Hindawi
Security and Communication Networks
Volume 2022, Article ID 4593002, 11 pages
https://doi.org/10.1155/2022/4593002

Research Article

Natural Backdoor Attacks on Deep Neural Networks via Raindrops

Feng Zhao,[1] **Li Zhou**,[1] **Qi Zhong**,[2] **Rushi Lan**,[1] **and Leo Yu Zhang**[2]

[1]*Guangxi Key Laboratory of Image and Graphic Intelligent Processing, Guilin University of Electronic Technology, Guilin 541004, China*
[2]*School of Information Technology, Deakin University, Geelong, VIC 3216, Australia*

Correspondence should be addressed to Qi Zhong; zhongq.lk@outlook.com and Rushi Lan; rslan2016@163.com

Received 12 December 2021; Accepted 1 March 2022; Published 26 March 2022

Academic Editor: Weizhi Meng

Recently, deep learning has made significant inroads into the Internet of Things due to its great potential for processing big data. Backdoor attacks, which try to influence model prediction on specific inputs, have become a serious threat to deep neural network models. However, because the poisoned data used to plant a backdoor into the victim model typically follows a fixed specific pattern, most existing backdoor attacks can be readily prevented by common defense. In this paper, we leverage natural behavior and present a stealthy backdoor attack for image classification tasks: the raindrop backdoor attack (RDBA). We use raindrops as the backdoor trigger, and they are naturally merged with clean instances to synthesize poisoned data that are close to their natural counterparts in the rain. The raindrops dispersed over images are more diversified than the triggers in the literature, which are fixed, confined, and unpleasant patterns to the host content, making the triggers more stealthy. Extensive experiments on ImageNet and GTSRB datasets demonstrate the fidelity, effectiveness, stealthiness, and sustainability of RDBA in attacking models with current popular defense mechanisms.

1. Introduction

Internet of Things (IoT) devices have infiltrated many industries and are now part of our daily lives. As a consequence, a vast amount of data will be created. Deep learning technology has been shown to be extremely effective in processing enormous volumes of high-dimensional data, and it is now widely employed in a variety of IoT applications, such as intelligent driving [1], computer vision [2,3], natural language processing [4–6], and *etc*. In another aspect, neural networks used in IoT are evolving into deeper and wider architectures to perform well in various tasks. This indicates that there are massive parameters to learn and a lot of computing resources to consume. Such requirements boost the development of machine learning-related industries, including machine learning as a service (MLaaS). In essence, various giant companies, such as Google and Amazon, have launched their own MLaaS platforms to facilitate users to outsource their model training projects. However, security vulnerabilities in deep neural networks may arise in any stage of the supply chain, including but not limited to unprotected open channels, unreliable data sources, and unreliable training processes.

Studies have shown that deep neural networks are prone to attacks from different stages, including inference-stage attacks [7–9] and training stage attacks. Inference-stage attacks, best known for adversarial attacks [10–13], generally aim to mislead the deep neural network to produce high-confidence error prediction results for the test data during inference. They are usually achieved by adding subtle input-specific perturbations to the test data before querying the target model.

Training-stage attacks usually refer to backdoor attacks [14], which intend to manipulate the model predictions for those attacker-specified instances by poisoning some normal training data. Specifically, attackers inject some patterns, dubbed triggers, into clean samples. These modified data are also referred to as poisoned data or backdoor samples, and they are assigned predefined target labels. By involving backdoor samples and normal data for model training, the trained model is

thus embedded with backdoors. At the inference stage, the backdoored model performs normally on benign inputs but predicts the target labels for instances containing triggers. This type of attack is stealthy since the victim model has state-of-the-art performance on clean inputs, which is indistinguishable from its corresponding clean model, while its backdoor behavior can only be activated by attacker-specified (unknown) inputs. In a nutshell, the potential malicious behavior could result in dire consequences in some security-critical areas, such as autonomous driving [15], face recognition [16], and speaker recognition [17–19], which will also cause serious obstacles to the DNN deployment and development.

In the literature, the most popular and effective backdoor triggers are simple, fixed, or unpleasant patterns, i.e., different clean data are patched with the same trigger in a fixed position without considering the host data content. In addition, the poisoned instances are generated by simply stamping triggers into benign host samples. For example, the trigger of BadNets [14], as shown in Figure 1, is a black and white pixel block in the bottom right corner of an image. Such poisoned data will inevitably have abnormal distributions and appear unnatural, raising the suspicions of model developers/users. They can be easily filtered out before the model training stage or rejected before the model inference. On the other hand, researchers have made tremendous efforts to improve the robustness of DNN models, and various backdoor countermeasures have been proposed to remove or suppress the backdoor behaviors of DNN models. It has been demonstrated that most of the existing backdoor methods can be successfully alleviated by some current popular defenses, e.g., fine-tuning [20], fine-pruning [21], and Grad-CAM based defenses [22,23].

Based on this understanding, in this work, we introduce a novel, simple, but effective backdoor attack method using raindrops, dubbed raindrops backdoor attack (RDBA). Specifically, we perform two different blur operations on uniformly distributed noise to simulate water droplets in real scenes with different sizes and directions and make them have motion blur. We then merge the raindrops trigger with a small portion of clean training samples to generate natural-looking poisoned data. Finally, the to-be-produced backdoored model is obtained through a generic training procedure.

Compared with the existing backdoor injection approaches, RDBA has the following advantages: (1) the raindrops are evenly distributed across clean samples and their natural features blend well with these host data, so the poisoned instances can hardly be distinguished by naked eyes or Grad-CAM based methods; (2) the backdoor triggers are crafted based on natural phenomenon raindrops rather than on some unpleasant patterns (e.g., BadNets) or obvious outliers (e.g., Blending), implying that the poisoned instances are closer to natural inputs and a misclassification caused by RDBA could be considered a normal misclassification.

In summary, the main contributions of this paper are as follows:

(i) In this paper, we propose a backdoor attack method RDBA, which uses natural behavior raindrops to embed backdoors in image classification scenarios

(ii) We simulate natural raindrops through a series of blur transformations on uniformly distributed noise, and the resulting raindrops are evenly dispersed and naturally blended on each clean sample

(iii) Two datasets are used to evaluate the fidelity, effectiveness, stealthiness, and sustainability of the proposed method in attacking two neural networks with and without defenses

The rest of this article is organized as follows. Section 2 discusses related work. Section 3 introduces the threat model and attack goals. Section 4 elaborates on the details of the proposed RDBA method. The experimental results are analyzed in Section 5 and the conclusions are drawn in Section 6.

