Detecting and Mitigating Backdoor Attacks with Dynamic and Invisible Triggers

Zhibin Zheng[1], Zhongyun Hua[1,2(✉)], and Leo Yu Zhang[3]

[1] School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, Shenzhen 518055, China
20s151088@stu.hit.edu.cn

[2] Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies, Harbin Institute of Technology, Shenzhen, Shenzhen 518055, China
huazhongyun@hit.edu.cn

[3] School of Information Technology, Deakin University, Victoria 3216, Australia
leo.zhang@deakin.edu.au

Abstract. When a deep learning-based model is attacked by backdoor attacks, it behaves normally for clean inputs, whereas outputs unexpected results for inputs with specific triggers. This causes serious threats to deep learning-based applications. Many backdoor detection methods have been proposed to address these threats. However, these defenses can only work on the backdoored models attacked by static trigger(s). Recently, some backdoor attacks with dynamic and invisible triggers have been developed, and existing detection methods cannot defend against these attacks. To address this new threat, in this paper, we propose a new defense mechanism that can detect and mitigate backdoor attacks with dynamic and invisible triggers. We reverse engineer generators that transform clean images into backdoor images for each label. The generated images by the generator can help to detect the existence of a backdoor and further remove it. To the best of our knowledge, our work is the first work to defend against backdoor attacks with dynamic and invisible triggers. Experiments on multiple datasets show that the proposed method can effectively detect and mitigate the backdoor with dynamic and invisible triggers in deep learning-based models.

Keywords: AI security · Backdoor attack · Backdoor detection

1 Introduction

Deep neural networks (DNNs) have been widely used in many applications such as image classification [7], object detection [15], and language processing [1]. The performance of a DNN model relies on the high complexity of the model and the large amount of training data. It is thus acknowledged that common DNN end-users do not possess the capability to train a well-performed model by themselves. Therefore, users often outsource the training process of DNNs to a third

M. Tanveer et al. (Eds.): ICONIP 2022, LNCS 13625, pp. 216–227, 2023.
https://doi.org/10.1007/978-3-031-30111-7_19

party or directly use a well-trained model from a model-sharing platform [6]. Either option introduces the attack surface of DNN backdoor.

Backdoor attacks aim to embed a backdoor into a DNN model [6]. The backdoor does not degrade the model's performance on clean inputs, but outputs the wrong results expected by the attackers when a specific trigger appears in the inputs [6]. Backdoor attacks are stealthy because the backdoor triggers are kept secret by the attacker, and model users with only clean inputs cannot activate the backdoor. In this regard, it is hard for model users to realize the existence of the backdoor [14]. Therefore, backdoor attacks cause a serious threat to the application of deep learning.

Various backdoor defense methods have been proposed to defend against backdoor attacks [2,4,20]. Most of these methods are empirical methods that are developed based on the characteristics of existing attack methods. For example, the trigger-synthesis-based defense methods [3,20,23] first reverse engineer a trigger for each label of the models and then run anomaly detection to determine the attacked target label. The saliency-map-based defense method [2] utilizes model visualization techniques to locate the potential trigger regions by finding common saliency regions in different input images. Besides, the input-filtering-based defense method [4] first adds different perturbations to the input image, and then calculates the randomness of the model's prediction to detect backdoor. However, all these defense methods are based on the assumption that a static trigger is involved in different images to launch an attack. They are ineffective against the newly proposed backdoor attacks [13,14] with dynamic and invisible triggers, since these attacks add different triggers for different clean images.

In this paper, we propose a backdoor defense method to detect and mitigate backdoor attacks with dynamic and invisible triggers. We reverse engineer the trigger pattern from the backdoored model. Different from existing trigger-synthesis-based methods that can reverse engineer only a static trigger, we reverse engineer a generator to transform clean images into backdoor images, which allows us to add different triggers for different images and capture more complex trigger patterns. The contributions of this paper are summarized as follows:

1. We design a new backdoor defense method that can defend against the backdoor attacks with dynamic and invisible triggers. We design a reverse engineering process of backdoor image generators using the characteristics of backdoor attacks.
2. The reversed generators are used for anomaly detection to detect the backdoor target. When a backdoor target is found, we use the generated backdoor images and a small subset of clean data to unlearn the backdoor.
3. We evaluate our proposed method on MNIST, CIFAR10, and GTSRB datasets with the most recent backdoor attack WaNet [14]. Experimental results show that the proposed method can effectively detect the backdoor and greatly reduce the attack success rate with only a small drop in clean accuracy.

