set Prediction

After capturing salient neuron set for each class in two layers, we use them to serve corresponding feature extractors and classifiers. For instance, we will have 10 neuron sets for

a task with 10 classes: each neuron set corresponds to one class. Taking $S(X_k)$ as an example, we strengthen or increase this neuron set' weights and weaken that of other neuron sets in the same layer, then obtain the k-th class's feature extractor $f_k(\cdot)$ and classifier L_k. Obviously, $L_k \circ f_k(\cdot)$ can only be used for prediction of the k-th category, rather than other categories. *Thus, f_k and L_k (a feature extractor plus a classifier) can judge whether an input sample x belongs to class k or not.* Therefore, we only remain the k-th prediction *confidence* for $L_k \circ f_k(x)$, denoted as $T(x)_k$. Then we use feature extractor and classifier of each class to predict an input sample x concurrently, and record all prediction confidences $\{T(x)_k\}_{k \in classes}$. Finally, we determine if each confidence meets certain criteria. The detailed process is described by **Algorithm 3**.

Intuitively, for a benign sample, there is only the ground-truth category that exhibiting high confidence, thus meeting the criteria. So we can obtain its prediction result, denoted as $T(x)_{highest}$. However, some special situations need to be excluded. For example, multiple classes' confidences are simultaneously high on the same level, which the fed input will be regarded as a trigger input. This can be easily understood as a fact that the trigger input has both benign features of its original ground-truth class and adversarial features of its targeted class, In this context, the LinkBreaker can throw an alert and refuse to provide prediction services for it. Obviously, the LinkBreaker can complete correct prediction and anomaly detection simultaneously, and consequentially fundamentally suppress backdoor attacks.

It is not always reliable to reckon on the raw confidence alone to complete the classification, so we introduce the measure of credible confidence, then collect the confidence range of clean samples in advance in the offline as a reference. And then, an median absolute deviation is acted as the credible criteria, determining the lower bound of the credible confidences. Specifically, the k-th class's criteria is $T(X_k)_{MAD}$. In addition, to address over-fitting of each feature extractor and classifier, we inject the appropriate noise into non-salient neuron set' activation values in selected layers before prediction. Through above-mentioned two mechanisms, we can complete correct prediction and abnormal detection for arbitrary input samples without modifying the model parameters, which greatly improves the robustness of our classification system.

Algorithm 3 Partial Neuron Set Prediction

Input: Feature extractor f, input sample x_*, salient neuron sets $S : \{S(X_1), S(X_2), \ldots, S(X_N)\}$, uniform distribution U
Output: Right prediction result or Trigger signal

```
Sample x_r from U, labels ← {}
Compute f(x_r)
for each k ∈ classes do
    L_k, f_k ← S(X_k)
    T(X_k) ← L_k ∘ (f_k(X_k), f(x_r))
for each k ∈ classes do
    T(x_*)_k ← L_k ∘ (f_k(x_*), f(x_r))
    if T(x_*)_k ∈ T(X_k)_MAD then
        labels ← labels + {k}
if number of labels is 1 then
    return labels
else
    return Trigger signal
```

V. Experimental Evaluation

We evaluate the LinkBreaker performance against all two backdoor attack strategies: *data poisoning* and *model poisoning*. The performance is measured by two key metrics: classification accuracy rate of clean data and attack success rate of trigger inputs with three distinct classification tasks. We compare LinkBreaker with three state-of-the-arts: Neural Cleanse [31], STRIP [11] and ABS [27]. The Neural Cleanse iteratively reverse-engineers a "minimal" trigger/perturbation that converts all samples from all classes to each target label, and uses outlier detection to find the smallest trigger to resemble the true backdoor trigger. More generally, it reverse-engineers k trigger candidates, each corresponding to one of the target labels from all k classes, and takes the smallest trigger as the real one if it is lower than a given threshold. The STRIP takes advantage of the inconsistent sensitivity of benign and trigger samples by intentionally injecting perturbations to detect trigger inputs. The ABS leverages simulation analysis on individual neurons activations to scan neural network models for the purpose of identifying compromised neurons and then catching potential backdoors. Significantly, we also evaluate the effectiveness of LinkBreaker against source-target backdoor attacks, adversarial examples and out-of-distribution samples.

A. Experiment Setup

1) Datasets and Model Architectures: The tasks of each dataset is briefly described below. The Table I summarizes model architectures that train the dataset in addition to other details of these datasets.

- **MNIST**. The goal of this task is to recognize the numbers that appear in the hand-written digit images [36]. MNIST has 10 different classes gray-scale images, namely 10 hand-written digits (0-9).
- **CIFAR-10**. The task is to classify 10 classes colorful physical pictures [37], such as air planes, horses, cats and etc.
- **GTSRB**. This task simulates an intelligent recognition of autonomous driving, where it is used to recognize 43 classes of traffic signs [38].

2) Trigger Patterns: According to the two attack strategies, we use three triggers to add triggers to the samples in above tasks:

- **Random square** is a random chosen trigger that is with small size. It is placed in the lower right corner of images, about 1% of the original image.
- **Trojan square** is a small trigger that is generated by reverse engineering, occupying 3% of image approximately.

TABLE I
DATASET AND MODEL ARCHITECTURE SUMMARY

Dataset	Labels	Image Size	Images		Model Architecture
			Training	Testing	
MNIST	10	$28 \times 28 \times 1$	50,000	10,000	2 Conv + 2 Dense
CIFAR-10	10	$32 \times 32 \times 3$	50,000	10,000	8 Conv + 3 Pool+ 3 Dropout +1 Flatten + 2 Dense
GTSRB	43	$32 \times 32 \times 3$	39,209	12,630	ResNet20

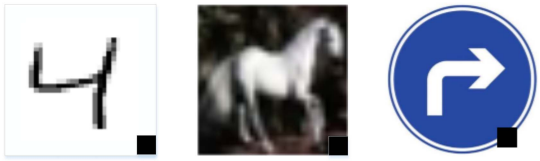


Fig. 3. Random squares used in data poisoning.

- **Trojan watermark** is also generated by reverse engineering while embedded in the samples, occupying 7% of image approximately.

3) Evaluation Metrics: To measure the performance of LinkBreaker, we validate the defense effectiveness by the prediction confidence of clean and trigger samples under each feature extractor in each classification task, and introduce the following metrics to evaluate the defense capability.

- **Clean Data Accuracy (CDA)** is the classification accuracy of deployed model for clean samples that are with no triggers.
- **Attack Success Rate (ASR)** is the probability that trigger inputs are misclassified into the attacker targeted labels.

