TEST-TIME DETECTION OF BACKDOOR TRIGGERS FOR POISONED DEEP NEURAL NETWORKS

Xi Li, Zhen Xiang, David J. Miller, George Kesidis

School of EECS, Pennsylvania State University

ABSTRACT

Backdoor (Trojan) attacks are emerging threats against deep neural networks (DNN). A DNN being attacked will predict to an attacker-desired *target class* whenever a test sample from any *source class* is embedded with a *backdoor pattern*, while correctly classifying clean (attack-free) test samples. Existing backdoor defenses have shown success in detecting whether a DNN is attacked and in reverse-engineering the backdoor pattern in a "post-training" scenario: the defender has access to the DNN to be inspected and a small, clean dataset collected independently, but has no access to the (possibly poisoned) training set of the DNN. However, these defenses neither catch culprits in the act of triggering the backdoor mapping, nor mitigate the backdoor attack at test-time. In this paper, we propose an "in-flight" *unsupervised* defense against backdoor attacks on image classification that 1) detects use of a backdoor trigger at *test-time*; and 2) infers the class of origin (source class) for a detected trigger example. The effectiveness of our defense is demonstrated experimentally for a wide variety of DNN architectures, datasets, and backdoor attack configurations.

***Index Terms*—** adversarial learning, backdoor attack, Trojan attack, in-flight detection, image classification

1. INTRODUCTION

Deep neural networks (DNN) have shown impressive performance in many applications, but are vulnerable to adversarial attacks [1]. Recently, backdoor attacks have been successfully launched against DNNs for image classification [2–6], speech recognition [4], text classification [7], 3D point cloud classification [8], and deep regression [9]. Typically, a backdoor attack is launched by embedding a specific backdoor pattern into a small number of training samples from one or more source classes and (mis)labeling them to an attacker-desired target class. The DNN being attacked will classify to the target class whenever a test sample originally from the source classes is embedded with the same backdoor pattern – while still correctly classifying backdoor-free samples [2, 10].

Early backdoor defenses aim to inspect the training set and detect samples embedded with the backdoor pattern [11, 12]. But they are not applicable to scenarios where the training set of the DNN is not available (*e.g.*, legacy or proprietary systems); thus they are not discussed here. A more practical *post-training* scenario assumes that the defender, *e.g.*, a downstream app user, has *no access* to the DNN's training set. Defenses for this scenario detect whether a trained DNN is backdoor attacked, infer the target class if an attack is detected, and usually reverse-engineer the backdoor pattern used by the attacker [13–17]. A post-training defender does possess a small, clean (backdoor-free) dataset collected independently – this dataset is not sufficient for training a clean DNN *from scratch* if an attack is detected.

However, existing post-training defenses cannot catch entities in the act of exploiting the backdoor mapping at test-time. In particular, while these defenses successfully detect backdoor attacks and correctly infer the attacker's target class, they neither decide whether a test sample classified to the target class is backdoor-free nor mitigate the attack by correctly classifying test samples embedded with the backdoor pattern to their true source classes (as a clean DNN will do).

In this paper, we make the following contributions: (1) We propose an *unsupervised* method that, at *test-time*, detects image backdoor triggers and infers the source class for detected backdoor trigger images, given the reverse-engineered backdoor pattern and the target class obtained from a post-training backdoor detector. While we focus on image classification here, our method can be extended to other domains. (2) Our detector requires no access to the DNN's training set nor any DNN training/fine-tuning. (3) We show the effectiveness of our detector experimentally for a wide variety of DNN architectures, datasets, and backdoor attack configurations.

2. RELATED WORK

Neural Cleanse (NC) [14] detects test images embedded with backdoor triggers by their activations on neurons that are most relevant to the (estimated) backdoor. If the input image has activations higher than a given threshold on those neurons, it is deemed a backdoor trigger image. Its performance highly depends on the choice of abnormal neurons and detection threshold. [15] proposes a black-box backdoor detection (B3D), which captures backdoor triggers based on the difference in model outputs for a test image with and without the reverse-engineered backdoor pattern embedded. The two model outputs are hypothesized to be very different for a clean input but very similar for a backdoored input. However,

for both backdoor-trigger images classified to the target class and clean target class images, embedding a backdoor trigger is expected to have little impact on the model outputs; thus they cannot be effectively distinguished by B3D. A STRong Intentional Perturbation (STRIP) method proposed in [18] linearly blends test images with a few clean images and detects backdoor triggers based on the average entropy of model posteriors of these blended images. A small entropy indicates a backdoor-trigger input. However, this method is sensitive to model complexity, as will be shown in Sec.5.

