Inspecting Prediction Confidence for Detecting Black-Box Backdoor Attacks

Tong Wang[1], Yuan Yao[1], Feng Xu[1], Miao Xu[2], Shengwei An[3], Ting Wang[4]
[1]State Key Laboratory for Novel Software Technology, Nanjing University, China
[2]University of Queensland, Australia
[3]Purdue University, USA
[4]Stony Brook University, USA
mg20330065@smail.nju.edu.cn, y.yao@nju.edu.cn, xf@nju.edu.cn
miao.xu@uq.edu.au, an93@purdue.edu, twang@cs.stonybrook.edu

Abstract

Backdoor attacks have been shown to be a serious security threat against deep learning models, and various defenses have been proposed to detect whether a model is backdoored or not. However, as indicated by a recent black-box attack, existing defenses can be easily bypassed by implanting the backdoor in the frequency domain. To this end, we propose a new defense DTINSPECTOR against black-box backdoor attacks, based on a new observation related to the prediction confidence of learning models. That is, to achieve a high attack success rate with a small amount of poisoned data, backdoor attacks usually render a model exhibiting statistically higher prediction confidences on the poisoned samples. We provide both theoretical and empirical evidence for the generality of this observation. DTINSPECTOR then carefully examines the prediction confidences of data samples, and decides the existence of backdoor using the shortcut nature of backdoor triggers. Extensive evaluations on six backdoor attacks, four datasets, and three advanced attacking types demonstrate the effectiveness of the proposed defense.

1 Introduction

Deep learning models have been found to be vulnerable to backdoor attacks (a.k.a. trojan attacks) (Gu, Dolan-Gavitt, and Garg 2017). For example, by poisoning the training data, a face recognition model can be easily manipulated to predict any person as the target person. Typically, a successful backdoor attack poisons a small set of training data, and renders a trained model satisfying the following two requirements: 1) *stealthiness*, the trojaned model does not have significant accuracy degradation on benign inputs; 2) *effectiveness*, it classifies a poisoned input (e.g., input stamped with a trigger) to a target label with a high probability.

Up to date, several defenses have been proposed to inspect whether a given model has been trojaned or not. Among them, Neural Cleanse (NC) (Wang et al. 2019) leverages the intuition that a backdoor trigger is usually of small size, and learns to search for such small areas to detect triggers. However, NC is observed to be sensitive to the trigger size, performing poorly when the trigger size is relatively large (Guo et al. 2020). ABS (Liu et al. 2019) proposes to analyze the neuron activation, based on the assumption that stimulating one inner neuron of a trojaned model can easily lead to the target label. However, ABS falls short against advanced attacks involving multiple triggers or multiple target labels (Gao et al. 2020). ULP (Kolouri et al. 2020) and MNTD (Xu et al. 2021) propose to train a meta-classifier based on a number of clean and trojaned models. Although the trained meta-classifier can well recognize the triggers that share similar patterns to those within the training data, it may not generalize well to unseen attacks or triggers. More importantly, a recent backdoor attack FTROJAN (Wang et al. 2022) shows that the above defenses can be easily bypassed by breaking their underline assumptions and implanting the backdoor in the frequency domain.

In this work, we propose a new backdoor defense DTINSPECTOR against black-box backdoor attacks. Different from the existing defenses, our method is motivated by the observation that an effective black-box backdoor attack usually needs to achieve high prediction confidence on the small amount of poisoned data at the training stage, so as to ensure a high attack success rate (ASR) on the poisoned inputs at the inference stage. We show that this observation generally holds for black-box attacks with both theoretical and empirical evidence. Moreover, this observation does not rely on the trigger size, the neuron activation pattern, or the trigger pattern, and thus has the potential to address the above limitations. DTINSPECTOR then carefully examines the prediction confidences of data samples, and decides the existence of backdoor using the shortcut nature (Geirhos et al. 2020; Li et al. 2021c) of backdoor triggers.

We evaluate DTINSPECTOR against six black-box backdoor attacks (from BADNET (Gu, Dolan-Gavitt, and Garg 2017) to FTROJAN (Wang et al. 2022)) and three advanced attacking types on four datasets. The studied advanced attacks (Wang et al. 2019) include partial attack (poisoning data from a specific label), MTOT attack (multiple triggers for one target label), and MTMT attack (multiple triggers for multiple target labels). The results show that our defense can accurately detect the trojaned models as well as the infected labels. We also compare DTINSPECTOR with four existing defenses (i.e., NC (Wang et al. 2019), ABS (Liu et al. 2019), ULP (Kolouri et al. 2020), and MNTD (Xu et al. 2021)). The results show that DTINSPECTOR outperforms these competitors: 1) it is less sensitive to the trigger size,

2) it can successfully detect advanced attacks, 3) it can be naturally applied to unseen attacks/triggers.

The main contributions of this paper include:

- A new perspective related to the prediction confidence for detecting black-box backdoor attacks, which is both theoretically and empirically supported.
- A new backdoor detection method DTINSPECTOR that is more robust against trigger size, neuron activation pattern, trigger pattern, and advanced attacking types.
- Extensive experimental evaluations showing the effectiveness of DTINSPECTOR.

Threat Model. We consider the commonly-studied black-box backdoor attacks (Gu, Dolan-Gavitt, and Garg 2017; Liu et al. 2018; Turner, Tsipras, and Madry 2018; Wang et al. 2022). The adversary defines a trigger beforehand, injects it into data samples, and then disseminates the data. The adversary does not have access to the training process and the model. We also assume that the adversary only attacks a minority of labels. For the defender, we assume she can access both the training data and the model.

2 Key Observation

Our key observation is that an effective black-box backdoor attack usually needs to achieve high prediction confidence on the poisoned training data, so as to ensure a high ASR.