As for the former CDA, a defense should retain it similar to the clean model counterpart. As for the later ASR, the defense should be maximally suppressing it.

B. Defense Against Data Poisoning

We first investigate the effectiveness of our LinkBreaker in defending against data poisoning attacks in which the trigger shares a salient neuron set—this is also the compromised neuron set from the backdoor perspective—of a targeted class given benign samples. We follow the methodology proposed by [10] and then compare our defense capability with other defense schemes. Below we first elaborate on how to distinguish trigger inputs.

1) Distinguish Trigger Inputs: We conduct data poisoning attacks on three tasks, where the trigger a random square, as shown in Fig. 3. For MNIST and CIFAR-10, we randomly select 2,000 clean samples and 2,000 trigger samples for evaluation, respectively, and select 4,000 clean samples and 4,000 trigger samples given GTSRB. For each input sample, we record the corresponding prediction confidence under each feature extractor.

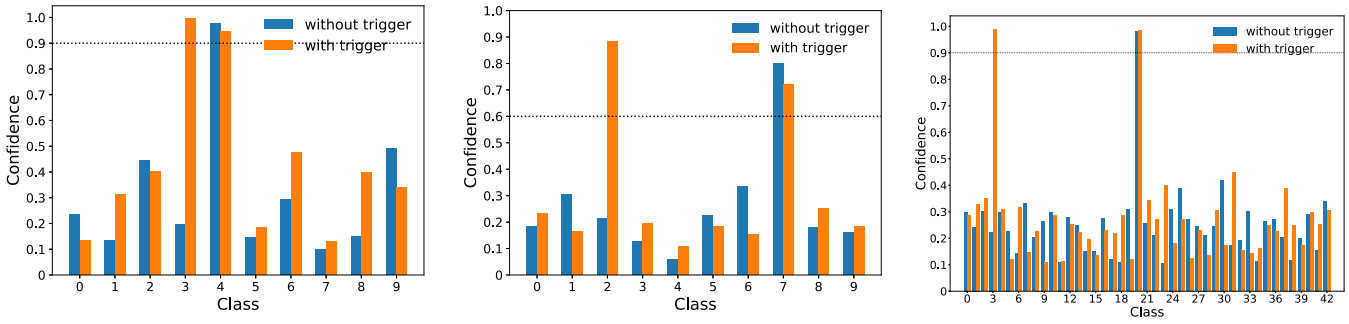
We show the confidence distributions for benign and trigger samples in defense against data poisoning in Fig. 4, where the ground-truth classes in each task are class 4, class 7, and class 20, respectively. Besides, we collect the confidence range of clean samples in the offline and compute $\{T(x)_k\}_{MAD}$, the lower confidence boundary as the threshold, as mentioned in **Algorithm 3**, where the minimum values of MNIST, CIFAR-10, and GTSRB are 0.9, 0.6 and 0.9, respectively. We can observe that in all three tasks, there is only one category with high confidence in the clean sample, while the trigger sample has two categories with extremely high confidence. In particular, it is clear from the comparison in MNIST and GTSRB that trigger samples achieve a high confidence of over 90% on both the source-class label and the targeted label. *It indicates that there are two salient features in the image at the same time.* Generally, for the benign sample, only one feature vector corresponding to the ground-truth label exceeds the threshold, while the rest do not. However, for the trigger sample, confidence from two feature vectors exceed the threshold. In other words, backdoored DNNs directly maps trigger samples to the benign feature space of targeted class. Therefore, the trigger can be distinguished by LinkBreaker due to the abnormal two high confidence values.

2) Defense Capability and CDA Comparison: Table II summarizes the ASR and CDA before and after apply LinkBreaker, where the CDA of the clean model is given as a benchmark. It can be seen that the CDA of backdoored model is similar to its clean model counterpart, while achieving extremely high attack success rate. Meanwhile, we can observe backdoored model have no notable CDA degradation, but the ASR has been suppressed to 1% to 3%. Most importantly, when the LinkBreaker is applied on a clean model, the CDA is also retained, this is important in practice as one yet knows whether the deployed is a backdoored model or clean model.

In overall, the LinkBreaker maintains the same level of defensive effect, keeping the ASR low to 2% in MNIST while less than 1% in CIFAR-10 and GTSRB. More specifically, LinkBreaker has outperforms the Neural Cleanse as it suppresses ASR lower in all three tasks. Because in Neural Cleanse, the trigger constructed by reverse engineering is not exactly the same as the original trigger, which could result in a variation in abnormal activation detection. Furthermore, the presented LinkBreaker maximizes the benign features of images to during inference, which can improve the robustness of the model, achieving slightly higher classification accuracy than others.

C. Defense Against Model Poisoning

Similar to defense against data poisoning, we also investigate the effectiveness of our LinkBreaker in defending against poisoning attacks in which the trigger links a compromised neuron set that is unrelated to begin neuron set of any class. Recall that the compromised neuron set of the data poisoning enabled backdoor attack shares the same begin neuron set of the targeted class. We use the attacking methods



(a) MNIST (target class is 3) (b) CIFAR-10 (target class is 2) (c) GTSRB (target class is 3)

Fig. 4. Confidence distributions for benign and trigger samples in defense against data poisoning.

TABLE II
THE ATTACK SUCCESS RATE AND CLASSIFICATION ACCURACY RATE IN DATA POISONING ATTACK BEFORE AND AFTER DEFENSE

	Schemes	Clean Model	Before Defense		After Defense	
		CDA	CDA	ASR	CDA	ASR
MNIST	Neural Cleanse [32]	99.03%	98.91%	99.76%	97.12%	2.7%
	STRIP [12]				98.91%	1.85%
	ABS [28]				97.04%	4.8%
	Our				98.06%	2%
CIFAR-10	Neural Cleanse [32]	88.27%	87.13%	100%	87.17%	1.6%
	STRIP [12]				87.25%	0%
	ABS [28]				85.72%	6.3%
	Our LinkBreaker				89.36%	0.4%
GTSRB	Neural Cleanse [32]	96.33%	95.74%	94.86%	96.02%	0.7%
	STRIP [12]				95.11%	0%
	ABS [28]				95.21%	4.7%
	Our LinkBreaker				98.28%	0.3%

Fig. 5. Trojan square and Trojan watermark are used in model poisoning.