For inferring the class of origin for a detected backdoor-trigger image, NC [14] proposes to patch the poisoned DNN by fine-tuning the DNN on 10% of the original (backdoor-free) training set, 20% of which are embedded with the reverse engineered backdoor pattern and correctly labeled. However, it is unreasonable to assume the defender has access to the clean training set, which is inconsistent with the post-training scenario.

3. THREAT MODEL AND DEFENSE ASSUMPTIONS

Classification domain: Like most existing works [11–18], we focus on image classification for simplicity; our method is easily extended to domains other than images.

Attacker's goals: An attacker aims to have the DNN learn to classify to a target class $t \in \mathcal{C}$, whenever a test image from any source class $c \in \mathcal{S}_A \subseteq \mathcal{C} \setminus \{t\}$ is embedded with a specific backdoor pattern; while not degrading the DNN's accuracy on backdoor-free test images. Here, $\mathcal{C}$ is the category space of the classification domain. For example, in a face recognition system, a backdoored model will misclassify a person wearing a pair of glasses (backdoor pattern) as one owning a higher privilege (the target class) [3]. Meanwhile, users without these specific glasses are correctly classified.

Attacker's knowledge: The attacker has full knowledge of $\mathcal{C}$ and the ability to collect valid images from source classes $\mathcal{S}_A$. But the attacker has no specific knowledge about any defenses that may be deployed to detect the attack.

Attack strategy: We consider classical backdoor attacks launched by poisoning the DNN's training set [2, 3]. The attacker embeds the same backdoor pattern that will be used at test-time into a small set of images collected from $\mathcal{S}_A$, (mis)labels them to the target class t, and injects them into the DNN's training set. The backdoor pattern can be an imperceptible additive perturbation bounded by its L_p norm [13], or a perceptible (but hopefully scene-plausible) small patch embedded in (or blended with) an image [2–4].

Defender's goals: Given any *single* test image predicted to the target class t inferred by a post-training defense, our in-flight detector *aims* to: 1) detect whether the image contains the backdoor pattern (*i.e.*, a backdoor trigger), and, if so, 2) infer its true class of origin (source class).

Defender's knowledge: The defender has access to: 1) the DNN detected as attacked, together with the target class t[1] inferred by the post-training defense [13, 14]; and 2) a (small) set of clean images from all classes of $\mathcal{C}$. The defender does require the backdoor pattern estimated by the same post-training defense, but does not make any assumptions about the type, shape, or location of the attacker's backdoor pattern. Thus, our defense can be coupled with any post-training defense and addresses a variety of backdoor patterns. Finally, our defender has no knowledge of the source classes involved in the attack, nor enough legitimate data or computational resources to train a clean DNN *from scratch*.

4. METHODOLOGY

4.1. Notation

We denote the DNN (for which a post-training defense has detected an attack) by $f : \mathcal{X} \to \mathcal{C}$, where $\mathcal{X}$ is the input (image) space and $\mathcal{C}$ is the label space. We denote f_L as the output of any internal layer L. Let t be the detected target class. Then, the actual source classes $\mathcal{S}_A$ involved in the attack, unknown to our defender, is a subset of $\mathcal{S} = \mathcal{C} \setminus \{t\}$, i.e., all classes except the target class. We define $\mathcal{D}_{\text{Defense}} = \bigcup_{c \in \mathcal{C}} \mathcal{D}_c$ as the clean dataset possessed by the defender, where $\mathcal{D}_c$ contains images labeled to class c. Backdoor patterns of different types may be crafted and embedded in images in very differently ways. For simplicity, we use a universal notation Δ to denote a backdoor pattern irrespective of its type. Moreover, we define the embedding function associated with Δ as $g(\cdot, \Delta) : \mathcal{X} \to \mathcal{X}$, such that a clean image $\boldsymbol{x} \in \mathcal{X}$ embedded with Δ can be written as $\tilde{\boldsymbol{x}} = g(\boldsymbol{x}, \Delta)$. Finally, we denote the backdoor pattern estimated by the post-training defense as $\widehat{\Delta}$.