2.1 Theoretical Results

To verify our observation, we first analyze the relationship between the prediction confidence on poisoned training samples and the ASR on poisoned testing inputs. Assume the data point (X, Y) is sampled uniformly at random from $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} = \mathbb{R}^d$ and $\mathcal{Y} = [c]$. Take multi-class learning as a typical example, which aims to optimize the following risk,

$$\mathcal{R}(\mathbf{g}) = \mathbb{E}_{(X,Y)\sim p(x,y)}[\ell(\mathbf{g}(X), \mathbf{e}^Y)]$$

with a minimizer $\mathbf{g}^*$: $\mathcal{X} \to \mathbb{R}^c$. Here, $\mathbf{e}^Y$ is the standard canonical vector with the Y-th entry to be one and all others zero. ℓ is often defined to be the cross-entropy loss.

Given a backdoor attack which poisons some training data in different labels to the target label t, and using a_z to denote the poisoning rate in label z, the above risk becomes

$$\begin{aligned}\mathcal{R}_a(\mathbf{g}) \quad = \quad & \sum_{z\in[c]} a_z \cdot p(Y=z)\mathbb{E}_{X\sim p(x|y=z)}[\ell(\mathbf{g}(X), \mathbf{e}^t)] + \\ & \sum_{z\in[c]} (1-a_z)p(Y=z)\mathbb{E}_{X\sim p(x|y=z)}[\ell(\mathbf{g}(X), \mathbf{e}^z)]\end{aligned}$$

with a minimizer denoted as $\mathbf{g}_a^*$. Here, we assume that the trigger involves very small perturbations and can be approximately ignored in the above equation. Based on the above two equations, we have the following theorem.

Theorem 1 *For an effective black-box backdoor attack that does not change the prediction of clean data, with a probability at least* $1-\delta$

$$\begin{aligned}\mathcal{R}(\mathbf{g}_a^*) - \mathcal{R}(\mathbf{g}^*) \le 4\mathfrak{R}_n(\ell\circ\mathcal{G}) + 2M\sqrt{\frac{\log(2/\delta)}{2n}} + \\ 2\sum_{z\in[c]} a_z \sum_{x_i\in D_z} |\ell(\mathbf{g}^*(x_i), \mathbf{e}^z) - \ell(\mathbf{g}_a^*(x_i), \mathbf{e}^t)|,\end{aligned}$$

where $\mathfrak{R}_n(\cdot)$ *is the Rademacher Complexity of a function class for sample size* n*,* M *is the upper bound of the loss function* ℓ*, and* D_z *is the subset of poisoned training data with label* z*.*

PROOF. See the supplementary material.

Remarks. $\mathcal{R}(\mathbf{g}^*)$ and $\mathcal{R}(\mathbf{g}_a^*)$ are the lowest risks we could get on the given data distribution and function class. Higher $\mathcal{R}(\mathbf{g}_a^*) - \mathcal{R}(\mathbf{g}^*)$ essentially indicates that the trojaned classifier $\mathbf{g}_a^*$ incurs a higher risk than the clean classifier $\mathbf{g}^*$. In other words, higher $\mathcal{R}(\mathbf{g}_a^*) - \mathcal{R}(\mathbf{g}^*)$ indicates the possibility of higher ASR. In the above theorem, $\ell(\mathbf{g}^*(x_i), \mathbf{e}^z)$ stands for the empirical loss of data sample x_i before poisoning, and $\ell(\mathbf{g}_a^*(x_i), \mathbf{e}^t)$ stands for its empirical loss after poisoning. Therefore, the above theorem gives the insight that, when the prediction confidence of a data sample before poisoning is close to that after poisoning (i.e., $\ell(\mathbf{g}^*(x_i), \mathbf{e}^z)$ is close to $\ell(\mathbf{g}_a^*(x_i), \mathbf{e}^t)$), and the number of poisoned data $\sum_z a_z|D_z|$ is relatively small,[1] it is less likely to have an effective backdoor attack with a high ASR.

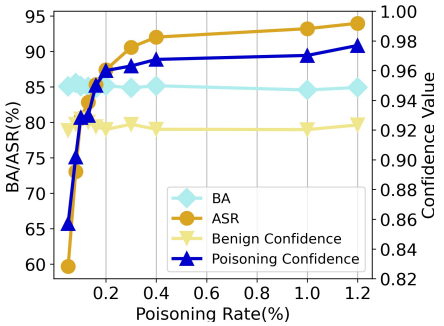
2.2 Empirical Results

To further verify our observation, we provide some empirical evidence here. Specifically, we show that, when we slightly increase the poisoning rate to ensure a highly effective backdoor attack, the confidence of poisoned data becomes significantly higher than that of clean data. We conduct experiments with various backdoor attacks including BADNET (Gu, Dolan-Gavitt, and Garg 2017), TROJANNN (Liu et al. 2018), CL (Turner, Tsipras, and Madry 2018), SIG (Barni, Kallas, and Tondi 2019), REFOOL (Liu et al. 2020), and FTROJAN (Wang et al. 2022). The results are shown in Figure 1.

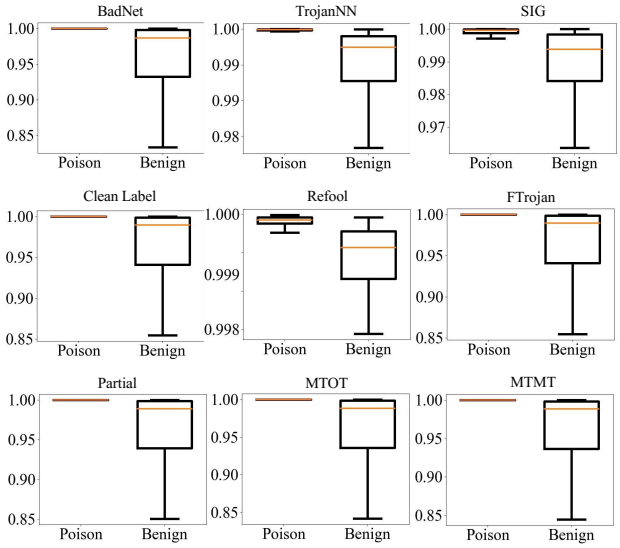
In Figure 1a, we vary the poisoning rate and show the average results of benign accuracy (BA) on clean inputs, ASR on poisoned inputs, and prediction confidences of poisoned training data and clean training data in the target label. For brevity, we show the results of BADNET here, and similar results are observed on the other attacks. We can first observe that BA and prediction confidence of clean data are relatively stable indicating the stealthiness of the backdoor attack. Second, as the poisoning rate increases, along with the increase of ASR, the average prediction confidence of poisoned data becomes significantly higher than that of clean data. For example, when the ASR increases up to 85%, the average prediction confidence of poisoned data is already 3.3% higher than that of clean data (0.95 vs. 0.92).