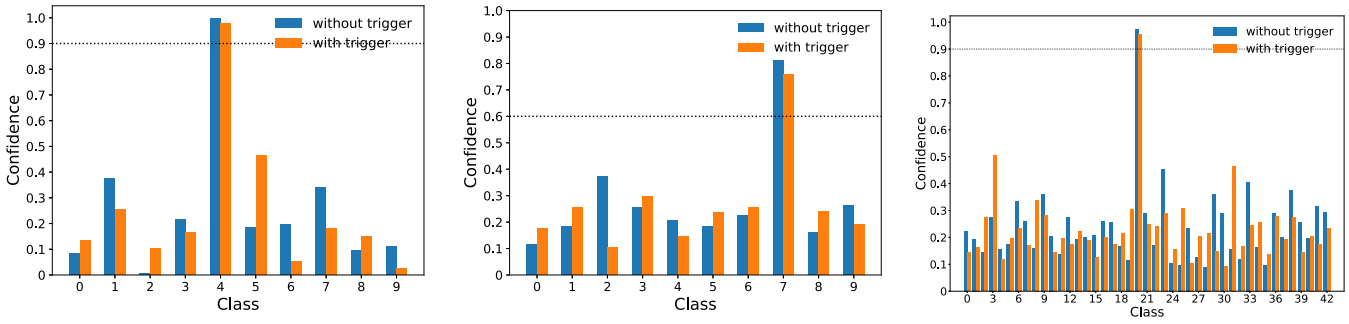
following [21], and then compare our defense capability with other defense schemes.

1) Distinguish Trigger Inputs: We also conduct Trojan attacks on three tasks, where the trigger are Trojan square and Trojan watermark, as shown in Fig. 5. Specifically, in MNIST and CIFAR-10, the Trojan square is the trigger, while the Trojan watermark is the trigger in GTSRB. To be clear, the trigger in the model poisoning enabled backdoor attack can not be arbitrarily crafted, they are rather reverse-engineered through the model poisoning attack process [21].

We compare the prediction results of clean samples and trigger samples, where the ground-truth classes in each task are class 4, class 7, and class 20, respectively. The corresponding prediction confidences are illustrated in Fig. 6. Apparently, only the prediction probability of ground-truth class is far beyond others in the clean sample. As for trigger inputs, it exhibits one much higher prediction probability in one specific class. This is different from the trigger input in the data poisoning enabled backdoored model, which has two high prediction confidences. This is because we only use the neuron set related to benign features for prediction, and the neuron set associated with Trojan triggers are obviously not among them. More specifically, the compromised neuron set related to the trigger has been excluded in the LinkBreaker salient neuron set step. Since this compromised neuron set cannot be activated by validation dataset containing no trigger samples. Therefore, after deploying LinkBreaker, the trigger sample is consistent with the clean sample as its trigger feature has been excluded. The results indicate that our LinkBreaker can suppress the trigger's effect when defending data poisoning enabled backdoor attacks, which helps classifier correctly predict trigger samples.

2) Defense Capability and CDA Comparison: Table III summarizes the comparison results of the ASR and CDA of model poisoning enabled backdoor attacks on all tested tasks. After applying LinkBreaker, the backdoored models have very low ASR in presence of trigger inputs, while also maintaining a high CDA in clean inputs. Although the ASR slightly varies in the three tasks, it is noticeable that we manage to reduce the ASR to be no more than 1.5%, while achieving comparable CDA as that of clean model. Through suppress the effect of trigger features, our scheme can remove the connection between Trojan triggers and neurons, which hence reduces ASR.

Compared to Neural Cleanse and ABS, our LinkBreaker has the distinct advantage of significantly reducing the ASR in all three tasks. Compared with STRIP, our LinkBreaker is



(a) MNIST (b) CIFAR-10 (c) GTSRB

Fig. 6. Confidence distributions for benign and trigger samples in defense against model poisoning.

TABLE III

THE ATTACK SUCCESS RATE AND CLASSIFICATION ACCURACY RATE IN MODEL POISONING ATTACK BEFORE AND AFTER DEFENSE

	Schemes	Clean Model	Before Defense		After Defense	
		CDA	CDA	ASR	CDA	ASR
MNIST	Neural Cleanse [32]	99.03%	98.06%	97.8%	97.43%	4.1%
	STRIP [12]				96.27%	1%
	ABS [28]				97.39%	3.1%
	Our LinkBreaker				99.1%	0.6%
CIFAR-10	Neural Cleanse [32]	88.27%	87.36%	100%	86.79%	5.6%
	STRIP [12]				87.4%	0.5%
	ABS [28]				85.52%	4.6%
	Our LinkBreaker				88.73%	1.3%
GTSRB	Neural Cleanse [32]	96.33%	96.25%	100%	96.16%	8.3%
	STRIP [12]				96.33%	0%
	ABS [28]				95.36%	3.5%
	Our LinkBreaker				98.2%	0.8%

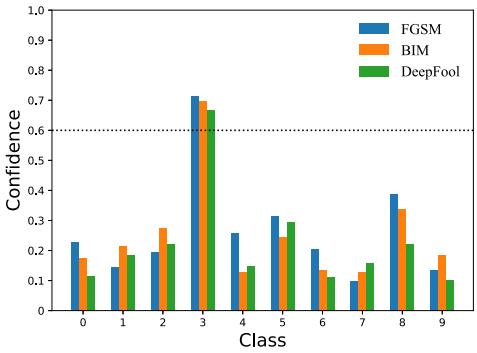


Fig. 7. Confidence distributions for different adversarial example attacks of the LinkBreaker.

neck-and-neck in terms of defensive effectiveness, slightly lower on CAFAR-10 and GTSRB but better on MNIST. Specifically,we decrease the ASR lower than 1% on MNIST. Note that so far, we have only considered source-agnostic backdoor attacks that all Neural Cleanse, STRIP, and ABS are designed to defend. Later we validate that the LinkBreaker can defend against the source-specific backdoor attack, where all other three fail to be effective.

The Fine-Pruning [16] also assumes that the backdoor is hidden in compromised neurons that are dormant when predicting benign samples, which to some extent is related to the LinkBreaker that only pre-selects neurons in charge of benign samples. Therefore, we have reproduced fine-tuning and compared its ASR suppression and CDA with the LinkBreaker. We conduct backdoor attacks with the trigger of Trojan Square on CIFAR-10. The results show that when 50 percent of the neurons are pruned as the first step of Fine-Pruning, the ASR is suppressed to 1.5%, but the CDA deteriorates to 57.3% at the same time. After implementing the fine-tuning strategy as the second step of the Fine-Pruning, the CDA is improved to 82.6%, but the ASR is also increasing to 13.1%. By contrast, the LinkBreaker achieves a CDA of 88.73% while maintaining an ASR as low as 1.3%. Overall, both the classification accuracy and defense effect of the Fine-Pruning is not as effective as the presented LinkBreaker. To be precise, LinkBreaker has about 6% CDA improvement and more than 10% ASR reduction in the same experimental setting.