4.2. In-flight Backdoor Defense

The estimated backdoor pattern $\widehat{\Delta}$ obtained by an *effective* post-training defense will likely elicit the same targeted misclassification to the backdoor target class t as the true backdoor pattern Δ (used by the attacker and unknown to the defender) when they are embedded in source class images (as will be shown by our experiments). But these two patterns may have very different intensity values in the image space, as visualized by [14]. Therefore, it is unreliable to determine whether a test image classified to class t is embedded with Δ by directly looking for the estimated $\widehat{\Delta}$ in the image. Even if a test image embedded with Δ is successfully detected, directly removing an estimated $\widehat{\Delta}$ (*e.g.*, subtracting the estimated additive perturbation pattern [13]) from the image will not likely remove the true backdoor pattern Δ completely – the image may still be classified to the target class t.

However, in deep layers close to the DNN output (*e.g.*, the penultimate layer), the true backdoor pattern Δ and its empirical estimation $\widehat{\Delta}$ will likely activate the *same* set of neurons

[1] We only consider the cases where the post-training defender successfully detects the backdoor attack with correct inference of the target class.

when embedded in images from the same source class. These neurons are trained (on the poisoned training set) to activate for the backdoor mapping, and are likely *different* from the neurons activating for *typical* target class images. Thus, if a test image classified to the target class t is actually an image from some source class embedded with the backdoor pattern Δ, its deep layer activations are expected to be: a) similar to the activations for most images from the same source class embedded with the estimated pattern $\widehat{\Delta}$, and b) different from the activations for typical images from class t.

Based on the above intuition, our in-flight backdoor detection steps are as follows: First, for each non-target class $\forall c \in \mathcal{S}$, we embed the estimated backdoor pattern $\widehat{\Delta}$ in the clean images used for detection and get $\tilde{\mathcal{D}}_c = \{g(\boldsymbol{x}, \widehat{\Delta}) | \boldsymbol{x} \in \mathcal{D}_c\}$. Then we feed all images in $\bigcup_{c \in \mathcal{S}} \tilde{\mathcal{D}}_c$ and the clean target class images $\mathcal{D}_t$ to the DNN being attacked to get their deep layer features $\mathcal{Z}_c = \{f_L(\tilde{\boldsymbol{x}}) | \tilde{\boldsymbol{x}} \in \tilde{\mathcal{D}}_c\}$, $\forall c \in \mathcal{S}$ and $\mathcal{Z}_t = \{f_L(\boldsymbol{x}) | \boldsymbol{x} \in \mathcal{D}_t\}$, respectively. All the features in $\mathcal{Z} = \bigcup_{c \in \mathcal{C}} \mathcal{Z}_c$ are then standardized to have mean 0 and standard deviation 1. For each class $c \in \mathcal{C}$, we learn a density model (*e.g.*, Gaussian mixture model) with parameters $\boldsymbol{\theta}_c = \arg\max_{\boldsymbol{\theta}} \Pi_{\boldsymbol{z} \in \mathcal{Z}_c} P[\boldsymbol{z}|\boldsymbol{\theta}]$. *At test-time*, for any test image $\boldsymbol{w}$ with $f(\boldsymbol{w}) = t$, we measure its likelihood $\mathcal{L}_c = P[f_L(\boldsymbol{w})|\boldsymbol{\theta}_c]$ under the density model for each $c \in \mathcal{C}$. If $\arg\max_{c \in \mathcal{C}} \mathcal{L}_c \neq t$, $\boldsymbol{w}$ is deemed to contain the backdoor pattern. We then reject the prediction of the DNN and infer that it is originally from class $s = \arg\max_{c \in \mathcal{C}} \mathcal{L}_c$. Otherwise, $\boldsymbol{w}$ is deemed clean and its class prediction t is accepted.

Our method has the following advantages over existing ones. First, our detector is *unsupervised*. It performs hypothesis testing by comparing data likelihoods under different density models. Works like [14, 15, 18] are *supervised* detection methods and sensitive to the choice of hyper-parameters, *e.g.*, a detection threshold. Besides, our method only needs relatively few clean images (100 images per class in our experiments) for detection, and is computationally cheap, as it does not involve tuning a complicated DNN, for example. Note that the most time-consuming part of our defense – learning class-conditional density models – is done offline.