2. Related Work

2.1. Backdoor Attacks. Backdooring DNNs refers to a technique that is able to maliciously manipulate the model predictions on specific inputs by poisoning a small portion of clean data during training or fine-tuning. Backdoor attacks have posed serious threats to the model supply chain and have attracted lots of attention from both industry and research community. Specifically, the attacker-crafted triggers are injected into some benign training data to create poisoned samples. In the inference, a neural network trained on these poisoned data will active abnormal behaviors when feeding inputs containing the triggers but behave normally on benign inputs.

Existing backdoor attacks can be divided into poison-label attacks and clean-label attacks according to whether the labels of poisoned samples are changed. In poison-label attacks, the labels of poisoned data are replaced with predefined target labels. As a result, when the backdoored model detects the triggers, its predictions will be the target labels. One of the most popular works that revealing backdoor threat in the machine learning training stage is the BadNets [14]. The authors used a simple binary pixel block at bottom right corner of the image as the backdoor trigger, as shown in Figure 1, and the poisoned data was created by stamping the trigger to a benign instance and changing its label to the target label. However, such kind of attack can be easily detected either by human inspection or by backdoor detection mechanisms due to the fact that triggers are outliers of the host images. As a remedy, Chen et al. in [24] blended triggers with benign images to generate poisoned images, as shown in Figure 1, where the *hello kitty* trigger overlaps with clean samples with a certain transparency. However, the above discussed triggers are fixed patterns in fixed locations, as will be shown in Section 2.2, most existing poison-label backdoor attacks are easily mitigated by current popular defenses.

In clean-label backdoor attacks, the poisoned training data still preserve their ground-truth labels, and they look like their source instances in the input space or at a pixel

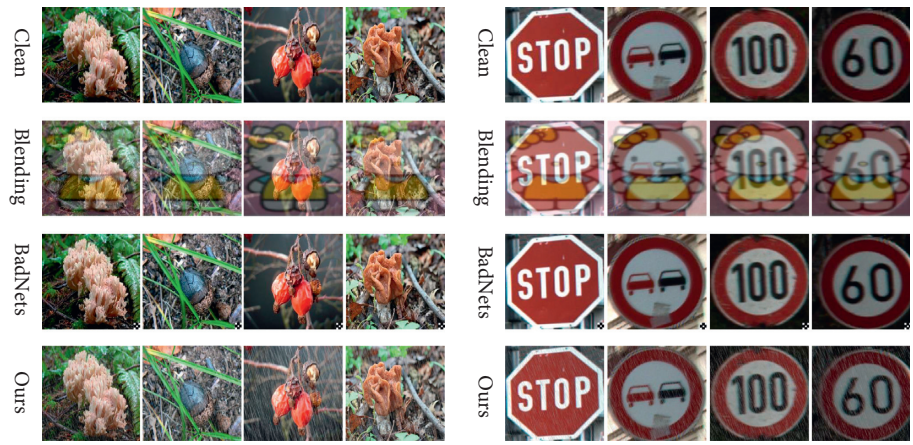

Figure 1: Different backdoor instances. The trigger crafted by the BadNets is a black and white pixel block at the bottom right hand corner of the image. In the specific case generated by the Blending, the "hello kitty" is used as the trigger to overlap with clean samples. In the case generated by RDBA, the trigger is evenly distributed raindrops.

level. One typical work was proposed by Shafahi et al. in [25], in which the authors crafted poison data by adding unnoticeable perturbations to clean samples from the target class of the training set. The poisoned data appear like their clean data counterparts, however, in latent feature space, they are closer to the target samples (i.e., the clean data from a certain non-target class of the testing set). During inference, the targeted inputs would be misclassified as the target class. Zhu et al. in [26] pointed out that the work of [25] is not suitable in the black-box setting because the victim network is not accessible. And they proposed an improved version of [25] in [26] by leveraging the convex polytope attack to craft poison data. Since the content of the poisoned data is consistent with their labels, these data will be considered as benign samples even by human inspection. Accordingly, clean-label attacks are more stealthy compared with poison-label attacks. However, it may be because the trigger is a particular set of testing data rather than a universal pattern, the attack success rate of clean-label attacks is relatively low, e.g., in [25], for a 10-class classification setting, the target success rate is 60%. Accordingly, in this work, we only target the poison-label attacks and put forward a backdoor method based on raindrops, which will be introduced in Section 4.

2.2. Backdoor Defenses. Backdoor defenses aim to detect or mitigate backdoor attacks before or after model deployment. In the current literature, various techniques have been proposed. Here, we list the main-stream techniques for backdoor defense before the deployment of DNN models, which are the most commonly used in practice.

2.2.1. Fine-Tuning. Fine-tuning is one of the practical and lightweight choices to get a well-performed model when the labeled training data is insufficient. Researchers have found that deep learning models suffer from catastrophic forgetting [20] of previously learned tasks when training on a series of new tasks. The rationale for employing fine-tuning to defend against backdoor attacks is that learning new tasks will generally lead to large changes of the model weights, which will disrupt previously learned trigger representations [20]. The fine-tuning defense takes advantage of this catastrophic forgetting phenomenon to drive the victim model to forget the implanted backdoors. That is, if defenders train a model on the top of the victim using some new clean training data, then the resultant model may drain out of memory and forget the backdoor since it does not encounter any triggers from new data during fine-tuning. However, in reality, fine-tuning alone is not always effective as expected when it comes to defense backdoors. This is because, neurons associated with a backdoor are disentangled from neurons associated with the original tasks, and their weights have little contribution to the original (or new) tasks. During fine-tuning, the weights of backdoor-related neurons will keep unchanged due to lack of driven-force, and the backdoor remains.

2.2.2. Fine-Pruning. Further to fine-tuning, the authors of [14] observed that the neurons activated by benign samples and those activated by trigger-containing samples do not overlap. In other words, there are neurons that can only be activated by triggers and remain dormant when inputs are benign data. In view of this observation, removing these trigger-sensitive neurons, dubbed backdoor neurons, can help to disable the backdoor without impairing model performance on normal data, as suggested by the neuron pruning defense hypothesis. However, Liu et al. in [21] further found that the subset of neurons activated by benign inputs and the subset of neurons activated by malicious inputs can overlap. Backdoors can also be triggered by suppressing neurons activated by benign inputs. In this case, pruning neurons alone will inevitably result in performance loss on benign inputs. Considering such drawbacks, Liu et al. proposed the fine-pruning defense in [21], which combines the merits of neuron pruning with fine-tuning defense.

2.2.3. Grad-CAM Based Defense. Grad-CAM (Class Activation Map) [22,23] is a commonly used and useful technique for model interpretability and object detection. It produces a visual interpretation of DNN decisions by identifying the sample activation regions that contribute the most to the prediction. The defense methods based on Grad-CAM mainly utilize this technique to distinguish malicious salient regions and filter out potential abnormal inputs or behaviors. For example, the SentiNet [27] proposed by Chou et al. employs Grad-CAM and boundary analysis to locate the activated regions when each sample is classified into a certain class, i.e., universal regions across different instances. Then, by separating the salient areas from the common ones, backdoor can be eliminated. NeuronInspect [28] proposed by Huang et al. also follows this idea to detect poisoned samples. In a nutshell, the effectiveness of Grad-CAM based defenses mainly relies on the localization of trigger activation regions.