D. Comprehensive Capability Comparison

We give the comparison of comprehensive capacities between our LinkBreaker and other defense schemes in terms of access privileges, robustness to variant backdoor attacks and adversarial samples, model accuracy enhancement and performance overhead. Table IV summarises the comparison results. Compared to other defense schemes, LinkBreaker demonstrates a better holistic defense capability, which is easy to deploy with reasonable user assumptions while keeping an acceptable overhead. Therefore, even model users with limited computational power can accomplish backdoor defense by deploying the LinkBreaker. Moreover, our scheme is also effective against adversarial example attacks and several types of backdoor variants, which significantly reduces the model

TABLE IV
COMPARISON OF DEFENSE CAPACITIES

Schemes	Access to Trojaned samples	Access to clean samples	Robustness to adversarial example attacks	Robustness to variant backdoor attacks	Enhance CDA	Total Cost
Neural Cleanse [32]	×	✓	×	×	×	High
STRIP [12]	×	✓	×	✓	×	Low
ABS [28]	×	✓	×	×	×	Medium
Our LinkBreaker	×	✓	✓	✓	✓	Medium

TABLE V
COMPARISON OF TIME COST

Schemes	Offline Phase (ms)	Online Phase(ms)
Neural Cleanse [32]	169190	0.4
STRIP [12]	1070	10.4
Our LinkBreaker	9190	2.78

TABLE VI
ROBUSTNESS COMPARISON AGAINST ADVERSARIAL EXAMPLE ATTACKS

Schemes	FGSM	BIM	DeepFool
STRIP [12]	0%	0%	9%
Our LinkBreaker	86%	77%	64%

security risks. Meanwhile, our scheme is based on image content where the prediction results is related to the image's semantic information, which enhances the classification accuracy of the model to a certain extent.

The comparison of computational overhead are presented in Table V. In terms of its offline overhead, it is much lower compared with Neural Cleanse, reduced by 18× in our tests. When compared with the STRIP during the online phase, it can be seen that LinkBreaker has a 3× latency reduction. Smaller the latency, better the online detection methods. Because it is important to reduce the latency for real-time applications. As for the LinkBreaker, the latency can be further optimized by using parallel computing. This is facilitated by the design of the LinkBreaker, as now we can concurrently run each benign neuron set (equal to a feature extractor) corresponding to each class at the same time given one input, avoiding to sequentially execute them.

E. Holistic Robustness

1) Adversarial Examples: LinkBreaker demonstrates robustness against adversarial examples. We conduct experiments on CIFAR-10 against three adversarial example attacks (FGSM [39], BIM [40] and DeepFool [41]). The results are illustrated in Fig. 7, where the ground-truth class is class 3. Apparently, only the prediction confidence of the ground-truth class is far beyond others in adversarial examples, which demonstrates the robustness of the presented LinkBreaker against adversarial example attacks. As a comparison, we evaluate the robustness of STRIP against adversarial example attacks, results of which are summarized in Table VI. It can be seen that these adversarial example attacks can trivially bypass the STRIP as the STRIP fails to produce ground-truth prediction, but are invalid to our scheme. This is because LinkBreaker suppresses the role of the particular perturbations in adversarial examples and uses only benign features for prediction. Therefore, the LinkBreaker can still make correct predictions in most cases given adversarial examples.

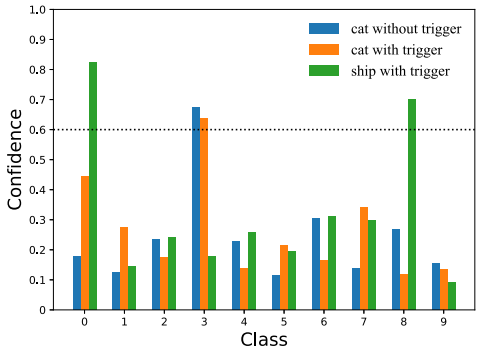


Fig. 8. Confidence distributions against source-specific backdoor attack. "Airplane" is the target class, "ship" is the source class and "cat" is the non-source class.

2) Source-Specific Backdoor: Majority of backdoor defenses are devised against the common input-agnostic backdoor, where *any* input with the trigger regardless its source class will activate the backdoor. However, there are various backdoor attacks, one is the source-specific backdoor attack, which is challenging to be detected by most of current state-of-the-arts [20]. As for the source-specific backdoor attack, the backdoor activation depends not only on the trigger but also the input source. This means that only samples from *certain attacker chosen classes* stamping with the trigger, the backdoor will be effect. In principle, the LinkBreaker is applicable against source-specific backdoor attack as we do not have the assumption requiring the backdoor to be input-agnostic that are commonly used [12], [32].

We implement the source-specific backdoor following the attack strategy in [12]. We select the Trojan square as the trigger pattern with CIFAR-10 classification task, where the source classes are set to "horse" (class 7) and "ship" (class 8), and the attack target class is set to "airplane" (class 0). The backdoor model has a classification accuracy of 85.86% for clean samples and a high success rate of 91.06% for source class samples with embedded triggers, while the attack success rate decreases to 7.16% for non-source class samples, which implies that the source-specific backdoor has been implanted. After applying LinkBreaker, as shown in Fig. 8, two classes of distinctive features exist for the source class samples with embedded triggers, while only one class of distinctive feature exists for the clean samples (including the non-source class samples with embedded triggers). This demonstrates the LinkBreaker is also effective against source-specific backdoor attacks.

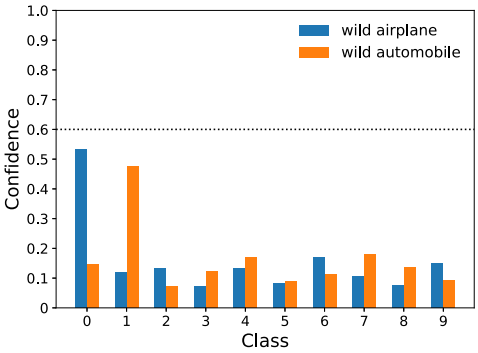


Fig. 9. Confidence distributions for OOD samples of the LinkBreaker.