5. EXPERIMENTS

5.1. Experiment Setup

Dataset: We mainly use the benchmark CIFAR-10 dataset, which contains 60k 32×32 color images from 10 classes, with 5k images per class for training and 1k images per class for testing [19]. Experiments on other datasets including MNIST [20], PubFig [21], and F-MNIST [22] are in Sec. 5.3.
Data allocation: Following the assumptions in Sec. 3, we randomly split the test set of CIFAR-10 into $\mathcal{D}_{\text{Defense}}$ and $\mathcal{D}_{\text{Test}}$, where $\mathcal{D}_{\text{Defense}}$ consists of 100 images per class, and $\mathcal{D}_{\text{Test}}$ is used for performance evaluation.
Attack settings: We consider typical backdoor attacks described in Sec. 3. We arbitrarily choose class 9 as the target class for all attacks. We consider the following three backdoor patterns: 1) an additive perturbation that looks like a "chess board" (**CB**) used in [13]; 2) a single pixel set to 255 (**SP**) [23]; and 3) a 3×3 white box (**WB**) embedded in the bottom right of an image [2]. For each type of backdoor pattern, we create two attacks by: 1) embedding the backdoor pattern in 1000 training images randomly selected from class 0 (single class attack); 2) embedding the backdoor pattern in 100 training images randomly selected from each class, except for the target class (multi-class attack).
Training settings: For each attack, we train a DNN with the widely used ResNet-18 [24] architecture on the backdoor poisoned CIFAR-10 training set. Training is performed for 150 epochs with learning rate 0.1 (reduced by 0.5 per 50 epochs) and batch size 32.
Defense settings: For each attack, we first apply a post-training detector to the DNN being attacked to infer the target class t and reverse-engineer its corresponding backdoor pattern $\widehat{\Delta}$, using $\mathcal{D}_{\text{Defense}}$ reserved for detection. In particular, we apply the detector in [13] to attacks with backdoor pattern CB since it is an additive perturbation; and the detector in [14] to attacks with backdoor patterns SP or WB as they are small patches (not additive perturbations) embedded in an image. For our in-flight detector, we consider Gaussian mixture models and choose the penultimate layer features for (maximum likelihood) density model estimation.

5.2. Main Experimental Results on CIFAR-10

First, we show that the attacks we created are sufficiently effective for thoroughly evaluating detection performance. Such effectiveness is evaluated by: 1) the attack success rate (**ASR**) defined as the fraction of clean images from the source classes in $\mathcal{D}_{\text{Test}}$ that are misclassified to the target class when the ground truth (**GT**) backdoor pattern is embedded; and 2) the clean test accuracy (**ACC**) defined as the DNN's accuracy on $\mathcal{D}_{\text{Test}}$. As shown in Table 1, all attacks have high ASR and almost no degradation in ACC compared with the baseline ACC of a DNN trained with no backdoor. Second, we show the ASR for each attack with the backdoor pattern reverse-engineered (**RE**) by the post-training defense instead of the GT pattern in Table 1. The RE patterns induce similarly high misclassification rate to the target class, when embedded in clean source class images from $\mathcal{D}_{\text{Test}}$, as GT patterns.

We evaluate the effectiveness of our detector in comparison with three other in-flight detectors NC [14], B3D [15], and STRIP [18]. The metrics for performance evaluation include true positive rates (**TPR**, *i.e.*, the fraction of backdoor-trigger images correctly detected), false positive rates (**FPR**, *i.e.*, the fraction of clean images falsely detected) and source class inference accuracies (**SIA**, *i.e.*, the fraction of backdoor-trigger images with correct inference of the source class). As shown in Table 2, for all attacks, our detector performs nearly perfectly – almost all the backdoor-trigger images are cor-

Attack pattern	Single class attack		Multi-class attack	
	ACC	ASR	ACC	ASR
No attack	0.9387	NA	0.9387	NA
CB-GT	0.9360	0.9955	0.9381	0.9954
CB-RE	NA	1.0000	NA	0.9876
SP-GT	0.9354	0.9488	0.9337	0.9565
SP-RE	NA	0.9900	NA	0.9953
WB-GT	0.9324	0.9411	0.9340	0.9497
WB-RE	NA	0.9970	NA	0.9354

Table 1: ASR and ACC for attacks using GT patterns; and ASR for the RE patterns obtained by post-training defenses applied to these attacks. "NA" represents "not applicable".