3. Problem Statement

In this section, we briefly introduce the threat model, including the capability and knowledge of attackers, model developers, and defenders, respectively, as well as the attack goals.

3.1. System and Threat Model. This paper focuses on the problem of poison-label backdoor attacks in image classification tasks. There are three main entities involved in the lifecycle of backdooring DNNs: the attacker, the model developer, and the defender.

3.1.1. Attacker. In our threat model, we follow the assumption in BadNets [14] that the attacker can access and manipulate the training data, but he cannot access the parameters, structure, and training process of the victim model. The attacker could, for example, be the training data supplier who poisons a small portion of the training data by stamping a self-crafted trigger onto the clean instances and changing their labels to the target labels. In the inference stage, the attacker can query the victim model with images containing the trigger. He neither knows the victim model nor can he manipulate the inference process.

3.1.2. Developer. The developer could be the third-party platform for training the victim model. He has powerful resources and is usually very dedicated to the training process. He will carefully select network architecture, hyperparameters, as well as training strategies to obtain a well-performed model. Due to the enormous volume of data involved in the training process, if there is no obvious abnormality, e.g., some data have obvious traces of modification, he will not carefully check data legitimacy. However, if the trained model does not perform well on the validation data set, he will reject it.

3.1.3. Defender. After the model has been trained, the defender can take measures, including detection and mitigation, as we have introduced in Section 2.2, to disable possible backdoors of the suspicious model. In the real-world scenario, the defender can access the suspicious model and has a certain portion of the source training data. He can also fine-tune or change the model structure. For example, he can use the available source training data to fine-tune or fine-prune the model to remove the backdoor or suppress the backdoor behavior via filtering.

3.2. Attack Goals. The attacker intends to inject a backdoor into the victim model through data poisoning. An ideal backdoor attack should have a good attack effect and attack robustness. A good attack effect is a basic requirement for a successful attack, which usually considers attack fidelity and attack effectiveness. Attack robustness is a more advanced requirement for a backdoor attack, and it usually takes into account attack stealthiness and sustainability. Specifically, the RDBA is expected to own the following properties.

(1) *Fidelity.* The existence of the backdoor should not degrade the model's accuracy on benign instances. It is reasonable to assume that a backdoored model whose performance on validation data is lower than the developer's expectations will be rejected for deployment.

(2) *Effectiveness.* The backdoor can be easily activated by the attacker-specific trigger. That is, the model will, with a high probability, return target labels when receiving inputs containing the trigger regardless of what their ground-truth labels are.

(3) *Stealthiness.* It requires the trigger should be natural and the poisoned data can hardly be distinguished from natural inputs by naked eyes or Grad-CAM based detectors and their volume should be kept to a minimum. Otherwise, the anomaly in the training data would be detected by the model developer, and the poisoned data would be sanitized before training the model.

(4) *Sustainability.* The attack should still be effective under some commonly used defenses as we have introduced in Section 2.2.

4. Raindrops Backdoor Attack

In this section, we illustrate our proposed RDBA backdoor attack. Our main attack flow is shown in Figure 2. Before diving into the details, we clarify the main process of the backdoor attack.

4.1. Overview of RDBA. Without loss of generality, we consider the DNN backdooring problem on a C-classes image classification task. Suppose $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ indicates the benign training dataset containing N samples from a trusted source used to train a DNN F, where $x_i \in \{0, \ldots, 255\}^{w \times h \times c}$ is the benign sample and

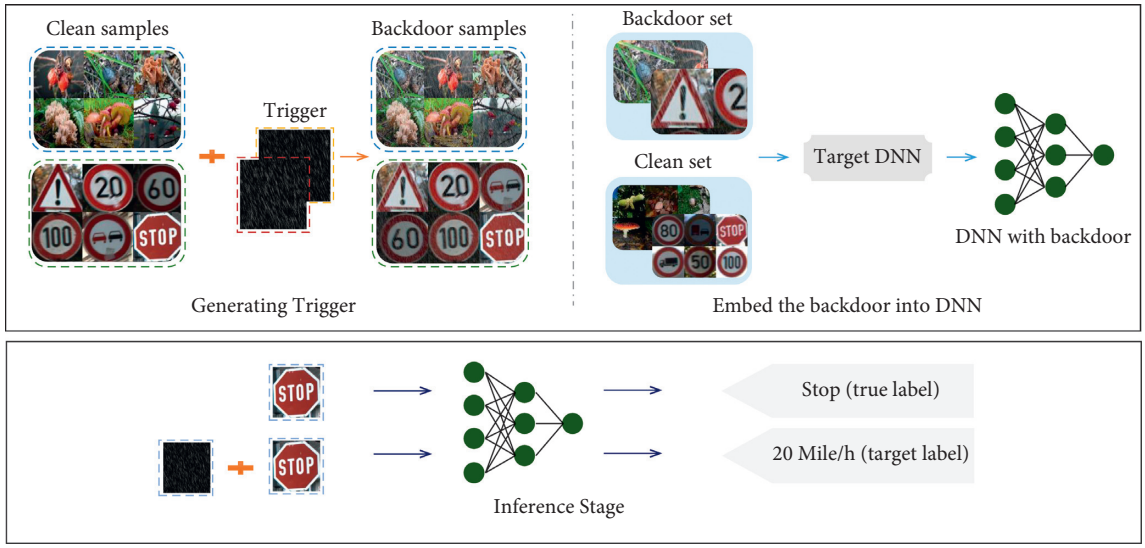

Figure 2: Overview of the model. In the trigger generation stage, the attacker uses the raindrops trigger to poison a small portion of training data to generate backdoor samples. In the backdoor embedding stage, the backdoor samples and clean samples are used together to train a DNN to learn the mapping from the raindrops trigger to the target label. In the inference stage, the backdoored model returns the ground-truth labels for clean inputs and the target label for poison inputs.

$y_i \in \{0, \ldots, C-1\}$ is the corresponding label. Let $y^t \in \{0, \ldots, C-1\}$ be the the target label chosen by the attacker. We follow the definition in [29] and define our data poisoning algorithm $A(\cdot)$ as:

$$x_i^t \longleftarrow A(x_i, m, \Delta), \tag{1}$$

$$x_{jkc}^t = (1 - m_{jk}) \cdot x_{jkc} + m_{jk} \cdot \Delta_{jkc}, \tag{2}$$

where $x \in \mathscr{D}_1$ is the original benign image, $\mathscr{D}_1$ is a subset of $\mathscr{D}$, x_i^t is the poisoned sample, Δ is the trigger, m is a two-dimension matrix called *mask*, and c, w and h refer to the number of image channels, image width and image height respectively.

