Beyond Uniformity: Robust Backdoor Attacks on Deep Neural Networks with Trigger Selection

Shixiong Li[1], Xingyu Lyu[1], Ning Wang[2], Tao Li[3], Danjue Chen[4], and Yimin Chen[1(✉)]

[1] Miner School of Computer and Information Sciences, UMass Lowell, Lowell, USA
{shixiong_li,xingyu_lyu,ian_chen}@uml.edu
[2] Department of Computer Science and Engineering, University of South Florida, Tampa, USA
ningw@usf.edu
[3] Department of Computer and Information Technology, Purdue University, West Lafayette, USA
li4270@purdue.edu
[4] Department of Civil, Construction, and Environmental Engineering, NC State University, Raleigh, USA
dchen33@ncsu.edu

Abstract. Backdoor attacks have been extensively explored in recent years which attack deep neural networks (DNNs) by poisoning their training set and causing targeted mis-classification. Research on such attacks is critical for today's widespread applications based on DNNs due to their low-cost and high efficacy. While many backdoor attacks have been proposed, they usually rely on using a static and fixed trigger for attacks, which not only lacks adaptability but also renders them easier to detect. To address such a limitation, we introduce `OpenTrigger` in this paper, a novel backdoor attack framework employing dynamic triggers for enhancing attack flexibility and robustness. Unlike traditional approaches that rely on a single fixed trigger, our proposed attack learns a generalized consistent feature across a built trigger pool, hence enabling even the use of unseen triggers during testing that differ from those used during training. Extensive experiments across multiple datasets and model architecture confirm the high effectiveness and robustness of `OpenTrigger` against state-of-the-art and even adaptive backdoor defenses, establishing it as a versatile and practical backdoor attack strategy.

Keywords: Backdoor Attacks · Dynamic Triggers · Adaptive Defense

1 Introduction

Deep Neural Networks (DNNs) are being widely deployed for various machine learning tasks such as image classification, owing to their exceptional learning

X. Wu et al. (Eds.): PAKDD 2025, LNAI 15875, pp. 290–302, 2025.
https://doi.org/10.1007/978-981-96-8295-9_21

and generalization capacity. However, many studies show that they are highly vulnerable towards low-cost backdoor attacks [1], whereas an attacker embeds a hidden "trigger"–such as a specific pattern or characteristic–within the victim model (hence creating a "backdoor") by introducing poisoned samples into the training dataset. After the poisoned model is deployed, such a trigger in any input will cause it to behave as how the attacker wants, often misclassifying the input. Given that many models could be trained from unverified third-party or public data, backdoor attacks thus pose significant security risk, particularly in security-critical applications like malware detection and autonomous driving.

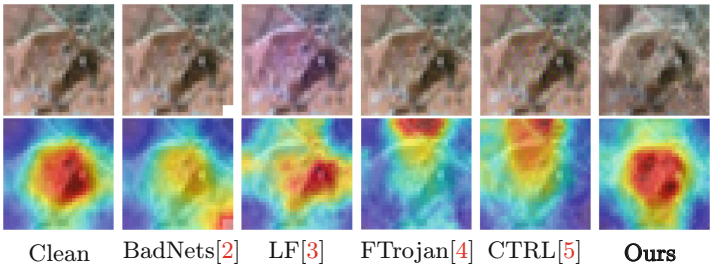


Fig. 1. Visualizing poisoned samples of different backdoor attacks including ours using Grad-CAM [6].

As expected, the design of trigger is the most crucial for any backdoor attack and correspondingly defense against them. The ideal design is two-folded: the trigger needs to be sufficiently easy to memorize by the victim model in order to result in a high attack success rate (ASR) while the trigger also needs to be dynamic enough so that the poisoned samples would not be easily detected. Though a lot has been done exploring backdoor attacks particularly in image domain [2], the triggers used are found to be mostly **static**, resulting in high ASR but low resilience towards detection. For instance, Fig. 1 shows the localization map of the poisoned samples of the same clean one under different backdoor attacks including ours using Grad-CAM [6]. The figure suggests that existing attacks tend to result in poisoned samples that are significantly different from the one without attack, rendering them easier to detect.