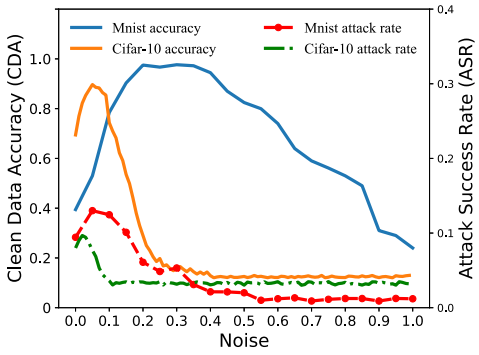


Fig. 10. The impact of noise amplitude on the defense.

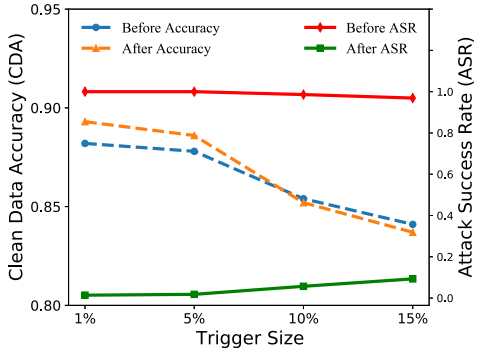


Fig. 11. The impact of trigger size on the defense.

Overall, the LinkBreaker suppresses ASR of source-specific backdoor attack to be 13.2% from up to 91.06%, and retains the CDA of the clean samples to be 82.62%. The slightly notable CDA drop is because after embedding the trigger, some non-source class samples are identified as trigger samples, which reduces the accuracy. If we exclude this case, the CDA is comparable to the original CDA of 85.86%. In fact, the attacker should make sure non-source trigger images are correctly classified into their ground-truth labels to maintain stealthiness. Therefore, given a strictly implemented source-specific backdoor attack, the LinkBreaker is with high efficacy. Note both Neural Cleanse and STRIP are ineffective to deal with such backdoor attack.

3) Out-of-Distribution (OOD) Detection: OOD detection refers that the model performs prediction on OOD sample, where the OOD samples are not in identical distribution with the samples of the original training dataset. We apply LinkBreaker on OOD samples for evaluating its feasibility in this extreme case. Specifically, we use some images of "airplane" and "automobile" that are out-of-distribution with CIFAR-10 and reduce the size to 32×32. As shown in Fig. 9, the prediction confidences of these out-of-distribution benign samples in each class are lower than the threshold, which is inconsistent with the expected prediction of benign samples. However, observe that the confidence in class 0 ("airplane") and class 1 ("automobile") still remains highest over other classes, implying that LinkBreaker can make correct predictions for benign samples even outside the distribution. From another point of view, samples with all confidences below the threshold can be considered as OOD samples and thus be detected by LinkBreaker.

VI. Discussion

This part discusses other factors that may influence the LinkBreaker performance and the defense performance of LinkBreaker on several additional advanced backdoor variants.

A. Noise Amplitude

To mitigate the overfitting problem of the feature extractors, we have introduced random noise features in Algorithm 3 to offset the impact of classification, so the noise amplitude may affect the defense effect. We explore the impact of noise amplitude on the LinkBreaker performance, based on MNIST and CIFAR-10 tasks, separately. Figure 10 shows the CDA and ASR when noise amplitude varies. Clearly, the classification accuracies in both tasks are greatly affected by the amplitude: either too high or too low amplitude renders difficulty to achieve the optimal accuracy. A potential reason is that all feature extractors will be over-fitting when the amplitude is low, so all classes' confidence is too high to be recognized normally, while the useless features will randomize prediction results when the amplitude is high. As for the ASR, it initially increases slightly as the noise amplitude increases, and then decreases sharply and maintains a low level. Therefore, considering the impact of noise amplitude on the classification accuracy and attack success rate, we can choose the appropriate noise amplitude to help achieve the optimal performance.

B. Trigger Size

The size of trigger patterns may also affect defense effectiveness. We explore this influence by experiments on CIFAR-10, in which the compromised DNNs are trained using trigger samples with Trojan squares accounting for 1%, 5%, 10% and 15% of the image, respectively. Fig. 11 depicts the results for the CDA of clean samples and the ASR of trigger samples. One hand, the CDA of the compromised DNN decrease as the trigger size increases, indicating that the increase in trigger size has an adverse effect on the model accuracy. On the other hand, trigger samples can easily activate the backdoor in the compromised DNN, and the

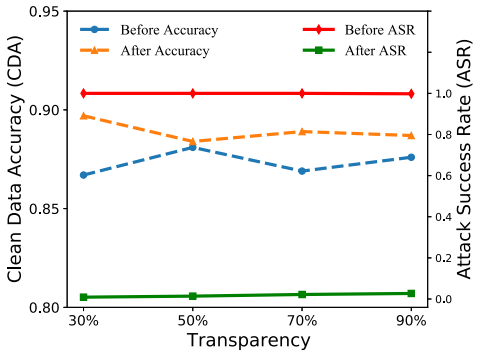


Fig. 12. The impact of trigger transparency on the defense.

ASR approaches nearly 100% before applying LinkBreaker. Although LinkBreaker's defensive effect on compromised DNNs decreases a little as the trigger size increases, it can significantly reduce the ASR to below 9%. It suggests that our scheme is affected by the size of triggers, but still has a strong robustness to a large-scale trigger.

C. Trigger Transparency

The visibility of a trigger often affects backdoor attacks, so an attacker will increase the concealment by changing its transparency. Therefore, the transparency of the trigger may affect the defensive performance. We conduct controlled experiments on CIFAR-10 with transparency settings of 30%, 50%, 70%, and 90%, where the trigger is Trojan square. As shown in Fig. 12, the ASR stays below 2.7% under the four transparency settings. Therefore, the results indicate that the LinkBreaker is insensitive to trigger transparency. Because the more transparent the trigger is, it is more difficult to map into the corresponding feature space, and finally be detected or correctly classified, which is slightly better than STRIP.

D. Model Depth

Model depth may be another contributing factor affecting defensive performance. We evaluate this impact by performing data poisoning attacks on three models (ResNet18, ResNet50, and ResNet101) with random squares as triggers on GTSRB task. We select uniformly distributed 4 hidden layers in ResNet18, ResNet50, and ResNet101 models to search for some representative neurons, respectively, and an extended dataset of the same size for the defense process. The results are presented in Table. VII. Compared to ResNet18, the CDA of ResNet50 and ResNet101 is slightly lower, and the ASR is slightly high. It indicates that the increase in model complexity has a slight negative impact on the performance of our LinkBreaker. However, it is worth noting that although the ASRs of ResNet50 and ResNet101 are higher compared to that of ResNet18, they are still well below 5%. In other words, our LinkBreaker works efficiently to complex models, despite a negligible impact on the CDA.