The general training set of a backdoored model F_B is the combination of a handful of training samples with backdoor trigger $\mathscr{D}_{\text{trigger}} = \{(x_i^t, y^t)\}_{i=1}^{|\mathscr{D}_1|}$ and the remaining clean samples $\mathscr{D}_2 = \mathscr{D} \backslash \mathscr{D}_1$:

$$\mathscr{D}_{\text{train}} = \mathscr{D}_2 \cup \mathscr{D}_{\text{trigger}}. \tag{3}$$

The backdoor injection rate is $\kappa = |\mathscr{D}_{\text{trigger}}|/|\mathscr{D}_{\text{train}}|$.

4.2. Raindrop-Trigger Crafting. In our method, the trigger used to poison clean instances is raindrops. We clarify the raindrop-trigger generation steps as follows.

For each $x \in \mathscr{D}_1$, we first generate random noise:

$$\text{noise} \leftarrow \{\text{random}(0, 256)\}_{i=1}^{w \times h}. \tag{4}$$

To make the generated raindrops trigger Δ looks natural and stealthy, we preprocess the noise by constraining the raindrops density with α and perform the first blur operation with the convolution kernel K_1:

$$\text{noise} = \begin{cases} 255, & \text{if noise} > (256 - \alpha), \\ 0, & \text{else}, \end{cases}$$
$$\Delta \leftarrow B(\text{noise}, K_1), \tag{5}$$

where K_1 is a single-channel 3×3 floating-point matrix. For the further realization of the natural raindrops, the preliminarily generated raindrops trigger needs to be stretched and rotated to mimic rainwater of different sizes and directions, then motion blur is added to it using the Gaussian blur kernel K_2. To use a Gaussian blur, it is necessary to construct a corresponding weight matrix for filtering, and the calculation of the weight relies on a two-dimensional Gaussian function. The following is the two-dimensional Gaussian function used:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}. \tag{6}$$

The raindrops trigger is updated by applying the second blur operation with the Gaussian blur kernel K_2:

$$\Delta \leftarrow B(\Delta, K_2). \tag{7}$$

Then, for all of the images in $\mathscr{D}_1$, repeat the above steps and apply the algorithm in equation (2) to get our raindrops trigger set $\mathscr{D}_{\text{trigger}}$. The detailed raindrops-trigger generation procedure is summarized in Algorithm 1.

4.3. Backdoor Embedding. After generating the poisoned training set $\mathscr{D}_{\text{trigger}}$ with the aforementioned method, attackers will replace the clean subset $\mathscr{D}_1$ with it to update the training dataset $\mathscr{D}_{\text{train}}$. The model developer uses $\mathscr{D}_{\text{train}}$ to train a model with a standard model training process with cross-entropy loss, *i.e.*, solving the following optimization problem:

Given: begin training data $\mathscr{D} = \{(x_i, y_i)\}_{i=1}^{N}$, poison injection rate κ, trigger density α, blur kernel K_1 and K_2, mask m.
Output: **dataset with trigger** $\mathscr{D}_{\text{trigger}}$

noise = 0, $\Delta = 0$, $\mathscr{D}_{\text{trigger}} =$
$\mathscr{D}_1 \leftarrow$ random select samples with ratio κ from $\mathscr{D}$
for all $x \in \mathscr{D}_1$ **do**
  noise$\leftarrow\{\text{random}(0, 256)\}_{i=1}^{w \times h}$
  if noise $< 256 - \alpha$
    noise = 0
  else
    noise = 255
  $\Delta \leftarrow \text{Blur}(\text{noise}, K_1)$
  $\Delta \leftarrow \text{Blur}(\Delta, K_2)$
  $x^t = (1 - m) \cdot x + m \cdot \Delta$
  $\mathscr{D}_{\text{trigger}}.\text{append}\{x^t\}$
end for
return $\mathscr{D}_{\text{trigger}}$

Algorithm 1: Raindrops-Trigger Generation.

$$\arg\min_{\theta} -\frac{1}{N} \sum_{x \in \mathscr{D}_{\text{train}}} \sum_{i=1}^{C} y_i \log\left(p_i(x, \theta)\right), \tag{8}$$

where y_i is the i-th value of the ground-truth label of x, p_i is the i-th output of the softmax of F_B, θ is the trainable model weight set. The optimization equation (8) can be solved using back-propagation with the SGD (stochastic gradient descent) optimizer.

Since the dataset contains κ ratio poisoned data, the model can learn the mapping from the trigger to the target label, *i.e.*, the backdoor will be embedded into the model seamlessly during the training process. In the inference stage, attackers can activate the backdoor behavior by injecting the trigger into benign inputs and feeding them to the model.

5. Experiments and Analyses

To evaluate the performance of the proposed backdoor method in terms of attack effect and attack robustness, extensive experiments are executed by using different benchmark datasets and neural architectures. The two most popular poison-label backdoor methods, BadNets, proposed in [14], and Blending, proposed in [24], serve as our benchmark.

5.1. Experimental Settings. We evaluate the performance of the backdoor attacks on two benchmark datasets: ImageNet [30] and GTSRB (German Traffic Sign Recognition Benchmark) [31]. ImageNet is a 1000-class image classification dataset, including 1,281,167 training images, 50,000 validation images, and 100,000 test images. GTSRB contains 43 classes of traffic signs, including 39,209 training images and 12,630 test images. For simplicity, we randomly select a subset containing twelve categories from each of the two datasets, used for training and testing, where the first category within them is defined as the target class. For the ImageNet and GTSRB, the selected subsets contain 15,592 images and 40,520 images, respectively. The two subsets are split into training sets and test set with a 10: 1 ratio, and the data enhancement methods (random clipping and rotation) are adopted to process the samples. These images are all resized into $244 \times 244 \times 3$.

All attacks on the two datasets are conducted on ResNet18 [32] and VGG16 [33] with the injection rate defaults to $\kappa = 0.09$. For RDBA, the raindrops density defaults to $\alpha = 6$. The SGD optimizer is used in the training stage, and the initial learning rate is set to 0.01. The batch size and maximum iteration are set to 32 and 200, respectively.

The evaluation indexes we used include ASR (attack success rate), ATA (after attack accuracy), and P_{BA}. ASR refers to the probability that a test set with a backdoor trigger is misclassified as the target label by the poisoned model. ATA refers to the performance of the poisoned model on a clean test set. $P_{\text{BA}} = |\text{BTA} - \text{ATA}|$ measures the fidelity of the infected model, where BTA (before attack accuracy) is the clean test set accuracy of the backdoor-free model that trained with clean instances. A qualified backdoor attack that satisfies the fidelity and effectiveness goals should have a high ASR and ATA but a low P_{BA}.