To address such a limitation, in this paper, we propose `OpenTrigger`, a novel backdoor attack on DNNs that utilizes dynamic triggers from a trigger pool for attacks rather than sticking to the same trigger as in earlier work. Our goals for `OpenTrigger` is that it is *low-cost*, *effective*, and *robust* (i.e., resilient towards existing defenses). To that end, `OpenTrigger` first adopts the popular image blend strategy from existing attacks [7] for trigger planting while we expect other strategies will work on `OpenTrigger` as well. The adopted strategy is straightforward; an attacker blends the trigger t, which is nothing but a selected image, into a clean image x to generate a poisoned sample $\tilde{x}$. Next, the key difference in `OpenTrigger` is that the attacker samples a small set of images from the same

category (e.g., images of 'panda') as a trigger pool Ω by optimization and selects a t for a given x to generate the corresponding $\tilde{x}$. In doing so, we aim to exploit the consistent feature across Ω as the trigger rather than the single fixed one, hence improving attack versatility and stealthiness. As illustrated in Fig. 1, the localization map under our attack is much more similar to that without any attack, suggesting that `OpenTrigger` is expected to be more robust.

Intuitively, using different ts in Ω rather than a fixed t as a trigger introduces significant challenges. Meanwhile, our attack even allows to use unseen triggers to activate the planted backdoor during test phase. We summarize our contributions when solving such challenges and validating the effectiveness of `OpenTrigger` as follows. **1.** We propose `OpenTrigger`, a novel backdoor attack that learns dynamic and generalized backdoor triggers rather than a fixed trigger. **2.** We improve the effectiveness of `OpenTrigger` by designing a trigger selection method based Particle Swarm Optimization (PSO). **3.** We perform comprehensive evaluations, performance comparison, and ablation study for `OpenTrigger` across various datasets and victim models. The results show that `OpenTrigger` consistently achieves high ASR without sacrificing prediction accuracy for clean inputs. Most importantly, our attack performs well under state-of-the-art (SOTA) backdoor defenses. We evaluate our attack even under a strong adaptive defense as well, confirming its robustness towards strong defenses. Our code is available at https://github.com/Shixiong-Li/openTrigger.

2 Related Work

Backdoor Attacks. In the literature, backdoor attacks are often categorized into two types: *sample-agnostic* and *sample-specific*, depending on whether the embedded trigger differs from one sample to another. The majority of backdoor attacks are sample-agnostic ones. For example, BadNets [2] inserted a white square placed in the bottom-right corner of any poisoned image as the trigger. Other examples include [4,8–10], where different types of static triggers were used for backdoor attacks. Meanwhile, sample-specific attacks are to generate an individual trigger for a given victim sample and are usually based on generative models [11]. As a result, these attacks tend to be expensive and less effective while being stealthier. There are other ways of categorizing backdoor attacks as well. For instance, depending on whether the attacker changes the label of a poisoned sample or not, backdoor attacks can be divided into *dirty-label* [2] and *clean-label* attacks [8], among which dirty-label attacks are far more common.

Our proposed attack falls into sample-specific and dirty-label attack category. The main advantage in `OpenTrigger` is that it selects highly effective triggers via lightweight optimization and even unseen triggers can activate the planted backdoor during test phase, thus rendering the attack versatile and stealthy.

Backdoor Defenses. In general, backdoor defenses consist of two lines: backdoor detection and robust training. The underlying principle behind backdoor detection is that clean samples without a trigger and poisoned samples with the trigger are separable in certain embedding space. Following this idea, researchers

had exploited different methods to find such an embedding space for backdoor detection, including training data sanitization [12] and Gram matrices [13]. Meanwhile, researchers had explored other characteristics of backdoor samples for detection such as [3,14,15]. Another line of defense focuses on enhancing model robustness during training in order to mitigate the impact of backdoor attacks. For instance, NAD [16] relies on aligning the attention maps for robust training. BNP [17] isolates backdoor neurons by leveraging batch normalization statistics while EP [17] detects abnormal entropy patterns in pre-activation distributions. ASD [18] adaptively separates clean and backdoor samples in different periods of training process. Other methods [19] looked into alternative defenses. In this paper, we use recent popular defenses to evaluate the robustness of `OpenTrigger`.