Attack pattern	Single class attack			Multi-class attack		
	TPR	FPR	SIA	TPR	FPR	SIA
Likelihood-based in-flight backdoor defender						
CB	0.9922	0.0	0.8392	0.9997	0.0	0.6946
SP	0.9813	0.0	0.7728	0.9454	0.0	0.632
WB	0.9847	0.0	0.8607	0.9992	0.0	0.8945
NC						
CB	0.9855	0.0488	0.9444	0.9962	0.0533	0.8765
SP	0.8088	0.0544	0.8833	0.9043	0.0511	0.8963
WB	0.0	0.0522	0.8667	0.8644	0.0522	0.8086
B3D						
CB	0.0788	0.0511	NA	0.9872	0.9955	NA
SP	0.5333	0.1066	NA	0.1814	0.0522	NA
WB	0.0011	0.0500	NA	0.0535	0.0511	NA
STRIP						
CB	0.0822	0.0533	NA	0.0218	0.0555	NA
SP	0.1333	0.0588	NA	0.1555	0.0588	NA
WB	0.0088	0.0588	NA	0.0011	0.0633	NA

Table 2: TPR, FPR and SIA for our defense, compared with three other in-flight defenses, NC, B3D, and STRIP, against all the created attacks. "NA" signifies "not applicable".

rectly identified, and no clean target class images are falsely reported. Following [14, 15, 18], we build in-flight detectors for NC, B3D, and STRIP. As they do not announce the thresholds for detection, we test them with various thresholds and exhibit the *best* TPR at an FPR of around 5%[2]. For STRIP, we set the weight of the incoming input as 0.5 in image blending. NC does not correctly detect any backdoor-trigger images in a single class attack using the pattern WB. B3D does not perform well under all the attacks, as discussed in Sec.2. STRIP does not perform well either (for ResNet18 and CIFAR-10) – STRIP is more effective for the DNN architectures, datasets, and attack configurations that were used in [18].

Since the NC paper does not mention the learning rate used for fine-tuning the DNN, in our experiments, we fine-tune the poisoned DNN using various learning rates. From Table 3, the severe fluctuations in SIAs of models poisoned by the multi-class attack using pattern WB shows the sensitivity of NC to the learning rate. We thus apply NC with all the learning rates, reporting the *best* SIA in Table 2. Our defender performs relatively well in inferring source classes for the detected backdoor trigger images, though it is not as good as the *best* results of NC. However note that our method requires neither a clean set as large as the one used by NC for fine-tuning, nor careful choices of hyper-parameters.

[2]A 5% FPR is not achievable by B3D for some attacks (even as the detection threshold is varied over a wide range).

LR	10	1	10^{-1}	10^{-2}	10^{-3}	10^{-4}
SIA	0.1114	0.4291	0.6033	0.8086	0.7103	0.6348

Table 3: SIA of NC fluctuates with the learning rate (LR) used for DNN fine-tuning.

	PubFig		MNIST		F-MNIST	
	SQ	WM	CB	WB	CB	WB
TPR	0.9856	1.0	1.0	1.0	1.0	0.9951
FPR	0.1428	0.1428	0.0033	0.0022	0.0023	0.0127
SIA	0.7194	0.5507	0.9413	0.9764	0.6948	0.8039

Table 4: TPR, FPR and SIA for our in-flight backdoor detector on datasets PubFig, MNIST and F-MNIST.

5.3. Experimental Results on Other Datasets

We also evaluated our detector on datasets including PubFig, MNIST and F-MNIST. For PubFig, we randomly choose 20 classes, each with 80 training samples and 20 test samples. We arbitrarily select class 19 as the target class and embed a backdoor trigger in 2 training samples from each class except for the target class. The backdoor patterns considered for PubFig are trojan square (**SQ**) and trojan watermark (**WM**) [4], and the poisoned training set is then used for training a DNN with the VGG-16 [25] architecture. We reserve 5 test images per class for detection. For each of MNIST and F-MNIST, we consider two attacks with backdoor patterns CB and WB, respectively, and the same attack settings for the multi-class attacks described in Sec.5.1. For each attack, a DNN with LeNet-5 [26] architecture is trained on the poisoned training set. As shown in Table 4, our defender achieves similarly good performance on these datasets as for the CIFAR-10 dataset.