E. Data Augmentation Method

Different data augmentation methods can also affect defense effectiveness. We perform a comparative experiment with

TABLE VII

THE ASR AND CDA IN ABLATION EXPERIMENTS OF MODEL COMPLEXITY, IN PARTICULAR, DEPTH

	ResNet18	ResNet50	ResNet101
CDA	94.86%	92.47%	93.52%
ASR	0.3%	3.9%	4.1%

TABLE VIII

THE ASR AND CDA IN MODEL POISONING ATTACK USING DIFFERENT DATA AUGMENTATION METHODS

Data Augmentation Methods	CDA	ASR
Neighbor Samples	88.73%	1.3%
Perturbation Samples	86.18%	7.1%

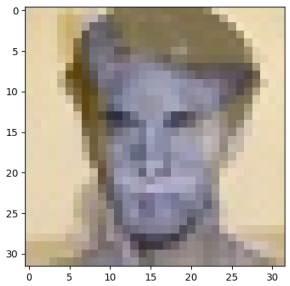


Fig. 13. A reflection image ('cat + trigger') bypassing the LinkBreaker.

perturbation samples (adding random noise) on CIFAR-10, where the Trojan square is as a trigger. We generate five perturbation samples for each sample by directly injecting random noise, and then obtain an augment dataset. The results are summarized in Table VIII. As for the method of perturbation samples, the CDA of the compromised model for benign samples is measured to be 86.18%, while the ASR is 7.1%. The neighbor samples suppress the ASR to be as low as 1.3% while achieving the CDA to be 88.73%. Therefore, the results indicate that the scheme based on perturbation samples can work in defending, but its effect is inferior to that of neighbor samples.

F. Against More Backdoor Variants

We also investigate the LinkBreaker's performance on a slightly deeper network against additional state-of-the-art backdoor variants, hidden trigger backdoor [42], reflection backdoor [26], composite backdoor [27] and input-aware dynamic backdoor [43]. Hidden trigger backdoor attacks exploit the so-called "feature collision attack" to make poisoned image content consistent with its target label, bypass human inspection by eyes. Reflection backdoor attacks use semantic reflections as triggers. Composite backdoor attacks use triggers that are composed of benign features from multiple classes for data poisoning, to evade backdoor scanning. Input-aware dynamic backdoor attacks [43] design unique triggers for each input sample by generative adversarial neural networks.

All experiments are conducted on ResNet20 with the CIFAR-10 dataset. We record the CDA drop of each backdoor variant compared to the clean model, and the ASR before and

TABLE IX
THE ASR AND CDA IN STATE-OF-THE-ART BACKDOOR VARIANTS

Backdoor Variants	Clean Model CDA	CDA Drop	ASR Before	ASR After
Hidden Trigger Backdoor [42]	90.3%	2%	78%	4.7%
Reflection Backdoor [26]		2%	83.92%	25.6%
Composite Backdoor [27]		-0.3%	70%	9.3%
Input-aware Dynamic Backdoor [43]		7%	63.52%	7.9%

after implementing LinkBreaker in Table. IX. The results show that some backdoor variants cause CDA drop of the model, influencing the performance of the model. Especially, input-aware dynamic backdoor attacks lead to performance degradation of the model, with CDA reduced by almost 7%. Although not as superior as against common backdoors inserted through data poisoning attacks (0.4%) and model poisoning attacks (1.7%), LinkBreaker still demonstrate effectiveness against these advanced variants, ASR of most attacks below 10%. As for reflection backdoor attacks, we check those trigger samples (Fig. 13) bypassing our LinkBreaker, and find that these samples often visually abnormal, easy identified by manual inspection—note the reflection attack crafted samples should be inconspicuous by its design.

G. Against Adaptive Attacks

We investigate the LinkBreaker's performance against a stronger attacker who can perform the adaptive attack. In this context, the attacker has full knowledge of LinkBreaker including the defense strategy and *our picked hiding layers*. Notably, both are non-trivial to gain by the attacker in practice. For example, the defender may use different backdoor defenses, which one chosen is unknown to the attacker. The attacker in the adaptive attack is allowed to evade the defense by optimizing her/his attack strategy. We conduct experiments with a 4-layer convolutional neural network on MNIST dataset. Specifically, to implant a backdoor and bypass our defense, the attacker needs to use triggers that the trigger samples with added triggers are indistinguishable from those of the **target class** in the input and hidden layers. Therefore, the backdoor model training consists of two phases: 1) optimizing the objective function that measures the difference between the prediction of the backdoored model and the target class; 2) iteratively updating the trigger to make the trigger sample close to the benign sample of the target class. We conduct experiments with a 4-layer convolutional neural network on MNIST dataset. The methodology of [44] is leveraged to optimize the attack objective function.

The results show that the attack success rate of the adaptive attack is dependent on the size of the trigger. As shown in Fig.14, the larger the size of the trigger, the higher the ASR and the easier it is to bypass the LinkBreaker. The ASR can reach to 90% when the trigger size is 20×20. The larger trigger size is a trade-off for the attacker to bypass the LinkBreaker as it stands for the stealthiness to a large extent. Fig.15 shows trigger samples with different size triggers. The trigger takes up 50% of the sample when the ASR is up to 90%, severely affecting the stealthiness of the trigger inputs. Note in previous works, it normally assumes the trigger size should be small to be stealth (i.e. 7% of the input image size [32]).

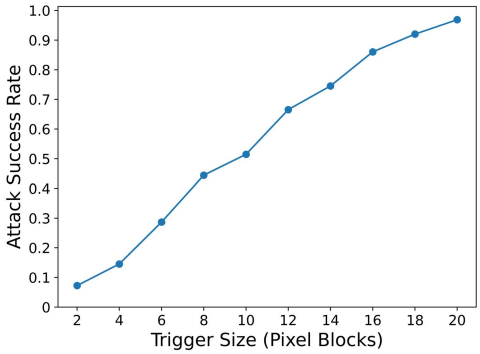
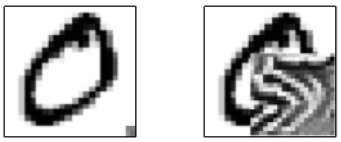


Fig. 14. The impact of trigger size on adaptive attacks defense.



(a) 2×2 pixels (b) 20×20 pixels

Fig. 15. Samples with different size triggers.