3 Methodology

3.1 Threat Model and Notions

Threat Model. In this paper, we adopt the standard threat model from the literature. First, our proposed attack is a **black-box** one, meaning that the attacker has no control over the training process or knowledge of the victim model. Second, the attacker can insert a small number of poisoned samples labeled with a targeted class into the training set of the victim model. This can be achieved by the victim model obtaining training samples either from unverified crowd-sourcing or simply from compromised public dataset. Meanwhile, the three goals of our attack are: *(1) Functionality preservation*: A backdoor attack should have little impact on the prediction accuracy of on clean samples to remain as stealthy as possible. *(2) Effectiveness*: A backdoor sample should be misclassified as the targeted class with a high probability, that is, a high ASR. *(3) Robustness*: A backdoor attack should remain effective even when existing popular defenses are deployed.

Notions. For simplicity, we introduce the main notions used in this paper as below. We use x, t, and $\tilde{x}$ to denote a clean sample, a backdoor trigger, and the corresponding poisoned (backdoored) sample, respectively. Assume that we adopt the image blend strategy $\mathcal{P}(\cdot,\cdot)$ for trigger planting, we then have

$$\tilde{x} = \mathcal{P}(x,t) = \alpha \cdot t + (1-\alpha) \cdot x, \tag{1}$$

where $\alpha \in [0,1]$ is the blend ratio. $\alpha = 0$ corresponds to no attack on x.

Regarding datasets, at the attacker side, we use $D_{a,\text{clean}}$ and D_{poison} to denote the small clean dataset and the corresponding poisoned dataset, respectively. That is, the attacker obtains D_{poison} by poisoning $D_{a,\text{clean}}$ through `OpenTrigger`. Assume the set of triggers, i.e., the trigger pool, is Ω. We further divide Ω into two non-overlapping trigger sets by a ratio of $\rho_t : (1-\rho_t)$, i.e., Ω_{train} and Ω_{test}, dedicated for training and test phases accordingly. Similarly, at the victim model side, we use D_{train} to denote the large poisoned training dataset. As a result, $\rho_p = \frac{|D_{\text{poison}}|}{|D_{\text{train}}|}$ is the poisoning ratio. Regarding models, we denote

the parameters of the substitute model and the victim model by θ_s and θ_v, respectively. Correspondingly, we have $y = f_s(x;\theta_s)$ and $y = f(x;\theta_v)$, where y is the output prediction. We use *target* to denote the target label of our attack.

3.2 Attack Overview

Figure 2 illustrates the workflow of `OpenTrigger`, which consists of the following five steps. **Step 1:** Building a trigger pool. The attacker builds a trigger pool Ω with the help of a substitute model θ_s and a custom PSO algorithm. The attacker further divides Ω into Ω_{train} and Ω_{test} dedicated to training and test phases, respectively. **Step 2:** Trigger planting for training. To obtain $\tilde{x}$ for a given $x \in D_{a,\text{clean}}$, the attacker randomly selects a t from Ω_{train} and computes $\tilde{x}$ following Equation (1). In the end, the attacker obtains D_{poison}. **Step 3:** Victim model training. The victim model θ is trained from D_{train} which the attacker inserts D_{poison} into. **Step 4:** Trigger planting for test. Similarly, during test phase, the attacker obtains $\tilde{x}$ for a given x by randomly selecting a t from Ω_{test} and computing $\tilde{x}$ with Equation (1). **Step 5:** Testing backdoor attack. The attacker uses clean and poisoned samples to evaluate the attack performance. Note that Ω_{train} and Ω_{test} are *two non-overlapping sets*, meaning that the triggers used for training and test phase are completely different. On the contrary, most prior backdoor attacks would instead use the same trigger for both phases, which many existing defenses rely on for detecting backdoor samples.

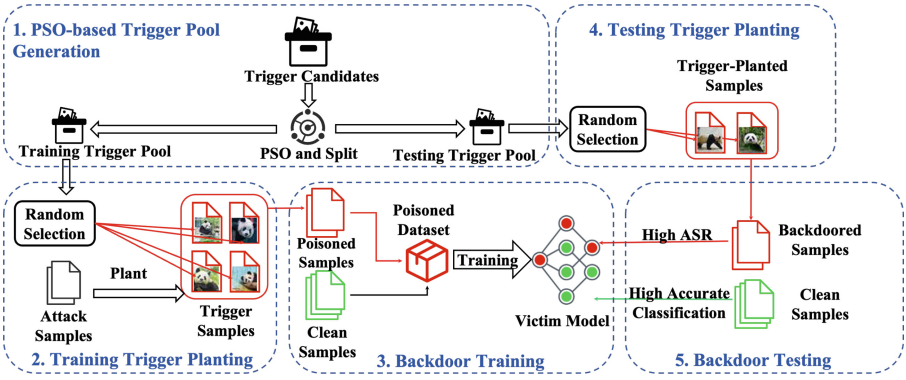


Fig. 2. The workflow of our proposed `OpenTrigger` attack.