2 Related Works

2.1 Backdoor Attacks

Backdoor attacks are aimed at embedding a backdoor associated with a trigger pattern into a DNN model. The model with a backdoor should preserve performance on clean inputs. However, when the inputs are patched with the trigger pattern, the model will output the results expected by the attackers.

Trigger pattern design is the core of backdoor attacks, where better trigger patterns can make backdoor attacks stealthy and effective. According to whether different triggers are applied to different images, backdoor attacks can be divided into static and dynamic backdoor attacks. For the case of static attacks, the injection of a static trigger into a clean image x can be formulated as

$$x' = (1 - m) \cdot x + m \cdot \Delta, \tag{1}$$

where Δ is the trigger pattern specified by the attacker, m is the mask to decide the location for the trigger to stamp on clean images. Gu et al. [6] proposed the first backdoor attack that use a small patch as the trigger pattern. Later works use optimized triggers [10], smooth triggers [21] for better stealthy.

For the case of dynamic backdoor attacks, Nguyen et al. [13] firstly suggested input-aware triggers. They optimize a generator to generate different triggers for different images with a diversity loss. When a trigger generated for one image is added to another image, the resulting image cannot activate the backdoor. WaNet [14] is another dynamic backdoor attack, which creates backdoor images using a small and smooth warping field. And the backdoor images contain triggers that vary from image to image. As the distortion caused by warping is slight, it is difficult to distinguish the backdoor images from the original images. Thus, the triggers in the WaNet attack are dynamic and invisible. Backdoor attacks with such triggers are more powerful than previous attacks as they not only break assumptions of various defense methods but also evade manual inspection.

2.2 Backdoor Defense Methods

Existing backdoor defense methods can be classified as directly reducing backdoor attack success rates (i.e., mitigation) or detecting the backdoor for a victim model (i.e., detection). For the case of mitigation, Liu et al. [12] proposed to prune neurons not useful for clean images and fine-tune the pruned model on clean images. Li et al. [11] used attention distillation to guide the fine-tuning process and showed a better result compared to pruning and fine-pruning. The major limitation of these methods is that they are blind to the existence of backdoor: if they are used for clean models, they also cause a decrease in model's accuracy.

For detection, trigger synthesis is the most popular method for its capability to not only detect the existence of the backdoor but also remove/unlearn the backdoor. Wang et al. [20] proposed the first trigger synthesis based defense

method Neural Cleanse (NC), which reverses triggers for each label and uses the L1 norm of trigger masks to detect backdoor target labels. Once a backdoor is detected, they mitigate the backdoor by filtering inputs or patching the backdoored model. Zhu et al. [23] proposed GangSweep which uses GAN [5] to better reconstruct triggers. Dong et al. [3] extended it to the black-box setting where the weights of the model is not accessible.

Our work also falls into the same category of backdoor detection and mitigation. But existing methods are based on the same assumption that a universal trigger is used, so they are not able to defend against attacks with dynamic triggers.

3 Proposed Method

3.1 Threat Model and Defense Goal

We consider that a user obtains a pre-trained model from an untrusted third party. The model could be backdoored. In particular, a stronger backdoor attack with dynamic and invisible triggers [14] instead of a static trigger [6,10,21] is possible.

Following the literature studies [11,20], the defender has full access to the model and has a subset of clean data. The set of clean data can be the well-labeled data used to test the performance of the model. The defender aims to detect whether the model is backdoored and find out what the attacker's target label is. Upon detection, the defender will also try to mitigate the backdoor.