In Figure 1b, we further show the distributions of prediction confidences of poisoned data and clean data in the target label. To ensure the effectiveness of each backdoor attack, in this experiment, we gradually increase the poisoning rate by 0.1% each step until the ASR is above 95%. As we can see, the confidences of most poisoned samples are significantly higher than those of clean samples, for all the six backdoor attacks and three advanced attacking types (i.e., partial attack, MTOT attack, and MTMT attack).

[1]Note that a large amount of poisoned data may degenerate the prediction accuracy of benign inputs, making the attack less stealthy.



(a) Prediction confidence and BA/ASR vs. poisoning rate.



(b) Prediction confidences of poisoned and clean data.

Figure 1: The empirical results for our observation: (a) as the poisoning rate increases, the average prediction confidence of poisoned data becomes significantly higher than that of clean data; (b) the prediction confidences of most poisoned data are significantly higher than those of clean data.

Overall, the above theoretical and empirical results show that it is less likely for a backdoor attack to be highly effective, while keeping the prediction confidence of poisoned data to be close to that of clean data.

3 The Proposed Defense

3.1 Overview

Effectively making use of the above observation is non-trivial. A straightforward way is to sample some high-confidence and low-confidence data and directly compare their distributions. However, such comparison is less effective as the distributions between clean samples and poisoned samples are difficult to distinguish, especially when the trigger involves only a small perturbation (which is also supported by our results in Section 4.2). To overcome this issue, we propose a technique to magnify their difference via using the shortcut nature of triggers.

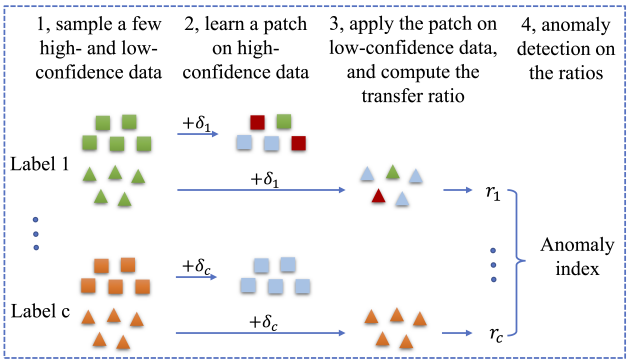


Figure 2: The overview of DTINSPECTOR. Squares and triangles represent high-confidence and low-confidence samples, respectively. Colors indicate the prediction labels. The key insight is that the transfer ratio of infected labels (e.g., Label c) would be significantly lower than that of clean labels (e.g., Label 1).

The overview of DTINSPECTOR is shown in Figure 2. Specifically, DTINSPECTOR first sorts the training data by confidence for each label z, and selects top-K samples from the high-confidence area and bottom-K samples from the low-confidence area. Second, DTINSPECTOR learns a small *patch* δ using only the high-confidence samples, with the goal of changing their predictions to any other labels but the current label z. Third, DTINSPECTOR applies the learned patch δ to the low-confidence data, and records the *transfer ratio* r meaning the ratio of samples whose prediction labels changed after applying patch δ. Finally, DTINSPECTOR performs anomaly detection on all the transfer ratios to spot the trojaned model and the infected label. If the transfer ratios of certain labels are significantly small, the model is suspected of being trojaned.

The key intuition of DTINSPECTOR is as follows. For a clean label, the learned patch from the high-confidence data can be seen as a universal adversarial perturbation (Moosavi-Dezfooli et al. 2017), and thus the transfer ratio of low-confidence data will also be high (e.g., Label 1 in Figure 2). In contrast, for an infected label, the learned patch tends to destroy the trigger due to the shortcut nature of triggers. That is, a trigger is a shortcut that leads the input to the target label (Geirhos et al. 2020; Li et al. 2021c), and destroying this shortcut is the easiest way to change the predictions of the high-confidence data (e.g., back to the original labels). This patch will not largely affect the predictions of low-confidence data that does not contain triggers, making the transfer ratio significantly lower for low-confidence samples in the infected label (e.g., Label c in Figure 2).

Note that DTINSPECTOR is also useful to mitigate the effect of backdoor. We can superimpose the learned patch to each sample from the infected label and remove the sample if it is classified to a different label. We then retrain the model, and find that such a strategy can significantly lower down the ASR, and meanwhile preserves the accuracy in most cases (see Section 4.2).

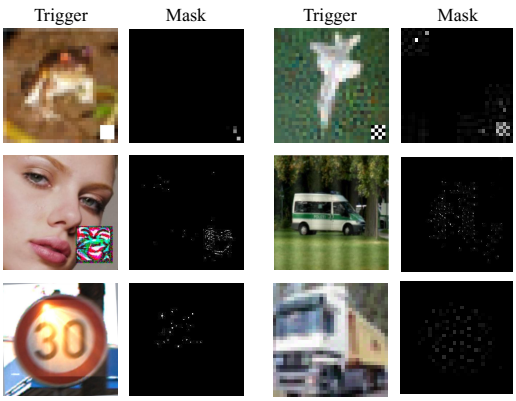


Figure 3: The poisoned images and the learned patches by DTINSPECTOR. The six examples correspond to six attacks, i.e., BADNET, TROJANNN, CL, SIG, REFOOL, and FTROJAN, respectively.