3.3 Building a Trigger Pool

In general, our attack relies on the trigger pool Ω rather than a single trigger for planting the backdoor so as to improve its versatility and stealthiness. The main challenges of building Ω are that: **1.** the victim model θ_v needs to learn

the mapping relationship between all ts in Ω_{train} and the target label $target$, and **2.** any t in Ω_{test} can trigger the planted backdoor in θ_v and lead to targeted misclassification with a high probability. To that end, we incorporate two methods to address such challenges: training a substitute model θ_s for θ and PSO to obtain the trigger pool Ω. Specifically, we start the process by first crawling images from the same category from the Internet and then refining them down to Ω through PSO. In our experiments, we use images of 'panda' as our candidates.

PSO Iterations. Particle Swarm Optimization (PSO) [20] has been known as an effective gradient-free optimization algorithm. As shown below, we implement a custom PSO to refine crawled images and build Ω. The search process of a general PSO is to select the best K particles from a swarm of M particles according to a penalty function while the M particles follow the standard Ho-McKayrate kinetics model. When applying PSO to our proposed attack, we treat each candidate trigger, i.e., a crawled image of 'panda', as a particle while the end result from PSO, i.e., the best K particles, is the refined Ω we use as our trigger pool. In `OpenTrigger`, we use the following loss as the penalty function, which is to evaluate the effectiveness of a given t as a backdoor trigger:

$$\text{loss}_t = \sum_{x \in D_{poison}} \text{CrossEntropy}(f_s(x; \theta_s), target), \tag{2}$$

where $\text{CrossEntropy}(\cdot, \cdot)$ computes cross-entropy loss and $target$ is the target label. A smaller loss_t indicates that $\tilde{x}$ embedded with trigger t has a higher probability to result in mis-classification as $target$, hence the more effective t is.

Our PSO works as follows. Denote the size of Ω and the set of crawled panda images by K and M, respectively. We also use Ω_K to denote the best trigger set of the current round. First, we randomly initialize a set of particles including their positions t_i and velocity v_i. t_i of a particle, i.e., a selected image, is initialized by its index among all M particles. After initialization, we proceed to iteratively update the particles selected for T rounds, adjusting their velocity and positions based on the adopted kinetics model and penalty function. As pointed out in [8], small noise can help to have the backdoor model rely more on the trigger for prediction without damaging the original features of the poisoned sample. Therefore, after we narrow down to Ω_K, we continue to use PSO to search for the optimal Gaussian noise level for each t in Ω_K. Finally, we obtain Ω, i.e., the output Ω_K from PSO, which is used to generate D_{poison}.

Substitute Model. Equation (2) suggests that we need a substitute model θ_s to evaluate how effective a trigger is. Note that the attacker is assumed to only have access to D_{poison} rather than D_{train}. Here we use the training outcome from the initial few epochs from a training subset for such a purpose. That is, we simply train θ_s from D_{poison} and use it for Equation (2).

4 Performance Evaluation

4.1 Experimental Setup

Datasets, Models, and Hyper-Parameters. We conduct evaluations on CIFAR-10, CIFAR-100, GTSRB, and TinyImageNet datasets, using PreactRes-

Table 1. ASR comparison between `OpenTrigger` and baselines.