6. CONCLUSION

We proposed an *unsupervised* detector of *test-time* images that contain a backdoor trigger. Our defense does not require setting a detection threshold, nor access to the DNN's training set/DNN training. The backdoor trigger detection and source class inference exploit the backdoor-trigger image's atypicality with respect to the clean target class images and its typicality with respect to images from its true (source) class of origin. We show the effectiveness and wide application of our detector experimentally for a wide variety of DNN architectures, datasets, and backdoor attack configurations.

7. REFERENCES

[1] David J. Miller, Zhen Xiang, and George Kesidis, "Adversarial learning targeting deep neural network classification: A comprehensive review of defenses against attacks," *Proc. IEEE*, 2020.

[2] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg, "Badnets: Evaluating backdooring attacks on deep neural networks," *IEEE Access*, 2019.

[3] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song, "Targeted backdoor attacks on deep learning systems using data poisoning," *arXiv:1712.05526*, 2017.

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

[5] Shaofeng Li, Benjamin Zi Hao Zhao, Jiahao Yu, Minhui Xue, Dali Kaafar, and Haojin Zhu, "Invisible backdoor attacks against deep neural networks," *arXiv:1909.02742*, 2019.

[6] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash, "Hidden trigger backdoor attacks," in *AAAI*, 2020.

[7] Jiazhu Dai, Chuanshuai Chen, and Yufeng Li, "A backdoor attack against lstm-based text classification systems," *IEEE Access*, 2019.

[8] Zhen Xiang, David J. Miller, Siheng Chen, Xi Li, and George Kesidis, "A backdoor attack against 3d point cloud classifiers," *ICCV*, 2021.

[9] Xi Li, George Kesidis, David J. Miller, and Vladimir Lucic, "Backdoor attack and defense for deep regression," *arXiv:2109.02381*, 2021.

[10] Yiming Li, Baoyuan Wu, Yong Jiang, Zhifeng Li, and Shu-Tao Xia, "Backdoor learning: A survey," *arXiv:2007.08745*, 2020.

[11] Brandon Tran, Jerry Li, and Aleksander Madry, "Spectral signatures in backdoor attacks," in *NeurIPS*, 2018.

[12] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian M. Molloy, and Biplav Srivastava, "Detecting backdoor attacks on deep neural networks by activation clustering," in *AAAI*, 2019.

[13] Zhen Xiang, David J. Miller, and George Kesidis, "Detection of backdoors in trained classifiers without access to the training set," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[14] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y. Zhao, "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *2019 IEEE Symposium on Security and Privacy*, 2019.

[15] Yinpeng Dong, Xiao Yang, Zhijie Deng, Tianyu Pang, Zihao Xiao, Hang Su, and Jun Zhu, "Black-box detection of backdoor attacks with limited information and data," *ICCV*, 2021.

[16] Wenbo Guo, Lun Wang, Xinyu Xing, Min Du, and Dawn Song, "TABOR: A highly accurate approach to inspecting and restoring trojan backdoors in AI systems," *arXiv:1908.01763*, 2019.

[17] Zhen Xiang, David J. Miller, and George Kesidis, "Revealing backdoors, post-training, in DNN classifiers via novel inference on optimized perturbations inducing group misclassification," in *ICASSP*, 2020.

[18] Yansong Gao, Change Xu, Derui Wang, Shiping Chen, Damith Chinthana Ranasinghe, and Surya Nepal, "STRIP: a defence against trojan attacks on deep neural networks," in *ACSAC*, 2019.

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

[20] Li Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, 2012.

[21] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar, "Attribute and Simile Classifiers for Face Verification," in *ICCV*, 2009.

[22] Han Xiao, Kashif Rasul, and Roland Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," 2017.

[23] Zhen Xiang, David J. Miller, and George Kesidis, "A benchmark study of backdoor data poisoning defenses for deep neural network classifiers and a novel defense," in *MLSP*, 2019.

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

[25] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.

[26] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, 1998.