3.2 Intuition and Overview

Backdoor attacks with dynamic and invisible triggers use different triggers for clean images. The backdoored model has learned to classify images with such triggers toward the attacker specified target label. We model the process of transforming clean images into backdoor images as a generator, taking clean images as input and outputting backdoor images, and try to reverse engineer such a generator from the backdoored model. The proposed method consists of three steps:

1. Reverse engineering backdoor image generator. Given a classifier model, we reverse engineer a backdoor image generator for every label by optimizing the generator to output images that not only close to the input images but also change the prediction of the model to a specific label regardless of the input images' ground truth labels.
2. Backdoor detection. We use an outlier detection algorithm to judge if there is a generator that can generate backdoor images with small modifications and achieve a high probability of misleading the model to predict them into a specific label.
3. Backdoor mitigation. We leverage the reversed generator to generate backdoor images and combine them with a set of clean images to fine-tune the backdoored model.

3.3 Reverse Engineering Backdoor Image Generator

We define the transformation of a clean image x to a backdoor image x' via a generator model G as $x' = G(x)$. The generator G is required to perform an image-to-image transformation, thus we choose U-Net [16] as the generator model. Compared with the basic encoder-decoder model, U-Net concatenates high-level features and low-level features to propagate context information and is easier to train on a small dataset, which is suitable for our task as we assume the defender only has a small set of clean data.

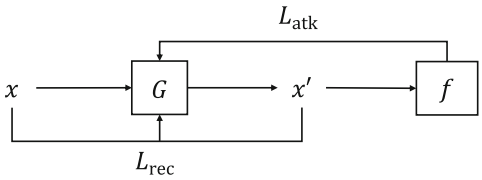

Fig. 1. The framework to reverse engineer the backdoor image generator.

As shown in Fig. 1, the reverse engineering of G from a classifier model f is the optimization for two objectives: effectiveness and stealthiness. The effectiveness goal requires the reversed generator to transform images from other classes into images that will be classified as a given class y_t with a high probability. The stealthiness goal requires that the transformed images be as similar to the original images as possible.

For the effectiveness of G to mimic trigger pattern, we use the cross-entropy loss L_{atk} to encourage the generated images to be classified as the target label y_t by model f, i.e.,

$$L_{\text{atk}} = \lambda_1 \cdot \text{CrossEntropy}(y_t, f(G(x)). \tag{2}$$

For the stealthiness of the G's output, we use L_{rec} to encourage G to reconstruct images from inputs. It is a combination of mean square error (MSE) and learned perceptual image patch similarity (LPIPS) [22] to measure reconstruction loss, which can be formulated as

$$L_{\text{rec}} = \lambda_2 \cdot \text{MSE}(x, G(x)) + \lambda_3 \cdot \text{LPIPS}(x, G(x)). \tag{3}$$

Here, the MSE loss measures the pixel value difference between the generated image and the input image, and the LPIPS loss measures the feature difference between the generated image and the input image.

The final loss function to train G is the combination of L_{atk} and L_{rec}:

$$L = L_{\text{atk}} + L_{\text{rec}}. \tag{4}$$

For each label of model f, we optimize the parameters of a generator using the above loss functions. We use a binary search to find the hyper-parameters λ_1, λ_2, λ_3 to achieve a high attack success rate.

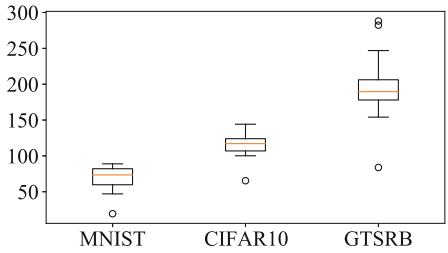

Fig. 2. Norm value distribution of clean and target labels on three datasets.

3.4 Backdoor Detection

With all the reversed generators associated with each label available, we then use them to detect the backdoor. Intuitively, for the attacker's target label, if the generator can capture the trigger pattern, it would be able to generate backdoor images with fewer modifications. To achieve a high attack success rate on clean labels, the generator needs to generate backdoor images with larger modifications. As there is not a universal trigger, the modification for label y_t is defined as the mean L1 norm of the residuals between the generated images and the clean input images:

$$m_{y_t} = \frac{1}{n} \sum_{i=1}^{n} |G_{y_t}(x_i) - x_i|, \text{ for } x_i \in X_{/y_t}, \tag{5}$$

where G_{y_t} is the generator reversed for label y_t, X is the set of clean images the defender has access to, $X_{/y_t}$ is the set of clean images that are not belonging to label y_t.