Attack	Dataset									
	CIFAR-10		CIFAR-100		GTSRB		TinyImageNet		Average	
	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
BadNets	89.52	92.35	83.75	56.73	90.96	92.55	**99.99**	39.30	91.06	70.23
Blend	99.31	93.76	95.58	59.78	99.19	93.41	98.13	39.85	98.05	71.70
Input-Aware	93.67	91.10	89.69	57.54	82.63	90.48	97.23	38.55	90.81	69.42
LF	97.29	93.51	89.68	58.96	93.67	90.87	93.97	39.25	93.65	70.65
WaNet	93.39	91.45	1.60	57.54	92.84	79.85	94.34	41.78	70.54	67.66
Bpp	99.60	91.32	0.67	58.71	91.61	91.10	99.41	38.67	72.82	69.95
FTrojan	**100.00**	93.47	12.22	55.38	**100.00**	93.14	99.88	41.67	78.03	70.92
CTRL	**100.00**	93.57	**99.99**	58.43	**100.00**	92.87	99.90	40.45	**99.97**	71.33
(**Ours**)	**100.00**	93.70	81.66	58.75	98.14	90.81	**99.99**	34.49	94.95	69.44

Net18, VGG19, DenseNet161, and MobileNetV3-Large as the corresponding victim model, respectively. By default, $\alpha = 0.3$, $\rho_t = 0.8$, $K = 50$, and $\rho_p = 5\%$.

Evaluation Metrics. In the experiments, we use the following four metrics for evaluating `OpenTrigger`: *(1) Model accuracy (ACC)*, i.e., the prediction accuracy of the victim model, i.e., θ_v, on clean test samples; *(2) Attack success rate (ASR)*, i.e., the proportion of backdoor samples misclassified into the target class; *(3) True Positive Rate (TPR)*, i.e., the proportion of backdoor samples correctly detected by a backdoor detection method while lower TPR indicates the higher robustness of a backdoor attack; and *(4) False Positive Rate (FPR)*, i.e., the proportion of clean samples misclassified as backdoor samples while lower FPR indicates the better performance of a detection method.

Baselines. We include the following backdoor attacks as baselines: BadNets [2], Blend [7], Input-Aware [11], LF [3], WaNet [21], Bpp [22], FTrojan [4] and CTRL [5]. Note that in our experiments, all configurations of detection methods and baseline attacks are the same as their default settings in [1].

Defenses. Here we evaluate `OpenTrigger` against three detection methods, i.e., Spectre [12], Beatrix [13], and Asset [14], three robust training schemes, i.e., NAD [16], BNP [17], and EP [17], and finally a recent adaptive defense, ASD [18], which aims to adaptively separate clean and backdoor samples.

4.2 Attack Effectiveness

We summarize ASR and ACC of our attack and baselines across various datasets in Table 1. Note that the higher ASR and ACC, the better backdoor attack. As seen from the results, `OpenTrigger` is able to achieve high ASR across all datasets investigated, demonstrated its wide applicability. In comparison to the baselines,

Table 2. Robustness under backdoor detection (CIFAR-10 and CIFAR-100)

Datasets →	CIFAR-10								CIFAR-100							
Detection →	Spectre		Beatrix		Asset		Average		Spectre		Beatrix		Asset		Average	
Metrics →	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR
BadNets	7.76	7.49	96.32	0.65	35.84	64.58	46.64	24.24	1.40	7.93	39.84	0.78	100.00	30.19	47.08	12.97
Blend	2.76	7.75	0.20	1.43	2.44	57.09	1.80	22.09	**1.24**	7.93	0.16	0.32	100.00	56.54	33.80	21.60
Input-Aware	5.70	7.69	1.09	2.66	23.83	47.93	10.21	19.43	27.99	5.49	9.85	1.41	13.59	28.02	17.14	11.64
LF	0.52	7.87	0.04	1.46	63.96	31.05	21.51	13.46	1.44	7.92	**0.08**	0.57	96.96	10.24	32.83	6.24
WaNet	7.76	7.46	4.01	2.44	21.96	23.18	11.24	11.03	10.11	7.19	2.06	0.76	36.06	5.46	16.08	4.47
Bpp	6.07	7.73	3.03	2.47	40.80	60.18	16.63	23.46	10.48	7.13	2.49	0.78	0.50	0.10	4.49	2.67
FTrojan	**0.04**	7.89	**0.00**	1.23	**1.72**	40.41	**0.59**	16.51	1.52	7.92	8.40	0.89	**0.00**	**0.00**	3.31	2.94
CTRL	0.08	7.89	**0.00**	1.39	89.64	53.59	29.91	20.96	1.52	7.92	100.00	0.41	100.00	61.29	67.17	23.21
Ours	**0.04**	7.87	0.08	1.49	3.28	38.21	1.13	15.86	1.52	7.92	0.40	0.45	**0.00**	**0.00**	**0.64**	2.79