3.2 Patch Learning

For patch learning, we define a patch as a tuple (M, P), where M is the 2D mask and P contains the 3D patch pixels. We then learn the patch that can lead the high-confidence data to predict a different label. That is, for a certain label z and high-confidence data D_h, we aim to optimize the following objective function,

$$\max_{M,P} \quad \sum_{x \in D_h} f(\mathbf{g}(x'), \mathbf{e}^z) - \lambda ||M||_1, \quad x \in D_h,$$
$$\text{s.t.} \quad x' = (1 - M) \odot x + M \odot P, \tag{1}$$

where M controls the area to be patched, P controls the value of the patch, and $\odot$ is element-wise multiplication. M_{ij} ranges from 0 to 1 indicating how much of the original pixel values will be retained or refilled with P. For function f, since we aim to lead the prediction to any other label but the current z, we use the mean square error on the z-th dimension. In other words, the above objective encourages the z-th dimension of $\mathbf{g}(x')$ to lean towards zero. λ is used to balance the importance of M's L_1 norm. Smaller λ tends to yield larger patches. λ will be dynamically adjusted during the optimization process in order to keep a relatively high transfer ratio (e.g., no less than 95%).

Some poisoned images and the corresponding learned patches (masks) are shown in Figure 3. We can observe that the learned patches successfully identify the trigger areas. Note that the learned patch does not necessarily need to erase the entire trigger, but only needs to destroy the key pixels, so as to lead the predictions of poisoned images from the target label back to their original labels.

3.3 Anomaly Detection

When a patch is learned for a label z, we apply it to the corresponding low-confidence data D_l, and calculate the transfer ratio r. Specifically, r is defined as $r = \frac{\sum_{x \in D_l} \mathbb{I}(g(x') \neq z)}{n}, \quad x \in D_l$, where $g(x)$ is the predicted label of x, and $\mathbb{I}$ is an indicator function. Based on the transfer ratio r for each label, we use the MAD outlier detection method to compute the *anomaly index*. Anomaly index larger than a certain threshold means that the corresponding model is trojaned. In this work, we determine the threshold following the three-sigma rule suggested by existing work (Iglewicz and Hoaglin 1993).[2] Further, if all the transfer ratios are greater than 90% for all labels, we consider the model to be clean even when the anomaly index is above the threshold. This could happen when, e.g., one transfer ratio is 95% and the rest transfer ratios are all 100%.

Remarks. Our approach is similar to NC (Wang et al. 2019), but bears several subtle and important differences. First, DTINSPECTOR and NC are built upon different observations. NC assumes that the triggers are of small size while we focus on the prediction behaviors related to confidence. Second, DTINSPECTOR and NC use different optimization objective functions. NC learns a shortcut on clean data that leads to a specific label. In contrast, DTINSPECTOR learns a patch on training data (can be both clean or poisoned) in a certain label and the patch leads the prediction to any other labels but the current one. Third, DTINSPECTOR and NC compute anomalies on different metrics. NC applies anomaly detection to the L_1 norms of the reversed triggers, while DTINSPECTOR applies anomaly detection on the transfer ratios of low-confidence data.

Dataset	Train/Test	Label	Classifier
CIFAR10	50,000/10,000	10	6 Conv + 2 Dense
GTSRB	4772/293	13	ResNet34
PubFig	12,800/3,200	16	ResNet50
ImageNet	20,567/800	16	ResNet50

Table 1: Summary of datasets and classifiers.

4 Evaluation

4.1 Experimental Setup

Datasets. We use four commonly-studied datasets in our experiments including CIFAR10 (Krizhevsky, Hinton et al. 2009), GTSRB (Stallkamp et al. 2011), ImageNet (Deng et al. 2009), and PubFig (Kumar et al. 2009). All the datasets are publicly available. For GTSRB, we use its subset with 13 labels provided by (Liu et al. 2020). For PubFig and ImageNet, we randomly select a subset of them and each dataset contains 16 labels. We also train different neural network models for these datasets, and the details can be found in Table 1.

Backdoor Attacks. We evaluate six black-box backdoor attacks,[3] including BADNET (Gu, Dolan-Gavitt, and Garg 2017), TROJANNN (Liu et al. 2018), CL (Turner, Tsipras, and Madry 2018), SIG (Barni, Kallas, and Tondi 2019), REFOOL (Liu et al. 2020), and FTROJAN (Wang et al. 2022). We consider different combinations of the six attacks and the four datasets. To ensure an effective attack, we gradu-

[2]The three-sigma rule suggests an anomaly index larger than 3.5 as the existence of anomalies.

[3]We consider black-box attacks and thus other recent attacks, e.g., (Nguyen and Tran 2020, 2021; Souri et al. 2022), that require the control of the training process are not included.

Attack	Dataset	Original Acc. (%)	Benign Acc. (%)	ASR (%)
BADNET	CIFAR10	85.85	85.05	96.34
TROJANNN	PubFig	83.88	79.75	98.25
CL	CIFAR10	85.85	84.06	96.12
SIG	ImageNet	84.00	77.00	97.75
REFOOL	GTSRB	99.32	98.07	99.98
FTROJAN	CIFAR10	85.85	85.47	99.95

Table 2: The evaluated backdoor attacks.

ally increase the poisoning rate by 0.1% each step until the ASR is above 95%. The original accuracy of the model, the benign accuracy on clean inputs, and the ASR are shown in Table 2, where we show the results of BADNET on CIFAR10, TROJANNN on PubFig, CL on CIFAR10, SIG on ImageNet, REFOOL on GTSRB, and FTROJAN on CIFAR10 for brevity.

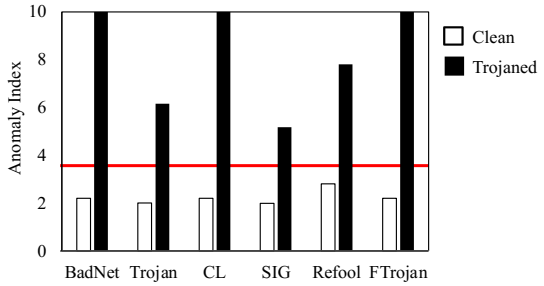
Backdoor Defenses. We compare with four state-of-the-art backdoor defenses that aim to detect whether a given model is trojaned, i.e., NC (Wang et al. 2019), ABS (Liu et al. 2019), ULP (Kolouri et al. 2020), and MNTD (Xu et al. 2021). For all of them, we use the open source code provided by their authors, and use their default settings.[4] For DTINSPECTOR, there are two key hyper-parameters, i.e., λ and $|D_h|$ in Eq. (1). For λ, we initialize it to 0.0001 and dynamically adjust it to a proper value following (Wang et al. 2019). For the sampling size $|D_h|$, we empirically found that 50 high-confidence samples and 50 low-confidence samples are sufficient, and thus set it to 50 by default (i.e., $|D_h| = |D_l| = 50$). We will evaluate the performance of DTINSPECTOR as this parameter varies. The experiments are run on a machine with 20-cores Intel i9-10900KF CPU, 256GB RAM, and one NVIDIA GeForce RTX3090 GPU.