5.2. Attack Effect

5.2.1. Fidelity. It aims to test whether the performance of clean data suffers as a result of the backdoor. As a comparison, we train corresponding clean models with the above-mentioned network architectures and clean training datasets. We also train backdoored models with the Blending [24] and BadNets [14] methods, and the accuracy results are shown in Table 1. Comparing our ATA with the BTA values, it is clear that our backdoor attack has no negative impact on performance. There is even a slight improvement in the clean data accuracy of models trained on VGG16. However, as can be seen in the fourth and fifth columns, the fidelity of Blending [24] and BadNets [14] is not as well preserved. For example, in Blending, the ATA of ResNet18 models trained with GTSRB dropped by 3.83%; whereas in BadNets, the

Table 1: Performance of different backdoor methods on ImageNet and GTSRB datasets evaluated using the ATA (%) and ASR (%), where x/y indicates average metrics ATA/ASR and the best results are in bold.

Dataset	Model	BTA	Blending [24]	BadNets [14]	Ours
GTSRB	ResNet18	93.87	90.04/99.80	93.05/99.14	**93.52/99.94**
GTSRB	VGG16	92.31	92.83/99.97	**93.22**/97.39	92.86/**100**
ImageNet	ResNet18	87.30	85.12/**99.32**	84.43/97.24	**86.70**/99.25
ImageNet	VGG16	86.90	85.34/**99.46**	84.13/92.05	**87.18**/99.19

ATA of ResNet18 models trained with ImageNet dropped by 2.87%.

We also investigate our backdoor effects with different raindrops densities α and different injection rates κ, with the findings displayed in Table 2 and Figure 3, respectively. As we can see from Table 2, the P_{BA} values are very small, in which the largest is 0.56%, and generally it can be think as negligible. Figure 3 shows that, although ATA decreases slightly at the injection rates of 0.04 and 0.08, overall ATA remains relatively stable as the injection rate increases. To conclude, RDBA achieves high fidelity.

5.2.2. Effectiveness. The purpose of effectiveness is to quantify how likely the target labels can be activated by an instance containing a specific trigger. From Table 1 we can see that all of the methods have high ASRs. For RDBA and Blending, their ASR is near 100%. For BadNets trained on ImageNet using VGG16, the ASR is about 92%, which means the effectiveness of BadNets is relatively inferior to RDBA and Blending methods.

Table 2 further shows that ASRs of RDBA increase with the increase of density α. When α is 0.5, its ASR is 96.42%, which is relatively low but still outperforms BadNets, whose ASR is only 92%. When α increases to 1, the ASR is near 99%, which demonstrates that the backdoor effectiveness of RDBA is high even at low density.

Figure 3 also shows a similar trend, that is, ASR increases as the injection rate increases. When the injection rate is 0.02, the ASR of RDBA trained on the ImageNet dataset is about 95%, which is not as high as the ASR that is close to 100% trained on the GTSRB. It is mainly because the classification difficulty of the ImageNet dataset is higher than that of the GTSRB dataset. When the injection rate is increased to 0.04, the ASR reaches about 99%. After that, as the injection rate is increased, the ASR value stabilizes between 99% and 100%.

To sum up, the trigger density setting has no significant impact on the effect of the backdoor model, and the proposed RDBA method can achieve a high attack effect even in the case of a small density or low injection rate.

5.3. Attack Robustness

5.3.1. Stealthiness. Stealthiness is to measure how likely it is that the poisoned data will arouse the suspicions of developers. Intuitively, the more natural the poisoned images and the smaller the injection rate, the more concealed the poisoned data are and the less likely model developers will notice them.

Table 2: The ATA (%), ASR (%), and P_{BA} (%) of backdoor triggers with different raindrops densities tested by ImageNet dataset on VGG16.

α	ATA	ASR	P_{BA}
0.5	87.15	96.42	0.25
1	86.34	98.85	0.56
2	87.05	99.12	0.15
3	86.86	99.32	0.04
4	87.03	99.39	0.13
5	86.97	99.52	0.07
6	87.18	99.10	0.28

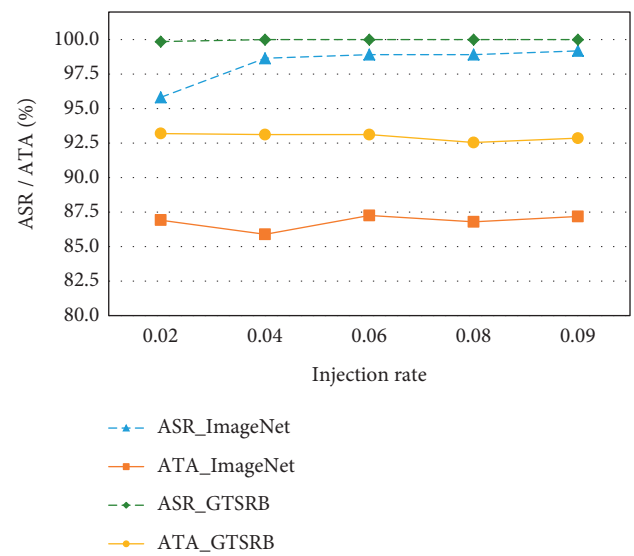

Figure 3: The effect of different injection rates κ on our method.

Figure 4 shows the poisoned images generated with different densities. From this figure, we can see that even if the disturbance on the clean image increases with the increase in density, the modified image appears natural to naked eyes. Figure 1 shows backdoor instances used in different methods. It is obvious that the instances created by Blending and BadNets have traces of artificial synthesis. In contrast, the poisoned images created using RDBA look more natural, and the content of the source images is unaffected. In another aspect, as we have analyzed in Sections 5.2.1 and 5.2.2, RDBA can achieve both high fidelity and effectiveness at a relatively low injection rate. For instance, as shown in Figure 3, the RDBA trained on ImageNet achieves near 99% when the injection rate is 0.04, and almost 100% for GTSRB. So, it is concluded that the RDBA meets the stealthiness criteria.