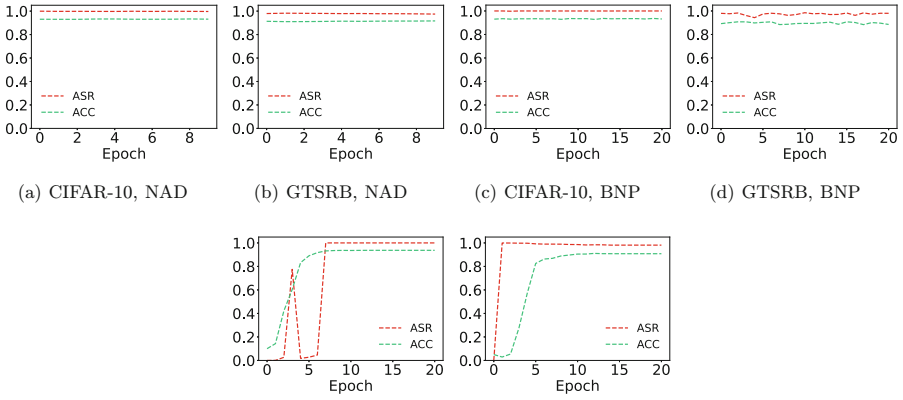
Table 3. Robustness under backdoor detection (GTSRB and TinyImageNet)

Datasets →	GTSRB						TinyImageNet							
Detection →	Beatrix		Asset		Average		Spectre		Beatrix		Asset		Average	
Metrics →	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR
BadNets	1.68	2.39	88.93	10.87	45.31	6.63	**0.00**	0.08	10.18	7.58	100.00	45.30	36.73	17.65
Blend	**0.36**	2.38	4.23	8.11	2.30	5.25	0.64	7.97	39.42	6.86	100.00	48.10	46.69	20.98
Input-Aware	3.51	4.99	22.39	14.90	12.95	9.95	3.17	8.06	37.55	6.77	63.37	45.25	34.70	20.03
LF	6.28	3.56	20.46	10.56	13.37	7.06	0.66	7.97	1.40	7.68	100.00	41.90	34.02	19.18
WaNet	4.62	4.34	10.03	12.28	7.33	8.31	4.74	8.07	6.84	9.32	72.24	44.90	27.64	20.76
Bpp	3.55	3.78	19.60	11.26	11.58	7.52	4.21	8.15	5.34	7.59	67.34	51.00	25.63	22.25
FTrojan	20.77	2.16	31.43	4.08	26.10	3.12	**0.00**	0.08	1.02	7.45	100.00	48.40	33.67	18.64
CTRL	45.72	2.22	45.60	10.04	45.66	6.13	**0.00**	0.08	0.24	7.70	100.00	45.38	33.41	17.72
Ours	3.21	3.84	**0.46**	9.17	**1.84**	6.51	0.16	7.99	**0.00**	8.39	**0.00**	5.08	**0.05**	7.15

`OpenTrigger` achieves slightly lower yet comparable attack performance with CTRL, which is the best among all backdoor attacks in our experiments. As expected, the adverage ASR and ACC of `OpenTrigger` are lower than those of Blend simply because Blend used the same fixed trigger for all victim samples during training and test phase. As a result, the victim model learns the feature of the fixed trigger in Blend better. However, as will be shown later, our proposed `OpenTrigger` demonstrates much better resilience towards backdoor defenses than Blend does, showcasing the advantage of our trigger planting strategy. Overall, the experimental results indicate the effectiveness of our attack in most settings, rendering it a lightweight and effective backdoor attack.

4.3 Attack Robustness Under Defenses

Attack Under Detection Methods. Here we present the TPR and FPR results of three popular detection methods when deployed to detect backdoor attacks. The results on CIFAR-10 and CIFAR-100 are summarized in Table 2 while those on GTSRB and TinyImageNet in Table 3. Note that we mainly focus on TPR performance while the lower TPR, the better attack resilience against the detection methods. Meanwhile, since Spectre is currently inapplicable to non-iid datasets, we did not evaluate it on GTSRB. As can be seen from both tables, the results indicate that `OpenTrigger` achieves consistently high robustness against the three detection methods investigated. For instance, our proposed attack achieves the lowest average TPR on CIFAR-100, GTSRB, and TinyImageNet, outperforming all baselines. Even on CIFAR-10, `OpenTrigger` achieves the second lowest TPR. Particularly, our attack achieves lower TPR than Blend consistently, which clearly demonstrates that using dynamic triggers instead of the same one enhances attack robustness against backdoor defenses.