4.2 Experimental Results

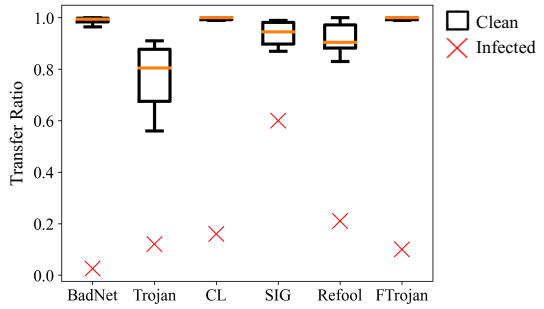
(A) Detection against backdoor attacks. We first evaluate the detection accuracy of DTINSPECTOR under the 21 attack-dataset combinations. The results show that DTINSPECTOR can accurately distinguish the trojaned/clean models as well as identify the infected labels in 20 out of the 21 combinations (the complete results are included in the supplementary material). As a comparison reference, existing attack REFOOL (Liu et al. 2020) has reported that it cannot be detected by NC, and FTROJAN (Liu et al. 2020) has reported that it cannot be detected by NC and ABS. The only exception is the REFOOL and PubFig combination. The reason is that the trigger of REFOOL covers the most influential areas (e.g., the central part) in the face images, and patching such areas may easily change the predictions. This limitation can be further addressed by, e.g., applying image restoration techniques such as GANs, and we leave this as future work.

Here, we further show in Figure 4 the results of BADNET on CIFAR10, TROJANNN on PubFig, CL on CIFAR10, SIG on ImageNet, REFOOL on GTSRB, and FTROJAN on CIFAR10. We can observe that the anomaly indices of trojaned models are significantly larger than that of clean models, and DTINSPECTOR successfully distinguishes them. Additionally, the infected label is identified as an outlier (denoted by red crosses), meaning that DTINSPECTOR can also successfully identify the infected label.

[4]Note that ABS and ULP only provide implementations for the CIFAR10 dataset. That is, ABS provides executable binaries and ULP provides the pre-trained meta-classifier for CIFAR10.



(a) Detecting trojaned models



(b) Detecting infected labels

Figure 4: The detection results against six existing backdoor attacks. DTINSPECTOR can successfully detected trojaned models and the infected labels.

(B) Detection against advanced types of backdoor attacks. Next, we show the detection results against partial attack, MTOT attack, and MTMT attack. Note that we build these advanced attacks upon the basic BADNET attack, which has been shown to be detected by all the compared defenses. The detection results of different defenses are shown in Table 3. We can see that DTINSPECTOR can detect these advanced backdoor attacks. In contrast, ABS fails to detect them (the REASR scores from both feature space and pixel space are zeros). This is probably due to the fact that ABS analyzes only one neuron and these attacks may involve multiple neurons. ULP and MNTD cannot successfully detect the MTMT attack, and the possible reason is that the trained meta-classifier cannot generalize well to the complex model behavior after MTMT attack. NC also cannot detect the MTMT attack. The reason is that NC identifies multiple reversed triggers in this case, which tends to decrease the degree of abnormality. For the MTOT attack, we further show the detected triggers in Figure 5. We can see that although NC can detect the attack, it identifies only one trigger; in contrast, DTINSPECTOR can detect all the three triggers since all of them are needed to flip the predictions.

Type	NC	ABS	ULP	MNTD	DTINSPECTOR
Partial	Y	N	Y	Y	Y (16.19)
MTOT	Y	N	Y	Y	Y (36.74)
MTMT	N	N	N	N	Y (62.72)

* For all the tables, unless otherwise stated, 'Y' means the model is identified as trojaned and 'N' indicates otherwise; numbers in the parentheses are anomaly indices, and the model is considered to be trojaned if the index exceeds 3.5.

Table 3: Detection results against advanced types of backdoor attacks. Compared to the competitors, DTINSPECTOR successfully detect all such attacks.

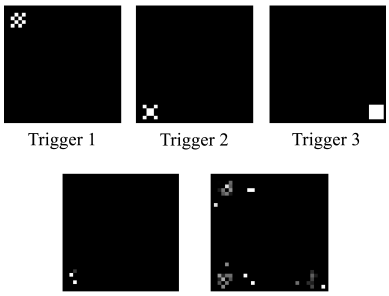


Figure 5: Detected triggers by NC and DTINSPECTOR under the MTOT attack. NC reverse-engineers one trigger while DTINSPECTOR finds all the three triggers (patches).

(C) Detection against unseen triggers. Next, we further compare DTINSPECTOR with ULP and MNTD. As mentioned before, ULP and MNTD may not generalize well to unseen triggers that are significantly different from those in the training data. To this end, we still use the BADNET attack on CIFAR10, and define two triggers that are not included in the training data of ULP and MNTD, i.e., a randomly colored 8×8 square and a red 12×12 square. The detection results are shown in Table 4. We can see that both ULP and MNTD cannot detect such attacks, while DTINSPECTOR can still detect them.