Though with the decreased stealthiness as a trade-off, the adaptive attack can bypass the LinkBreaker to a large extent. However, this is not surprising. It has been recognized that dealing with adaptive attack is very challenging, if not impossible, in the security-race of adversarial attacks on deep learning. This includes the adversarial example attacks [45] and backdoor attacks on deep neural network [46]. Because the attacker has full knowledge of defense strategy and can evade the defense by delicately optimizing the attack strategy, which greatly increases the difficulty of defense. As for backdoor defenses, it usually poorly performed against strong adaptive attacks. Tan *et al.* [46] implemented adaptive attacks by an adversarial backdoor embedding algorithm to optimize the feature representation of triggers, which successfully bypasses several advanced backdoor defense schemes [15], [32]. The NIC [30] acknowledges ineffectiveness of defeating against adaptive attacks, though it points out that it takes 420 seconds for adaptive attacks to generate an attack image to bypass the NIC detector, compared to only one second to generate a normal adversarial sample. Therefore, devising efficient backdoor countermeasures that can even greatly resist to (strong) adaptive attacks is still challenging future work.

VII. Conclusion

In this paper, we investigated the natural link between the trigger and the internal backdoor represented by a compromised neuron set and identified feature differences between benign and attack samples. Accordingly, we devised a new defense scheme called LinkBreaker that effectively defends against backdoor attacks by suppressing the role of the compromised neuron set associated to trigger features, severing the link between triggers and backdoors. LinkBreaker exploits features consistency check to identify trigger features so that the model output is only determined with benign features. Extensive experiments on three popular datasets validate the effective of the LinkBreaker when defeating the common input-agnostic backdoor attacks. Significantly, the LinkBreaker is with a holistic robustness against the source-specific backdoor attacks as well as the adversarial example attacks, which are missed by current state-of-the-arts.

References

[1] X. Guo *et al.*, "VeriFL: Communication-efficient and fast verifiable aggregation for federated learning," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 1736–1751, 2021.

[2] L. Zhou, A. Fu, G. Yang, H. Wang, and Y. Zhang, "Efficient certificateless multi-copy integrity auditing scheme supporting data dynamics," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 2, pp. 1118–1132, Apr. 2021.

[3] Z. Chen, A. Fu, R. H. Deng, X. Liu, Y. Yang, and Y. Zhang, "Secure and verifiable outsourced data dimension reduction on dynamic data," *Inf. Sci.*, vol. 573, pp. 182–193, Sep. 2021.

[4] C. Zhou, A. Fu, S. Yu, W. Yang, H. Wang, and Y. Zhang, "Privacy-preserving federated learning in fog computing," *IEEE Internet Things J.*, vol. 7, no. 11, pp. 10782–10793, Nov. 2020.

[5] Q. Wang, B. Zheng, Q. Li, C. Shen, and Z. Ba, "Towards query-efficient adversarial attacks against automatic speech recognition systems," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 896–908, 2021.

[6] Z. Chen, A. Fu, Y. Zhang, Z. Liu, F. Zeng, and R. H. Deng, "Secure collaborative deep learning against GAN attacks in the Internet of Things," *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5839–5849, Apr. 2021.

[7] A. Fu, X. Zhang, N. Xiong, Y. Gao, H. Wang, and J. Zhang, "VFL: A verifiable federated learning with privacy-preserving for big data in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 18, no. 5, pp. 3316–3326, May 2022.

[8] S. G. Finlayson, J. D. Bowers, J. Ito, J. L. Zittrain, A. L. Beam, and I. S. Kohane, "Adversarial attacks on medical machine learning," *Science*, vol. 363, no. 6433, pp. 1287–1289, 2019.

[9] G. Severi, J. Meyer, S. Coull, and A. Oprea, "Explanation-guided backdoor poisoning attacks against malware classifiers," in *Proc. 30th USENIX Secur. Symp. (USENIX)*, 2021, pp. 1487–1504.

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

[11] N. Akhtar, J. Liu, and A. Mian, "Defense against universal adversarial perturbations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3389–3398.

[12] 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.*, Dec. 2019, pp. 113–125.

[13] M. Javaheripi, M. Samragh, G. Fields, T. Javidi, and F. Koushanfar, "CleaNN: Accelerated trojan shield for embedded neural networks," in *Proc. 39th Int. Conf. Computer-Aided Design*, Nov. 2020, pp. 1–9.

[14] M. Du, R. Jia, and D. Song, "Robust anomaly detection and backdoor attack detection via differential privacy," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2020, pp. 1–19.

[15] B. Tran, J. Li, and A. Madry, "Spectral signatures in backdoor attacks," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2018, pp. 1–11.

[16] 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 (RAID)*, 2018, pp. 273–294.

[17] S. Kolouri, A. Saha, H. Pirsiavash, and H. Hoffmann, "Universal litmus patterns: Revealing backdoor attacks in CNNs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 301–310.

[18] H. Qiu, Y. Zeng, S. Guo, T. Zhang, M. Qiu, and B. Thuraisingham, "DeepSweep: An evaluation framework for mitigating DNN backdoor attacks using data augmentation," in *Proc. ACM Asia Conf. Comput. Commun. Secur.*, May 2021, pp. 363–377.

[19] Y. Li, Y. Jiang, Z. Li, and S.-T. Xia, "Backdoor learning: A survey," 2020, *arXiv:2007.08745*.

[20] T. Di, W. XiaoFeng, T. Haixu, and Z. Kehuan, "Demon in the variant: Statistical analysis of DNNs for robust backdoor contamination detection," in *Proc. 30th Secur. Symp. (USENIX)*, 2021, pp. 1541–1558.

[21] Y. Liu *et al.*, "Trojaning attack on neural networks," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2018, pp. 1–15.

[22] Z. Xi, R. Pang, S. Ji, and T. Wang, "Graph backdoor," in *Proc. 30th Secur. Symp. (USENIX)*, 2021, pp. 1523–1540.

[23] Z. Zhang, J. Jia, B. Wang, and N. Z. Gong, "Backdoor attacks to graph neural networks," in *Proc. 26th ACM Symp. Access Control Models Technol.*, Jun. 2021, pp. 15–26.

[24] M. Barni, K. Kallas, and B. Tondi, "A new backdoor attack in CNNS by training set corruption without label poisoning," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 101–105.

[25] 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.*, Nov. 2019, pp. 2041–2055.

[26] Y. Liu, X. Ma, J. Bailey, and F. Lu, "Reflection backdoor: A natural backdoor attack on deep neural networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2020, pp. 182–199.