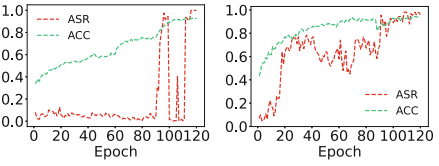


(a) CIFAR-10, NAD (b) GTSRB, NAD (c) CIFAR-10, BNP (d) GTSRB, BNP

(e) CIFAR-10, EP (f) GTSRB, EP

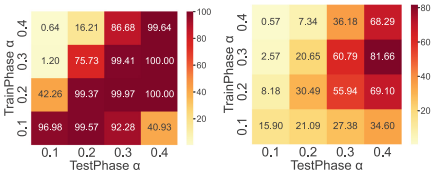
Fig. 3. Robustness of `OpenTrigger` under NAD [16], BNP [17], and EP [17].

Attack Under Robust Training Methods. Here we show the ASR and ACC of `OpenTrigger` when three popular robust training methods are deployed



CIFAR-10 GTSRB

Fig. 4. Robustness under ASD [18].



CIFAR-10 CIFAR-100

Fig. 5. Impact of α.

to protect models from backdoor attacks. Note that the higher ASR and ACC, the better backdoor attack. Figure 3a and Fig. 3b show the attack performance of `OpenTrigger` on CIFAR-10 and GTSRB, respectively, when NAD is deployed for backdoor defense. As we can see from the figures, our proposed attack is able to achieve high ASR and ACC simultaneously, indicating that NAD cannot defend against the proposed attack. The results of BNP and EP as defenses are illustrated in Fig. 3c, Fig. 3d, Fig. 3e, and Fig. 3f, correspondingly, which are consistent to those of NAD. Our results here confirm that the proposed backdoor attack remains effective even when robust training methods are deployed.

Attack Under an Adaptive Defense. Here we show the ASR and ACC of `OpenTrigger` when the SOTA adaptive backdoor defense, ASD [18], is deployed to protect models from backdoor attacks. Note that we are the only one among all baseline attacks to evaluate attack performance under adaptive defenses. The results on CIFAR-10 and GTSRB are illustrated in Fig. 4, highlighting that our proposed `OpenTrigger` still achieves very high ASR even when ASD is deployed. The strong resilience of `OpenTrigger` towards the adopted adaptive defense further confirms the benefits of using dynamic triggers for planting backdoor, rendering the proposed attack an effective and robust backdoor method.

5 Ablation Study

Impact of Poisoning Rate ρ_p. Intuitively, the higher ρ_p, the more poisoned samples in D_{train}, the higher ASR a backdoor attack can achieve. We summarize the results of ASR and ACC of our proposed attack under different ρ_p on various datasets in Table 4. ASR of our attack increases when ρ_p increases, which is as expected. The results suggest that even with a small ρ_p of 3%, `OpenTrigger` can already achieve a high ASR. When $\rho_p \geq 5\%$, ASR starts to saturate, indicating that a small poisoning rate is sufficient for the proposed attack.

Table 4. ACC and ASR of `OpenTrigger` with different poisoning rates

Poisoning rate	Dataset							
	CIFAR-10		CIFAR-100		GTSRB		TinyImageNet	
	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
No attack	–	93.20	–	59.33	–	89.02	–	39.36
3%	100.00	92.48	71.62	59.66	97.16	90.78	93.99	39.59
5%	100.00	93.70	81.66	58.75	98.14	90.81	99.99	34.49
8%	99.99	93.46	89.81	58.01	99.42	89.24	100.00	40.37
10%	99.99	93.36	89.53	57.63	99.36	90.41	99.99	37.86

Impact of K. We summarize the results of ASR and ACC of our proposed attack under different K in Table 5. We observe that ASR first increases when K increases. When K exceeds a certain threshold, e.g., 50, ASR decreases slightly when K increases. We anticipate that when K is small, a larger K can result in a higher probability of selecting a better trigger, therefore the increase of ASR. However, when K is large enough, a larger K cannot further increase the probability of getting a better trigger. Instead, a larger K will lead to a more diverse Ω, hence the decrease of ASR. The results suggest that a small Ω, e.g., $K = 50$, is sufficient for maintaining `OpenTrigger` low-cost and effective.