(D) Comparison with straightforward data inspection. Here, we compare DTINSPECTOR with a straightforward data inspection method. This method is also built on the prediction confidence observation, but directly compares the distributions of high-confidence and low-confidence data instead of using the proposed method described in Section 3. To setup the method, we use the same set of high-confidence and low-confidence data with DTINSPECTOR, but calculate the KL divergence between their averaged representations from the model's penultimate layer (normalized by softmax) before applying the anomaly detection. The results are shown in Table 5. We can observe that the data inspection method has a high mis-detection rate especially for clean models. This is due to the fact that the trigger is usually of small size and thus difficult to distinguish. In contrast, DTINSPECTOR performs much better as it uses a distribution transfer technique to magnify the differences between clean data and poisoned data.

New Trigger	ULP	MNTD	DTINSPECTOR
8×8 colored square	N	N	Y (122.00)
12×12 red square	N	N	Y (39.23)

Table 4: Comparisons with ULP and MNTD on unseen triggers. DTINSPECTOR can detect the trojaned models while ULP and MNTD cannot.

Attack	Data Inspection		DTINSPECTOR	
	Clean	Trojaned	Clean	Trojaned
BADNET	Y (6.95)	Y (31.90)	N (2.47)	Y (199.65)
TROJANNN	N (1.88)	Y (14.70)	N (2.02)	Y (4.47)
CL	Y (6.95)	Y (38.85)	N (2.47)	Y (77.34)
SIG	N (1.76)	N (2.82)	N (0.98)	Y (7.86)
REFOOL	Y (27.36)	Y (13.47)	N (1.21)	Y (6.34)
FTROJAN	Y (6.95)	Y (5.54)	N (2.47)	Y (144.50)
Partial	Y (6.95)	Y (152.48)	N (2.47)	Y (16.20)
MTOT	Y (6.95)	Y (21.42)	N (2.47)	Y (36.74)
MTMT	Y (6.95)	Y (48.78)	N (2.47)	Y (62.72)

Table 5: Comparisons with data inspection method. The wrong detection results are underlined. Data inspection results in high mis-detection especially for the clean models.

(E) Comparison with input filtering method. We also adapt an input filtering method STRIP (Gao et al. 2019), which is originally proposed to identify poisoned inputs at the inference stage. We use the default setting,[5] and apply it on the training data. The results show that STRIP always mis-classifies a relatively large set of clean data as poisoned. This makes it less effective when the poisoning rate is low. For example, on the clean CIFAR10 data, STRIP identifies 163 poisoned samples. When we inject 250 poisoned samples with poisoning rate 0.5% (ASR is larger than 90%), STRIP identifies 225 poisoned samples, 65 out of which are clean data.

(F) Sensitivity to trigger size. In Eq. (1), we use L_1 norm to restrict the size of the patch, which might cause our detection method to be sensitive to the trigger size. Here, we perform a sensitivity study by varying the trigger size, and the results show that DTINSPECTOR is more robust to trigger size than NC (see the supplementary material for details).

(G) Sensitivity to sampling size. Another parameter of DTINSPECTOR is the sampling size, i.e., the number of high- and low-confidence samples. We also evaluate the sensitivity of this parameter and find that DTINSPECTOR is relatively robust to it in a wide range (see the supplementary material for details).

(H) Detection against adaptive attacks. Next, we conduct an adaptive attack where the attacker can control the poisoning rate in a range to intentionally lower down the prediction confidences of poisoned samples. The results show that DTINSPECTOR is still effective as long as the backdoor attack is effective (e.g., the ASR is above 70%; see the sup-

[5]We also tune STRIP's two parameters of entropy boundary and clean data size, and found little difference.

Attack	Original Acc. (%)	Benign Acc. (%)	ASR (%)	Retrained Benign Acc. (%)	Retrain ASR (%)
BADNET	85.85	85.05	96.34	84.65	7.66
TROJANNN	83.88	79.75	98.25	81.25	14.25
CL	85.85	84.06	96.12	84.12	8.93
SIG	84.00	77.00	97.75	82.88	0.02
REFOOL	99.32	98.07	99.98	97.92	0.11
Partial	85.85	85.65	90.80	84.51	1.90
MTOT	85.85	84.53	96.05	84.40	9.23
MTMT	85.85	84.26	98.21	70.05	7.06

Table 6: The mitigation results of DTINSPECTOR. After removing the suspected data and retraining the model, the ASR can be reduced to less than 15%, while the benign accuracy can be preserved in most cases.

plementary material for details).

(I) Mitigation results using the proposed method. As mentioned before, the learned patch of DTINSPECTOR can also be used to remove the backdoor. Specifically, we consider the label whose transfer ratio is an outlier as the infected label. For all the training samples belonging to this infected label, we superimpose the patch to each sample and remove the sample if it is classified to a label different from the original label. Otherwise, the sample is retained. We then retrain the model on the cleaned dataset. The results are shown in Table 6.

We can see that our mitigation method can mitigate most of the attacks (significantly lowering down the ASR), and meanwhile preserves the accuracy in most cases. One exception is from the MTMT attack, where the accuracy decreases with a relatively large margin. This is due to the fact MTMT attack infects multiple labels; therefore, we have to run the sample removing multiple times for MTMT attack, which significantly shrinks the training data size. This issue can be mitigated by, again, applying image restoration techniques to fix some of the poisoned images instead of directly deleting them, which is left as future work.

5 Related Work

Backdoor Attacks. Starting from the seminal work (Gu, Dolan-Gavitt, and Garg 2017; Chen et al. 2017), various backdoor attacks have been proposed, which can be divided into black-box ones and white-box ones. This work focuses on detecting against black-box attacks. Another major line of such attacks is to improve the robustness against backdoor defenses (Yao et al. 2019; Saha, Subramanya, and Pirsiavash 2020; Shokri et al. 2020; Lin et al. 2020; He et al. 2021). Additionally, some researchers propose dynamic attacks that generate different triggers for different inputs (Salem et al. 2020; Nguyen and Tran 2020), clean-label attacks that do not change the labels of training data (Turner, Tsipras, and Madry 2018; Barni, Kallas, and Tondi 2019; Zhao et al. 2020b; Liu et al. 2020; Nguyen and Tran 2021), physical attacks that use physical objects as triggers (Wenger et al. 2021), invisible attacks that aim to hide the triggers (Nguyen and Tran 2021; Li et al. 2021b; Souri et al. 2022; Wang et al. 2022), and data-free attacks that do not need the access to training data (Liu et al. 2018; Tang et al. 2020; Costales et al. 2020; Pang et al. 2020; Qi et al. 2022).