To validate our intuition, we calculate the average L1 norm value on test dataset. Figure 2 shows the distributions of residuals norm of different labels on 3 datasets. For each dataset, the box-plot shows the norm value distribution of clean labels, while the dot under the box-plot is the norm value of the target label, which is much smaller than that of clean labels. Therefore, we can leverage this characteristic to detect the target label. By taking MNIST as an example, Fig. 3 visualizes this observation. The bright and dark spots on the residual images are indications of large image modifications. The residual image for the target label 0 is grayer and more smooth than those of other labels, which means that there are fewer modifications.

We use the Median Absolute Deviation (MAD)-based outlier detection method adopted in NC [20] over each label y_t with its residual m_{y_t} obtained from Eq. (5). The anomaly index of y_t is calculated as follows:

$$AI_{y_t} = \frac{|m_{y_t} - \tilde{m}|}{c \cdot \text{median}_{i \in \{1, \cdots, K\}}(|m_{y_i} - \tilde{m}|)}, \tag{6}$$

where $\tilde{m}$ is the median of all residuals, K is the number of classes of the dataset, and c is a constant to normalize the anomaly index. For the purpose of detecting

backdoor, we only need to focus on small residuals, so only labels with a residual smaller than the median residual value are considered. Similar to NC, we assume that the data satisfy normal distribution, c is thus set to 1.4826 and any label with an anomaly index larger than 2 will be considered as an attack target label with a 95% confidence level. Moreover, $\max_{t\in\{1,\cdots,K\}}(AI_{y_t})$ is called the model anomaly index.

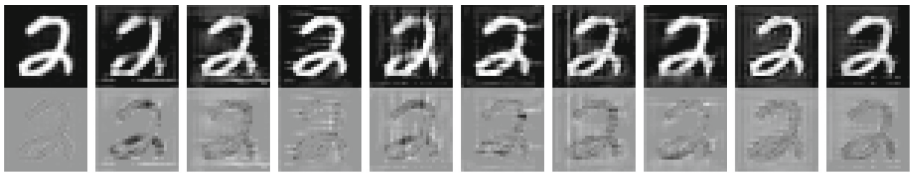

Fig. 3. Comparison of images generated by generators reversed for different labels. The first row is the generated images and the second row is the residuals between the generated image with the original images. The residuals are normalized from $[-1, 1]$ to $[0, 1]$. The label '0' is the target label.

3.5 Backdoor Mitigation

Once a backdoor is detected, we use the generator reversed for the detected target label to remove the backdoor. We use a subset of clean data and transform these clean images into backdoor images, label them with their original labels, and combine them with the subset of clean data to form a new dataset. We use the new dataset to fine-tune the backdoored model for a few epochs (empirically, 10 is enough).

4 Experiments

In this section, we describe our experiments on multiple classification tasks attacked by state-of-the-art backdoor attack method WaNet [14].

4.1 Experiment Setup

To evaluate the effectiveness of our defense method, we use three classification tasks: MNIST [9], CIFAR10 [8], GTSRB [19]. We use the default DNN models used in WaNet, namely, a simple classifier consisting of 3 ConvBlocks + 3 fcs for MNIST, and pre-activation Resnet-18 [7] for CIFAR10 and GTSRB. We also use VGG16 [18] and MobileNetV2 [17] for CIFAR10 and GTSRB. For each dataset and model architecture, we also train a clean model using the entire clean training dataset.

The hyper-parameters λ_1, λ_2, λ_3 are set to 0.1, 1, 0.1 respectively. We use an Adam optimizer with a learning rate of 0.001 to train the generator for each

label. Instead of training the generators with random initial weights, we first optimize the generator using L_{rec} for 20 epochs to get a pretrained generator. The pretained weights are then loaded as initial weights to optimize generators for each label using the final loss L. For backdoor detection, we use 1% of the clean training data for each dataset. For backdoor mitigation, we use 5% of the training data.