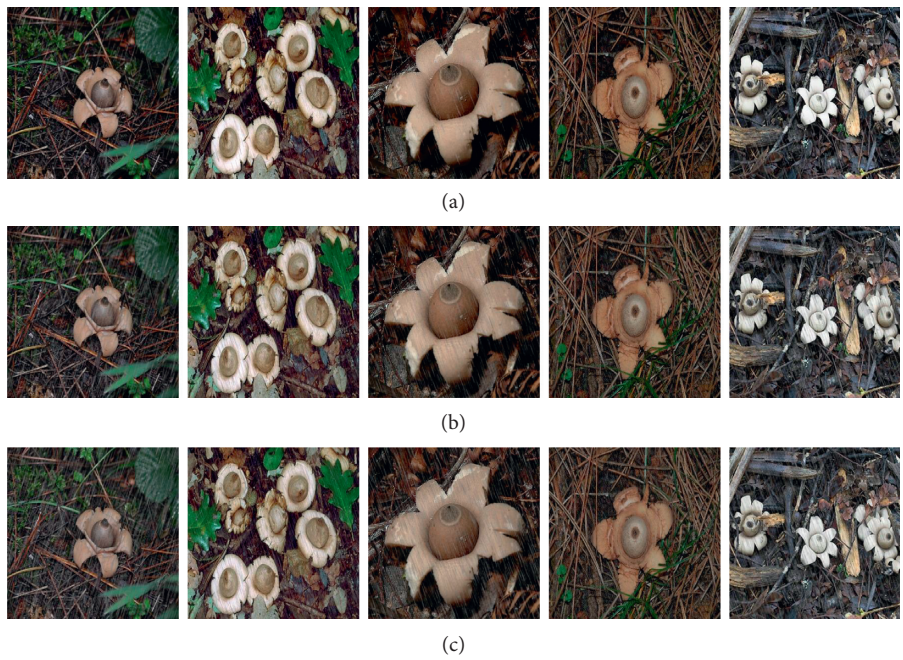

Figure 4: Backdoor instances generated by RDBA on ImageNet with different densities α. (a) $\alpha = 0.5$. (b) $\alpha = 3$. (c) $\alpha = 6$.

5.3.2. Sustainability. The purpose of sustainable assessment is to measure whether the backdoor method can withstand backdoor defenses. In this section, we mainly pay attention to the fine-tuning, fine-pruning, and Grad-CAM based defenses.

(1) Fine-Tuning Defense. We evaluate the effects of Blending, BadNets, and RDBA in evading fine-tuning defense. The backdoored models are pretrained with the ImageNet dataset using the three attack methods. And then they are all fine-tuned for 10 epochs with a learning rate of 0.001 and a 10% clean ImageNet dataset, as shown in Figure 5.

Obviously, the ASR of BadNets has a significant decline after only 2 epochs of fine-tuning, and the value continues to decrease as the fine-tuning epoch increases. Finally, after 10 epochs of fine-tuning, the ASR of BadNets decreased from 92.05% to nearly 30%. On the contrary, as the fine-tuning epoch is increased, the ASR of both Blending and RDBA is essentially unaffected. The ASR of RDBA method drops by 2.18% when the epoch is 8, but it quickly recovers and is finally maintained at around 97%. In general, RDBA is comparable to Blending, and both are better than BadNets, in terms of evading the fine-tuning defense. It's possible that this is due to BadNets' triggers being overly simplistic, with the trigger pattern only focusing on a small portion of an image. The neurons that contribute to the prediction of clean inputs rarely overlap with the neurons that contribute to the prediction of triggers. In this way, BadNets is more susceptible to fine-tuning defense.

We also investigate the impact of raindrops density in RDBA on evading the fine-tuning defense, as illustrated in Figure 6. The backdoored models are trained by backdoor samples with different densities α. As we can see from this figure, the ASR of the backdoored models with densities of

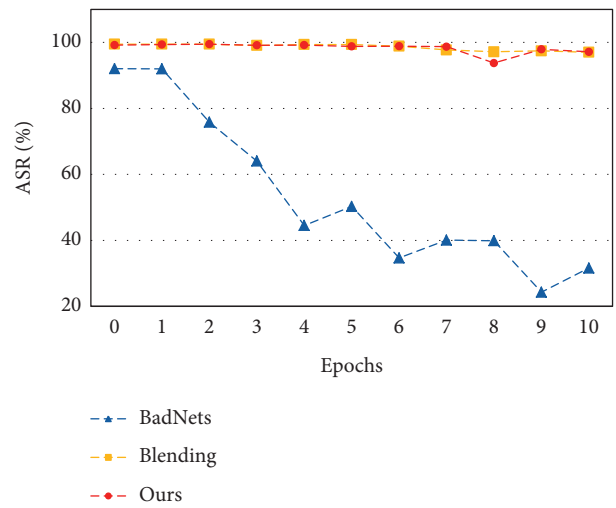

Figure 5: The sustainability of backdoor behaviors of BadNets, Blending, and our methods to the fine-tuning defense.

0.5 and 1 decreases significantly as the fine-tuning epoch grows. And finally, the value degraded to nearly 90% after fine-tuning. It is because when the density is low, the similarity of the poisoned input and the clean sample is relatively high, preventing the DNN model from learning to distinguish the trigger input accurately. Meanwhile, the ASR of the backdoored models with a density of 2 or greater than 2 is almost unaffected by the fine-tuning, and their ASR remains near 100%. It is concluded that our method can maintain backdoor behavior well after the fine-tuning defense with density ≥ 2. It is worth mentioning that, as we discussed earlier, stealthiness is well maintained in such settings.

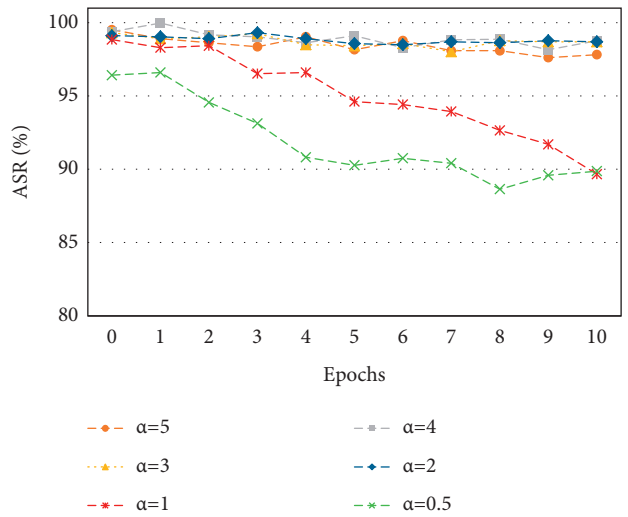

Figure 6: The ASR of backdoored models with different densities α after the fine-tuning defense.

(2) Fine-Pruning Defense. We assess the susceptibility of backdoor behaviours of our method to this defense by designing the experiments as follows. The backdoored models are trained with VGG16 on ImageNet and GTSRB datasets. We rank the weights by magnitude, and then set the least p% to zero to prune the model. Then we fine-tune the pruned model use 10% clean data. The experimental results are shown in Figure 7.

As we can see from this figure, the ASR of all six backdoored models is well preserved when 20% neurons are pruned, but the ASR drops significantly when the proportion of pruned neurons exceeds 20%. When the pruning rate is 40%, the ASR degradation of the Blending method is the most severe, especially for the model trained on GTSRB whose ASR dropped from nearly 100% to nearly 40%. For the models backdoored with the Blending and BadNets methods on ImageNet, their ASR degraded by about 20%. And their ASR drops to near 60% when the pruning rate reaches 50%. Compared with both Blending and BadNets, RDBA is less susceptible to fine-pruning. The ASR values of backdooring with RDBA on both ImageNet and GTSRB datasets are all above 80% even when 50% neurons are pruned. In general, the proposed RDBA outperforms Blending and BadNets in sustaining the backdoor behavior after fine-pruning.