Backdoor Defenses. Existing backdoor defenses can be divided into four categories, i.e., model inspection, data inspection, input filtering, and backdoor removal. Our defense belongs to the first category, which aims to detect whether a given model is trojaned. Examples include those search for triggers (e.g., (Wang et al. 2019; Chen et al. 2019b; Guo et al. 2020; Shen et al. 2021), analyze neuron activations (Liu et al. 2019; Ma and Liu 2019), and build meta-classifiers (Xu et al. 2021; Kolouri et al. 2020). Different from these work, our defense is built on an observation that is potentially universal for black-box backdoor attacks, making it orthogonal and complementary to existing defenses.

Defenses in the rest three categories are built upon the premise that the given model is trojaned. For example, data inspection methods aim to distinguish poisoned training samples from clean ones given that the current model is trojaned (Tran, Li, and Mądry 2018; Chen et al. 2019a; Hayase et al. 2021); input filtering methods aim to detect whether an input has been corrupted or how to erase triggers in the input (Cohen, Rosenfeld, and Kolter 2019; Udeshi et al. 2019; Gao et al. 2019; Doan, Abbasnejad, and Ranasinghe 2020); backdoor removal methods aim to remove the backdoor in the models (Liu, Dolan-Gavitt, and Garg 2018; Zhao et al. 2020a; Li et al. 2021a; Wu and Wang 2021; Guan et al. 2022; Huang et al. 2020). However, how to effectively adapt them to detect whether a model is trojaned still needs future exploration.

6 Conclusion

In this paper, we have proposed a new backdoor defense DTINSPECTOR. Our key insight is that an effective backdoor attack usually results in high prediction confidence on the poisoned training data, so as to ensure a high ASR. We provide both theoretical and empirical evidence for this observation, and then propose a distribution transfer technique to distinguish trojaned models from clean ones, using the shortcut nature of triggers. Extensive experiments demonstrate that the proposed defense: 1) can accurately detect the trojaned model as well as the infected label, and 2) outperforms existing defenses in terms of robustness to trigger size and effectiveness against advanced attacks and unseen triggers. In the future, we plan to extend our idea into natural language models and more computer vision tasks.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Grant #62025202), and the Collaborative Innovation Center of Novel Software Technology and Industrialization. Miao Xu is supported by the NVIDIA Academic Hardware Grant Program and the Australian Research Council Discovery Early Career Research Award (DE230101116). Ting Wang is partially supported by the National Science Foundation under Grant No. 1953893, 1951729, 2119331, and 2212323. Yuan Yao is the corresponding author.

References

Barni, M.; Kallas, K.; and Tondi, B. 2019. A new backdoor attack in CNNs by training set corruption without label poisoning. In *ICIP*, 101–105.

Chen, B.; Carvalho, W.; Baracaldo, N.; Ludwig, H.; Edwards, B.; Lee, T.; Molloy, I.; and Srivastava, B. 2019a. Detecting backdoor attacks on deep neural networks by activation clustering. In *Workshop on Artificial Intelligence Safety*.

Chen, H.; Fu, C.; Zhao, J.; and Koushanfar, F. 2019b. DeepInspect: A Black-box Trojan Detection and Mitigation Framework for Deep Neural Networks. In *IJCAI*.

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

Cohen, J.; Rosenfeld, E.; and Kolter, Z. 2019. Certified adversarial robustness via randomized smoothing. In *ICML*, 1310–1320. PMLR.

Costales, R.; Mao, C.; Norwitz, R.; Kim, B.; and Yang, J. 2020. Live Trojan attacks on deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 796–797.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*, 248–255.

Doan, B. G.; Abbasnejad, E.; and Ranasinghe, D. C. 2020. Februus: Input purification defense against trojan attacks on deep neural network systems. In *ACSAC*, 897–912.

Gao, Y.; Doan, B. G.; Zhang, Z.; Ma, S.; Zhang, J.; Fu, A.; Nepal, S.; and Kim, H. 2020. Backdoor attacks and countermeasures on deep learning: A comprehensive review. *arXiv preprint arXiv:2007.10760*.

Gao, Y.; Xu, C.; Wang, D.; Chen, S.; Ranasinghe, D. C.; and Nepal, S. 2019. Strip: A defence against trojan attacks on deep neural networks. In *ACSAC*, 113–125.

Geirhos, R.; Jacobsen, J.-H.; Michaelis, C.; Zemel, R.; Brendel, W.; Bethge, M.; and Wichmann, F. A. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11): 665–673.

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

Guan, J.; Tu, Z.; He, R.; and Tao, D. 2022. Few-shot Backdoor Defense Using Shapley Estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 13358–13367.

Guo, W.; Wang, L.; Xing, X.; Du, M.; and Song, D. 2020. Towards Inspecting and Eliminating Trojan Backdoors in Deep Neural Networks. In *ICDM*.

Hayase, J.; Kong, W.; Somani, R.; and Oh, S. 2021. SPECTRE: Defending Against Backdoor Attacks Using Robust Statistics. In *ICML*.

He, Y.; Shen, Z.; Xia, C.; Hua, J.; Tong, W.; and Zhong, S. 2021. RABA: A Robust Avatar Backdoor Attack on Deep Neural Network. *arXiv preprint arXiv:2104.01026*.

Huang, K.; Li, Y.; Wu, B.; Qin, Z.; and Ren, K. 2020. Backdoor Defense via Decoupling the Training Process. In *ICLR*.

Iglewicz, B.; and Hoaglin, D. C. 1993. *How to detect and handle outliers*, volume 16.

Kolouri, S.; Saha, A.; Pirsiavash, H.; and Hoffmann, H. 2020. Universal litmus patterns: Revealing backdoor attacks in cnns. In *CVPR*, 301–310.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Kumar, N.; Berg, A. C.; Belhumeur, P. N.; and Nayar, S. K. 2009. Attribute and simile classifiers for face verification. In *ICCV*, 365–372.