4.2 Detection Performance

For each clean and backdoored model, we repeat the experiments 10 times with different random seeds, and the averaged model anomaly index is reported. Table 1 shows the tested anomaly indices. For backdoored MobileNetV2 models, their model anomaly indices are around 3. For other backdoored models, the model anomaly indices are larger than 4. For clean models, the anomaly indices are smaller than 2. This shows that our method can accurately detect the backdoored models.

Table 1. The backdoor detection results.

Dataset	Model	Method	Anomaly Index		Detailed Detection Results			
			Clean	Backdoor	Case 1	Case 2	Case 3	Case 4
MNIST	3ConvBlocks+3fcs	NC	1.10	0.75	0/10	0/10	0/10	10/10
		Ours	1.07	5.41	9/10	1/10	0/10	0/10
CIFAR10	PreActRes18	NC	2.52	2.32	0/10	0/10	6/10	4/10
		Ours	1.66	5.06	9/10	1/10	0/10	0/10
	VGG16	NC	1.91	4.00	0/10	4/10	3/10	3/10
		Ours	1.94	4.01	8/10	1/10	0/10	1/10
	MobileNetV2	NC	1.30	1.59	0/10	0/10	1/10	9/10
		Ours	1.30	2.98	9/10	1/10	0/10	0/10
GTSRB	PreActRes18	NC	2.65	1.52	0/10	3/10	1/10	6/10
		Ours	1.99	4.92	9/10	1/10	0/10	0/10
	VGG16	NC	1.94	1.94	0/10	0/10	3/10	7/10
		Ours	1.89	6.36	6/10	4/10	0/10	0/10
	MobileNetV2	NC	2.42	3.98	2/10	8/10	0/10	0/10
		Ours	1.95	3.24	5/10	4/10	0/10	1/10

To further assess the proposed algorithm's capability in detecting the target labels, we compare it with NC [20] by considering the following four cases:

- Case 1: The backdoor is detected and only the target label is detected.
- Case 2: The backdoor is detected and the target label is detected, but at least one clean label is identified as target label.
- Case 3: The backdoor is detected but the target label is not detected.
- Case 4: The backdoor is not detected.

The results are listed in Table 1. From this table, for the backdoored model on MNIST dataset, the target label is detected accurately 9 times with one exception: a clean label is mistakenly identified as the target label. There are similar

results for all backdoored models on CIFAR10 dataset. For GTSRB dataset, the detection algorithm can detect the backdoor but has more false positive alarms for target labels on VGG16 and MobileNetV2 models than other models. This may be because there are more classes on GTSRB and the attack has side effects on other classes. While NC fails to detect the target label except for VGG16 model on CIFAR10 and MobileNetV2 model on GTSRB. The detection accuracy on these two models is still much lower than our method.

Visual Similarity. Figure 4 compares the original backdoor images and backdoor images generated by reversed generators. The generated backdoor images are close to the original backdoor images. The residuals with the clean image of the original backdoor image and the generated backdoor image both show the texture of the original images. It indicates that the optimization of the generator is capable of getting close to the trigger pattern.

Fig. 4. Comparison of backdoor images, generated backdoor images, and their residuals with the clean images of three datasets.

4.3 Backdoor Mitigation Performance

We use a Stochastic Gradient Descent (SGD) optimizer with a learning rate of 0.001 and a momentum of 0.9 to fine-tune the backdoored model for 10 epochs. We compare the mitigation results with basic fine-tuning and NAD [11].

The results of backdoor mitigation with different amount of clean data are shown in Table 2, where ACC stands for accuracy on clean inputs and ASR stands for attack success rate. The results show that on the MNIST and GTSRB datasets, even with only 1% clean training data, our method reduces the attack success rate to less than 1%, while the drop of the accuracy on clean data is less than 1%. On the CIFAR10 dataset, our method reduces the attack success rate to 2.58% using 5% clean data, while the accuracy on clean data drops by 1.18%. Without generated backdoor images of the reversed generator, fine-tuning has little effect on backdoor mitigation.