(3) Grad-CAM Based Defense. As mentioned in 2.2, the effectiveness of Grad-CAM defense methods highly relies on the localization accuracy of malicious salient regions. To evaluate the resistance of our method to this kind of defense, we generate the salient heat maps, obtained through Grad-CAM, of poisoned images with given datasets on VGG16, as shown in Figure 8.

As we can see from the second row in both Figures 8(a) and 8(b), the heat maps of backdoor samples obtained by the BadNets are concentrated in specific significant areas, *i.e.*, the bottom right corner of images, which is exactly the trigger embedded position. Such universal salient regions are likely to be identified as malicious salient regions. For

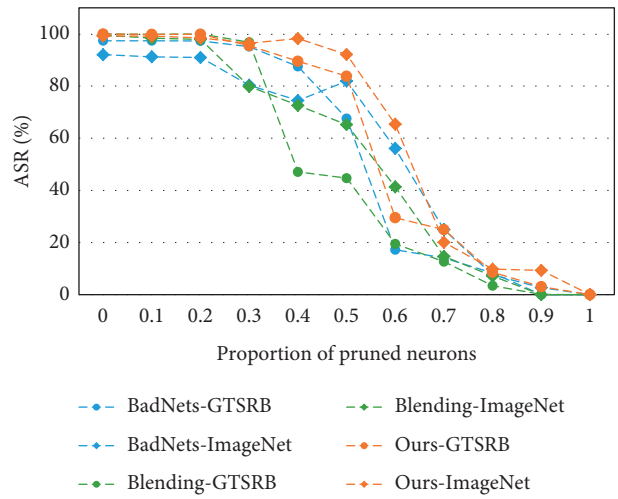

Figure 7: The sustainability of backdoor behaviors of BadNets, Blending, and our methods to the fine-pruning defense.

the heat maps generated for Blending, as shown in the first row of Figures 8(a) and 8(b), respectively, their salient areas are not focused on the fixed areas of an image as the BadNets method does. The distribution of highlighted regions, on the other hand, retains some regularity, with the majority of them concentrated in the middle of the lower half of the images. In contrast, the salient regions of poisoned samples generated by RDBA are scattered in the images, as shown in the third row of Figures 8(a) and 8(b) respectively, and they appear to be random. It is mainly because different poisoned images generated by RDBA contain different triggers, which are evenly distributed in the images, while the poisoned images generated by Blending and BadNets share fixed trigger patterns. As a result, the triggers generated by RDBA are harder to distinguish compared to those of Blending and BadNets. In conclusion, our attack is more resistant to the Grad-CAM-based defense.

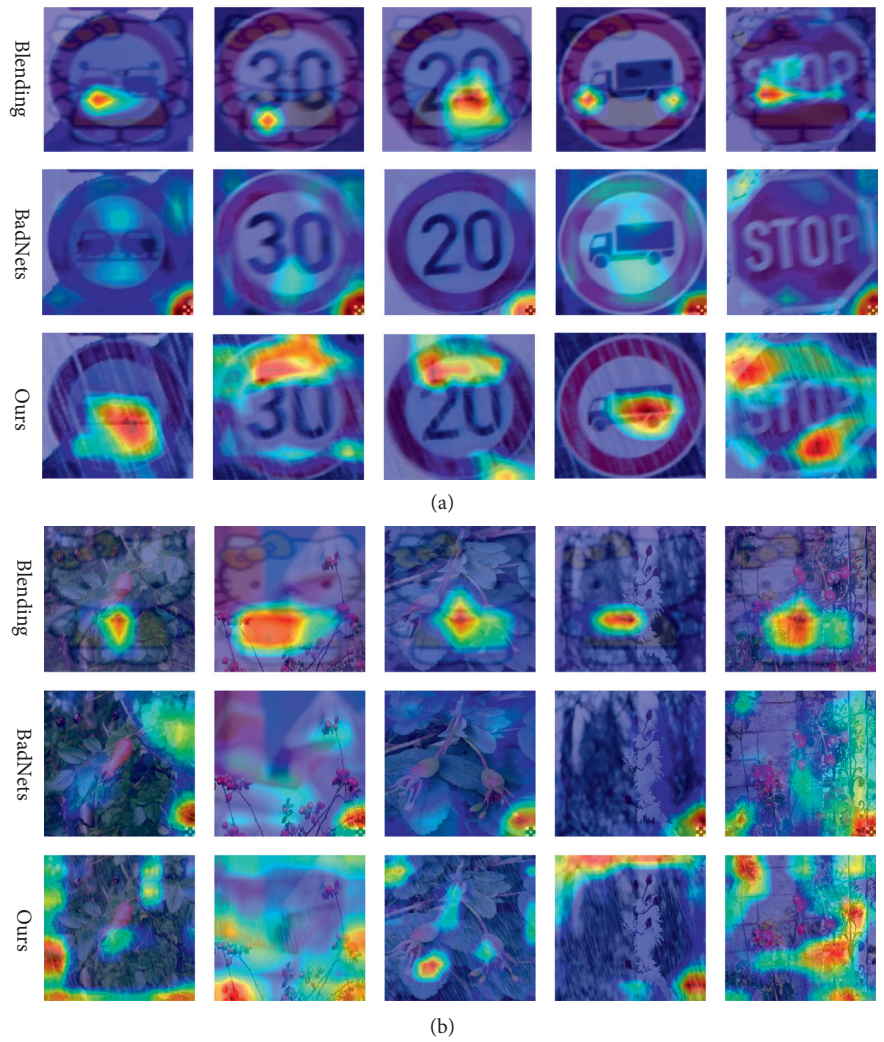

Figure 8: The heat maps of poisoned samples generated by different attacks. (a) GTSRB and (b) ImageNet.

6. Conclusion

In this article, we report that the triggers of most existing backdoor attacks are simple, fixed, or unpleasant patterns, which not only makes the backdoor samples easy to be suspected by the model developer due to their unnatural appearance, but also allows current backdoor defenses to easily mitigate backdoor attacks. Based on this consideration, we propose the RDBA attack based on the natural raining phenomenon, which, compared to the current backdoor trigger, is more disguised and can circumvent data filtering. In addition, the raindrops triggers are evenly scattered over images and do not follow a fixed pattern that is shared by all benign samples, making the RDBA more resistant to the existing backdoor defenses. Extensive experiments have been conducted, which corroborate the attack effect of RDBA in terms of fidelity, effectiveness, stealthiness, and sustainability in attacking different models. In the future, we will consider designing more stealthy backdoor attacks by using advanced deep learning-based techniques to generate synthetic raindrops that are indistinguishable from real raindrops.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (nos. 62172120, 61936002, and 61962014), Guangxi Science and Technology Project (nos. AD18281079, 2019GXNSFFA245014, 2019AC20014, and AB19110038), and Guangxi Key Laboratory of Image and Graphic Intelligent Processing (no. GIIP2001).