Li, Y.; Koren, N.; Lyu, L.; Lyu, X.; Li, B.; and Ma, X. 2021a. Neural Attention Distillation: Erasing Backdoor Triggers from Deep Neural Networks. In *ICLR*.

Li, Y.; Li, Y.; Wu, B.; Li, L.; He, R.; and Lyu, S. 2021b. Invisible Backdoor Attack With Sample-Specific Triggers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 16463–16472.

Li, Y.; Lyu, X.; Koren, N.; Lyu, L.; Li, B.; and Ma, X. 2021c. Anti-backdoor learning: Training clean models on poisoned data. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34.

Lin, J.; Xu, L.; Liu, Y.; and Zhang, X. 2020. Composite Backdoor Attack for Deep Neural Network by Mixing Existing Benign Features. In *CCS*, 113–131.

Liu, K.; Dolan-Gavitt, B.; and Garg, S. 2018. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *RAID*, 273–294.

Liu, Y.; Lee, W.-C.; Tao, G.; Ma, S.; Aafer, Y.; and Zhang, X. 2019. ABS: Scanning neural networks for back-doors by artificial brain stimulation. In *CCS*, 1265–1282.

Liu, Y.; Ma, S.; Aafer, Y.; Lee, W.; Zhai, J.; Wang, W.; and Zhang, X. 2018. Trojaning Attack on Neural Networks. In *NDSS*.

Liu, Y.; Ma, X.; Bailey, J.; and Lu, F. 2020. Reflection backdoor: A natural backdoor attack on deep neural networks. In *ECCV*, 182–199. Springer.

Ma, S.; and Liu, Y. 2019. Nic: Detecting adversarial samples with neural network invariant checking. In *Proceedings of the 26th Network and Distributed System Security Symposium (NDSS 2019)*.

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

Nguyen, T. A.; and Tran, A. 2020. Input-Aware Dynamic Backdoor Attack. In *NeurIPS*.

Nguyen, T. A.; and Tran, A. T. 2021. WaNet-Imperceptible Warping-based Backdoor Attack. In *International Conference on Learning Representations (ICLR)*.

Pang, R.; Shen, H.; Zhang, X.; Ji, S.; Vorobeychik, Y.; Luo, X.; Liu, A.; and Wang, T. 2020. A tale of evil twins: Adversarial inputs versus poisoned models. In *CCS*, 85–99.

Qi, X.; Xie, T.; Pan, R.; Zhu, J.; Yang, Y.; and Bu, K. 2022. Towards practical deployment-stage backdoor attack on deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 13347–13357.

Saha, A.; Subramanya, A.; and Pirsiavash, H. 2020. Hidden trigger backdoor attacks. In *AAAI*, 11957–11965.

Salem, A.; Wen, R.; Backes, M.; Ma, S.; and Zhang, Y. 2020. Dynamic backdoor attacks against machine learning models. *arXiv preprint arXiv:2003.03675*.

Shen, G.; Liu, Y.; Tao, G.; An, S.; Xu, Q.; Cheng, S.; Ma, S.; and Zhang, X. 2021. Backdoor Scanning for Deep Neural Networks through K-Arm Optimization. In *ICML*.

Shokri, R.; et al. 2020. Bypassing Backdoor Detection Algorithms in Deep Learning. In *EuroS&P*, 175–183.

Souri, H.; Fowl, L.; Chellappa, R.; Goldblum, M.; and Goldstein, T. 2022. Sleeper agent: Scalable hidden trigger backdoors for neural networks trained from scratch. In *Advances in Neural Information Processing Systems*, volume 35, 19165–19178.

Stallkamp, J.; Schlipsing, M.; Salmen, J.; and Igel, C. 2011. The German traffic sign recognition benchmark: a multi-class classification competition. In *IJCNN*, 1453–1460.

Tang, R.; Du, M.; Liu, N.; Yang, F.; and Hu, X. 2020. An embarrassingly simple approach for trojan attack in deep neural networks. In *KDD*, 218–228.

Tran, B.; Li, J.; and Mądry, A. 2018. Spectral signatures in backdoor attacks. In *NeurIPS*, 8011–8021.

Turner, A.; Tsipras, D.; and Madry, A. 2018. Clean-label backdoor attacks.

Udeshi, S.; Peng, S.; Woo, G.; Loh, L.; Rawshan, L.; and Chattopadhyay, S. 2019. Model agnostic defence against backdoor attacks in machine learning. *arXiv preprint arXiv:1908.02203*.

Wang, B.; Yao, Y.; Shan, S.; Li, H.; Viswanath, B.; Zheng, H.; and Zhao, B. Y. 2019. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *SP*, 707–723.

Wang, T.; Yao, Y.; Xu, F.; An, S.; Tong, H.; and Wang, T. 2022. An Invisible Black-box Backdoor Attack through Frequency Domain. In *European Conference on Computer Vision (ECCV)*.

Wenger, E.; Passananti, J.; Bhagoji, A. N.; Yao, Y.; Zheng, H.; and Zhao, B. Y. 2021. Backdoor Attacks Against Deep Learning Systems in the Physical World. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 6206–6215.

Wu, D.; and Wang, Y. 2021. Adversarial neuron pruning purifies backdoored deep models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34.

Xu, X.; Wang, Q.; Li, H.; Borisov, N.; Gunter, C. A.; and Li, B. 2021. Detecting ai trojans using meta neural analysis. In *SP*, 103–120.

Yao, Y.; Li, H.; Zheng, H.; and Zhao, B. Y. 2019. Latent backdoor attacks on deep neural networks. In *CCS*, 2041–2055.

Zhao, P.; Chen, P.-Y.; Das, P.; Ramamurthy, K. N.; and Lin, X. 2020a. Bridging mode connectivity in loss landscapes and adversarial robustness. In *ICLR*.

Zhao, S.; Ma, X.; Zheng, X.; Bailey, J.; Chen, J.; and Jiang, Y.-G. 2020b. Clean-label backdoor attacks on video recognition models. In *CVPR*, 14443–14452.