Figure 5 compares the mitigation performance of different methods with different learning rates when 5% of the clean training data is available. It shows that fine-tuning can be more effective when the learning rate is increased. However, fine-tuning the model with a larger learning rate has a risk of decreasing its

accuracy on clean data. Similar to fine-tuning, NAD is sensitive to learning rate settings. The accuracy on clean data would drop severely when the learning rate is larger than a certain value, which is unstable. Compared to fine-tuning and NAD, our method shows a more stable mitigation effect with a small learning rate. It may be because tuning model with only clean data mitigates the backdoor through forgetting the trigger pattern, while we explicitly force the model to unlearn the trigger pattern.

Table 2. Mitigation performance with different percentages of clean training data.

Dataset	# of data	w/o reversed		w/ reversed	
		ACC	ASR	ACC	ASR
MNIST	1%	99.37	99.77	99.09	0.07
	3%	99.34	99.55	99.16	0.08
	5%	99.38	98.49	99.26	0.09
CIFAR10	1%	93.74	97.66	93.88	19.20
	3%	94.08	83.28	93.93	9.45
	5%	92.50	82.04	92.96	2.58
GTSRB	1%	98.76	93.60	98.89	0.79
	3%	99.03	94.63	99.00	0.43
	5%	99.03	94.65	99.03	0.19

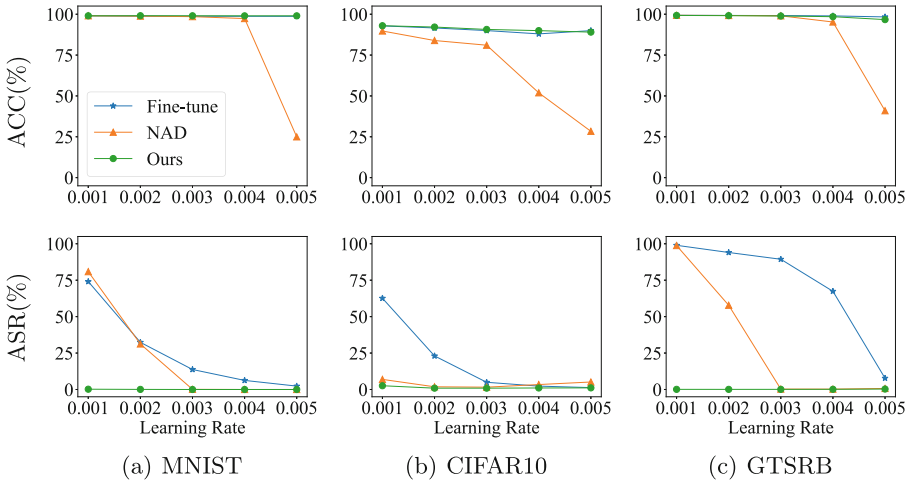

Fig. 5. Mitigation performance with different learning rates when 5% of clean training data is available.

4.4 Ablation Study

In Eq. (3), we use a combination of MSE and LPIPS. LPIPS uses a trained deep neural network to extract features to calculate features' difference and pays more attention to the image's overall changes. We conduct an ablation study without using the LPIPS loss. As shown in Fig. 6, when only the MSE loss is used, there are chances that the training falls into a local optimum. Although the generated image is basically the same as the original image in texture, there is a large change in color. This is because in an image with three channels, scattering the modifications to images across three channels may have equal MSE loss as centralizing the modifications to one channel. The LPIPS loss can help avoid this issue and better reconstruct the original image.

Fig. 6. Comparison results with and without LPIPS loss. Images from left to right are clean image, backdoor image, generated backdoor image with LPIPS loss, and generated backdoor image without LPIPS loss.