[27] J. Lin, L. Xu, Y. Liu, and X. Zhang, "Composite backdoor attack for deep neural network by mixing existing benign features," in *Proc. Conf. Comput. Commun. Secur.*, Oct. 2020, pp. 113–131.

[28] 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. Conf. Comput. Commun. Secur.*, Nov. 2019, pp. 1265–1282.

[29] W. Aiken, H. Kim, S. Woo, and J. Ryoo, "Neural network laundering: Removing black-box backdoor watermarks from deep neural networks," *Comput. Secur.*, vol. 106, Jul. 2021, Art. no. 102277.

[30] S. Ma, Y. Liu, G. Tao, W.-C. Lee, and X. Zhang, "NIC: Detecting adversarial samples with neural network invariant checking," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2019, pp. 1–15.

[31] B. Chen *et al.*, "Detecting backdoor attacks on deep neural networks by activation clustering," 2018, *arXiv:1811.03728*.

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

[33] Y. Ji, X. Zhang, S. Ji, X. Luo, and T. Wang, "Model-reuse attacks on deep learning systems," in *Proc. Conf. Comput. Commun. Secur.*, Oct. 2018, pp. 349–363.

[34] R. C. Fong and A. Vedaldi, "Interpretable explanations of black boxes by meaningful perturbation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3429–3437.

[35] A. Shafahi *et al.*, "Poison frogs! Targeted clean-label poisoning attacks on neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2018, pp. 6103–6113.

[36] 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.

[37] A. Krizhevsky *et al.*, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep. TR-2009, 2009.

[38] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: The German traffic sign detection benchmark," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Aug. 2013, pp. 1–8.

[39] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014, *arXiv:1412.6572*.

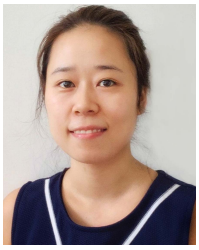
[40] A. Kurakin *et al.*, "Adversarial examples in the physical world," 2016, *arXiv:1607.02533*.

[41] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A simple and accurate method to fool deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2574–2582.

[42] A. Saha, A. Subramanya, and H. Pirsiavash, "Hidden trigger backdoor attacks," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 7, pp. 11957–11965.

[43] A. Nguyen and A. Tran, "Input-aware dynamic backdoor attack," 2020, *arXiv:2010.08138*.

[44] K. Doan, Y. Lao, and P. Li, "Backdoor attack with imperceptible input and latent modification," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2021, pp. 18944–18957.
[45] N. Carlini and D. Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," in *Proc. 10th ACM Workshop Artif. Intell. Secur.*, Nov. 2017, pp. 3–14.
[46] T. J. L. Tan and R. Shokri, "Bypassing backdoor detection algorithms in deep learning," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS P)*, Sep. 2020, pp. 175–183.



Zhenzhu Chen (Graduate Student Member, IEEE) received the B.S. degree in mathematics and applied mathematics from the Nanjing University of Science and Technology, China, in 2016, where she is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering. Her research interests include cloud computing security and outsourced computation.

Shang Wang received the B.S. degree in computer science and technology from Nanjing Normal University, China, in 2020. He is currently pursuing the M.S. degree with the School of Computer Science and Engineering, Nanjing University of Science and Technology, China. His current research interests include AI security, privacy preserving, and deep learning.

Anmin Fu (Member, IEEE) received the Ph.D. degree in information security from Xidian University in 2011. From 2017 to 2018, he was a Visiting Research Fellow with the University of Wollongong, Australia. He is currently a Professor with the Nanjing University of Science and Technology, China. He has published more than 80 technical papers, including international journals and conferences, such as IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE TRANSACTIONS ON SERVICES COMPUTING, IEEE INTERNET OF THINGS JOURNAL, *Computers & Security*, IEEE ICC, IEEE GLOBECOM, and ACISP. His research interests include the IoT security, cloud computing security, and privacy preserving.

Yansong Gao (Member, IEEE) received the M.S. degree from the University of Electronic Science and Technology of China in 2013 and the Ph.D. degree from the School of Electrical and Electronic Engineering, The University of Adelaide, Australia, in 2017. He is currently with the School of Computer Science and Engineering, Nanjing University of Science and Technology, China. His current research interests include AI security and privacy, hardware security, and system security.

Shui Yu (Senior Member, IEEE) is currently a Full Professor with the School of Software, University of Technology Sydney, Australia. His H-index is 32. He has published two monographs, edited two books, and more than 200 technical papers, including top journals and top conferences, such as IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS (TPDS), IEEE TRANSACTIONS ON COMPUTERS (TC), IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY (TIFS), IEEE TRANSACTIONS ON MOBILE COMPUTING (TMC), IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING (TKDE), IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING (TETC), IEEE/ACM TRANSACTIONS ON NETWORKING (ToN), and INFOCOM. He initiated the research field of networking for big data in 2013. He actively serves his research communities in various roles. His research interests include security and privacy, networking, big data, and mathematical modeling. He is currently serving on the Editorial Board of IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, *IEEE Communications Magazine*, IEEE INTERNET OF THINGS JOURNAL, IEEE COMMUNICATIONS LETTERS, IEEE ACCESS, and IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS. He is a member of AAAS and ACM, the Vice Chair of Technical Committee on Big Data of IEEE Communication Society, and a Distinguished Lecturer of the IEEE Communication Society. He has served more than 70 international conferences as a member of organizing committee, such as the Publication Chair for IEEE Globecom 2015 and IEEE INFOCOM 2016 and 2017, the TPC Chair for IEEE BigDataService 2015, and the General Chair for ACSW 2017.

Robert H. Deng (Fellow, IEEE) is currently the AXA Chair Professor of cybersecurity, the Director of the Secure Mobile Centre, and the Deputy Dean of the Faculty and Research, School of Computing and Information Systems, Singapore Management University. His research interests include data security and privacy, network security, and applied cryptography. He is a fellow of the Academy of Engineering Singapore. He received the Outstanding University Researcher Award from the National University of Singapore, the Lee Kuan Yew Fellowship for Research Excellence from SMU, and the Asia–Pacific Information Security Leadership Achievements Community Service Star from the International Information Systems Security Certification Consortium. He serves/served on the Editorial Board of the ACM TRANSACTIONS ON PRIVACY AND SECURITY, IEEE SECURITY AND PRIVACY, IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, and *Journal of Computer Science and Technology*, and the Steering Committee Chair of the ACM Asia Conference on Computer and Communications Security.