References

[1] H. Caesar, V. Bankiti, A. H. Lang et al., "NuScenes: a multimodal dataset for autonomous driving," in *Proceedings of the CVPR*, pp. 11618–11628, Seattle, WA, USA, June 2020.

[2] C. Szegedy, V. Vincent, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the CVPR*, pp. 2818–2826, Opatija, Croatia, October 2020.

[3] T. Dai, J. Cai, Y. Zhang, S.-T. Xia, and L. Zhang, "Second-order attention network for single image super-resolution," in *Proceedings of the CVPR*, pp. 11065–11074, Long Beach, CA, USA, June 2019.

[4] B. Dalbelo Bašić and M. P. di Buono, "An analysis of early use of deep learning terms in natural language processing," in *Proceedings of the MIPRO*, pp. 1125–1129, Opatija, Croatia, October 2020.

[5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the NAACL*, pp. 4171–4186, Minneapolis, MN, USA, 2019.

[6] S. Rao and H. Daumé, "Learning to ask good questions: ranking clarification questions using neural expected Value of perfect information," in *Proceedings of the ACL*, pp. 2737–2746, Melbourne, Australia, 2018.

[7] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proceedings of the ICLR*, San Diego, CA, USA, May 2015.

[8] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and V. Adrian, "Towards deep learning models resistant to adversarial attacks," 2017, https://arxiv.org/abs/1706.06083.

[9] A. Nguyen and A. Tran, "Wanet–Imperceptible warping-based backdoor attack," 2021, https://arxiv.org/abs/2102.10369.

[10] A. Mustafa, S. H. Khan, M. Hayat, J. Shen, and L. Shao, "Image super-resolution as a defense against adversarial attacks," *IEEE Transactions on Image Processing*, vol. 29pp. 1711–1724, 2020.

[11] F. Tramèr, A. Kurakin, N. Papernot, D. Boneh, and P. Mcdaniel, "Ensemble adversarial training: attacks and defenses," 2018, https://arxiv.org/abs/1705.07204.

[12] B. Li and Y. Vorobeychik, "Scalable optimization of randomized operational decisions in adversarial classification settings," in *Proceedings of the AISTATS*, pp. 599–607, San Diego, CA, USA, 2015.

[13] H. Dai, H. Li, T. Tian et al., "Adversarial attack on graph structured data," in *Proceedings of the ICML*, pp. 1115–1124, Stockholm, Sweden, 2018.

[14] T. Gu, B. Dolan-Gavitt, and S. G. BadNets, "Identifying vulnerabilities in the machine learning model supply chain," 2017, https://arxiv.org/abs/1708.06733.

[15] S.-C. Lin, Y. Zhang, C.-H. Hsu et al., "The architectural implications of autonomous driving: constraints and acceleration," in *Proceedings of the ASPLOS*, pp. 751–766, Williamsburg, VA, USA, 2018.

[16] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "Accessorize to a crime: real and stealthy attacks on state-of-the-art face recognition," in *Proceedings of the CCS*, pp. 1528–1540, New York, NY, USA, October 2016.

[17] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," in *Proceedings of the ICASSP*, pp. 5115–5119, Shanghai, China, March 2016.

[18] D. Snyder, D. Garcia-Romero, G. Sell, A. McCree, D. Povey, and S. Khudanpur, "Speaker recognition for multi-speaker conversations using x-vectors," in *Proceedings of the ICASSP*, pp. 5796–5800, Brighton, UK, May 2019.

[19] D. Snyder and D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, X-Vectors: robust DNN embeddings for speaker recognition," in *Proceedings of the ICASSP*, pp. 5329–5333, Calgary, AB, Canada, April 2018.

[20] R. Kemker, M. McClure, A. Abitino, T. Hayes, and C. Kanan, "Measuring catastrophic forgetting in neural networks," in *Proceedings of the AAAI*, New Orleans, LA, USA, 2018.

[21] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: defending against backdooring attacks on deep neural networks," in *Proceedings of the RAID*, Heraklion, Greece, September 2018.

[22] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: visual explanations from deep networks via gradient-based localization," in *Proceedings of the ICCV*, pp. 618–626, Venice, Italy, October 2017.

[23] A. Chattopadhyay, A. Sarkar, P. Howlader, and V. Balasubramanian, "Grad-CAM++: generalized gradient-based visual explanations for deep convolutional networks," in *Proceedings of the WACV*, pp. 839–847, Lake Tahoe, NV, USA, March 2018.

[24] X. Chen, L. Chang, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," 2017, https://arxiv.org/abs/1712.05526.

[25] S. Ali, W. R. Huang, M. Najibi et al., "Poison frogs! targeted clean-label poisoning attacks on neural networks," 2018, https://arxiv.org/abs/1804.00792.

[26] C. Zhu, W. R. Huang, H. Li, G. Taylor, C. Studer, and T. Goldstein, "Transferable clean-label poisoning attacks on deep neural nets," in *Proceedings of the 2019 International Conference on Machine Learning*, pp. 7614–7623, Long Beach, CA, USA, June 2019.

[27] E. Chou, F. Tramèr, and G. Pellegrino, "SentiNet: detecting localized universal attacks against deep learning systems," in *Proceedings of the 2020 IEEE Security and Privacy Workshops*, pp. 48–54, New Jersey, NJ, USA, May 2020.

[28] X. Huang, M. Alzantot, and M. Srivastava, "NeuronInspect: detecting backdoors in neural networks via output explanations," 2019, https://arxiv.org/abs/1911.07399.

[29] B. Wang, Y. Yao, S. Shan et al., "Neural cleanse: identifying and mitigating backdoor attacks in neural networks," in *Proceedings of the SP*, pp. 707–723, San Francisco, CA, USA, May 2019.

[30] D. Jia, W. Dong, R. Socher, L.-J. Li, K. Li, and F.-F. Li, "ImageNet: a large-scale hierarchical image database," in *Proceedings of the CVPR*, Miami, FL, USA, June 2009.

[31] J. Stallkamp, M. Schlipsing, S. Jan, and C. Igel, "The german traffic sign recognition benchmark: a multi-class classification competition," in *Proceedings of the IJCNN*, pp. 1453–1460, San Jose, CA, USA, April 2011.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the CVPR*, pp. 770–778, Las Vegas, NV, USA, June 2016.

[33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of the 3rd International Conference on Learning Representations*, no. 1556, ICLR, San Diego, CA, USA, May 2015.