5 Conclusion

In this paper, we propose to defend against backdoor attacks with dynamic and invisible triggers by reverse-engineering the backdoor image generator. By carefully designing the optimization objectives, we can effectively reverse engineer a generator that can capture the embedded trigger pattern. The target label can be thus detected and the reversed generators can help eliminate the backdoor. We conduct experiments on multiple datasets under the WaNet attack. The experimental results show that our method can detect backdoors more effectively than the existing trigger-synthesis-based methods. Compared to fine-tuning using only clean data, fine-tuning aided by the reversed generator has a more stable backdoor mitigation performance, thus avoiding the trade-off between preserving accuracy on clean data and reducing attack success rate.

Acknowledgements. This work was supported in part by the National Natural Science Foundation of China under Grants 62071142, by the Guangdong Basic and Applied Basic Research Foundation under Grants 2021A1515011406, by the Shenzhen College Stability Support Plan under Grant GXWD20201230155427003-20200824210638001, by the Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies under Grant 2022B1212010005.

References

1. Brown, T., et al.: Language models are few-shot learners. Adv. Neural Inf. Process. Syst. **33**, 1877–1901 (2020)
2. Chou, E., Tramèr, F., Pellegrino, G.: Sentinet: detecting localized universal attacks against deep learning systems. In: IEEE S&P Workshops, pp. 48–54 (2020)
3. Dong, Y., et al.: Black-box detection of backdoor attacks with limited information and data. In: ICCV, pp. 16482–16491 (2021)
4. Gao, Y., Xu, C., Wang, D., Chen, S., Ranasinghe, D.C., Nepal, S.: Strip: a defence against trojan attacks on deep neural networks. In: ACSAC, pp. 113–125 (2019)
5. Goodfellow, I.J., et al.: Generative adversarial nets. In: NeurIPS, pp. 1–9 (2014)
6. Gu, T., Liu, K., Dolan-Gavitt, B., Garg, S.: Badnets: evaluating backdooring attacks on deep neural networks. IEEE Access **7**, 47230–47244 (2019)
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR, pp. 770–778 (2016)
8. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images. University of Toronto, Technical Report (2009)
9. LeCun, Y., et al.: Learning algorithms for classification: a comparison on handwritten digit recognition. Neural Netw. Stat. Mech. Perspect. **261**(276), 2 (1995)
10. Li, S., Xue, M., Zhao, B.Z.H., Zhu, H., Zhang, X.: Invisible backdoor attacks on deep neural networks via steganography and regularization. IEEE Trans. Dependable Secure Comput. **18**(5), 2088–2105 (2021)
11. Li, Y., Lyu, X., Koren, N., Lyu, L., Li, B., Ma, X.: Neural attention distillation: erasing backdoor triggers from deep neural networks. In: ICLR, pp. 1–12 (2021)
12. Liu, K., Dolan-Gavitt, B., Garg, S.: Fine-pruning: defending against backdooring attacks on deep neural networks. In: Research in Attacks, Intrusions, and Defenses, pp. 273–294 (2018)
13. Nguyen, T.A., Tran, A.: Input-aware dynamic backdoor attack. In: NeurIPS, pp. 3454–3464 (2020)
14. Nguyen, T.A., Tran, A.T.: Wanet - imperceptible warping-based backdoor attack. In: ICLR, pp. 1–11 (2021)
15. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell. **39**(6), 1137–1149 (2016)
16. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 234–241 (2015)
17. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv 2: inverted residuals and linear bottlenecks. In: CVPR, pp. 4510–4520 (2018)
18. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR, pp. 1–14 (2015)
19. Stallkamp, J., Schlipsing, M., Salmen, J., Igel, C.: Man vs. computer: benchmarking machine learning algorithms for traffic sign recognition. Neural Netw. **32**, 323–332 (2012)
20. Wang, B., et al.: Neural cleanse: identifying and mitigating backdoor attacks in neural networks. In: S&P, pp. 707–723 (2019)
21. Zeng, Y., Park, W., Mao, Z.M., Jia, R.: Rethinking the backdoor attacks' triggers: a frequency perspective. In: ICCV, pp. 16473–16481 (2021)
22. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR, pp. 586–595 (2018)
23. Zhu, L., Ning, R., Wang, C., Xin, C., Wu, H.: Gangsweep: sweep out neural backdoors by gan. In: ACM MM, pp. 3173–3181 (2020)