Table 5. ACC and ASR of `OpenTrigger` with different K

K	Dataset							
	CIFAR-10		CIFAR-100		GTSRB		TinyImageNet	
	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
20	99.96	93.88	67.40	59.33	91.26	90.43	97.70	39.31
50	100.00	93.70	81.66	58.75	98.14	90.81	99.99	34.49
100	99.99	93.62	81.16	59.08	99.08	90.03	100.00	41.44
200	100.00	93.95	78.41	59.00	99.49	89.41	97.62	38.05

Impact of Blend Rate α. Intuitively, the higher α, the more trigger portion in a poisoned sample, therefore the higher ASR a backdoor attack can achieve. We summarize the results of ASR and ACC of our proposed attack under different α on CIFAR-10 and CIFAR-100 in Fig. 5, where we vary α of training and test phase between 0.1 and 0.4. The figure shows that ASR of our attack tends to increase when α increases, which is as expected. The results also suggest that the optimal α could depend on the specific dataset under attack. Nonetheless, α between 0.2 and 0.3 serves as proper choices for practical backdoor attacks.

6 Conclusion

We developed `OpenTrigger`, a flexible and robust backdoor attack on DNNs that utilizes dynamic triggers instead of a fixed trigger. Our attack is shown to be robust against existing backdoor defenses including recent adaptive ones. As a result, our attack allows that the attacker uses unseen triggers to activate the planted backdoor, which most existing attacks cannot. Extensive experiments and performance comparison confirm the better performance of `OpenTrigger`.

Acknowledgments. This paper is supported in part by the US National Science Foundation under grants CNS-2422863 and CMMI-2401555 and the Office of Naval Research subawarded through Virginia Tech under Grant N00014-24-1-2730. We would like to thank anonymous reviewers for their constructive comments and helpful advice.

References

1. Wu, B., et al.: Backdoorbench: a comprehensive benchmark of backdoor learning. In: Proceedings of the 36th International Conference on Neural Information Processing Systems, NIPS 2022, Curran Associates Inc., Red Hook, NY, USA (2022)
2. Gu, T., Liu, K., Dolan-Gavitt, B., Garg, S.: Badnets: evaluating backdooring attacks on deep neural networks. IEEE Access **7**, 47230–47244 (2019). https://doi.org/10.1109/access.2019.2909068
3. Zeng, Y., Park, W., Mao, Z.M., Jia, R.: Rethinking the backdoor attacks' triggers: a frequency perspective. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV). IEEE (Oct 2021). https://doi.org/10.1109/iccv48922.2021.01616
4. Wang, T., Yao, Y., Xu, F., An, S., Tong, H., Wang, T.: An Invisible Black-Box Backdoor Attack Through Frequency Domain, pp. 396-413. Springer Nature Switzerland (2022). https://doi.org/10.1007/978-3-031-19778-9_23
5. Li, C., Pang, R., Xi, Z., Du, T., Ji, S., Yao, Y., Wang, T.: An embarrassingly simple backdoor attack on self-supervised learning. In: 2023 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 4344-4355. IEEE (Oct 2023).https://doi.org/10.1109/iccv51070.2023.00403
6. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: visual explanations from deep networks via gradient-based localization. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 618-626. IEEE (Oct 2017). https://doi.org/10.1109/iccv.2017.74
7. Chen, X., Liu, C., Li, B., Lu, K., Song, D.: Targeted backdoor attacks on deep learning systems using data poisoning (2017). https://arxiv.org/abs/1712.05526
8. Turner, A., Tsipras, D., Madry, A.: Label-consistent backdoor attacks (2019). https://arxiv.org/abs/1912.02771
9. Singha, A., Bi, Z., Li, T., Chen, Y., Zhang, Y.: Securing contrastive mmwave-based human activity recognition against adversarial label flipping. In: Proceedings of the 17th ACM Conference on Security and Privacy in Wireless and Mobile Networks, WiSec 2024, pp. 31-41. , ACM (May 2024). https://doi.org/10.1145/3643833.3656123
10. Guo, H., Chen, X., Guo, J., Xiao, L., Yan, Q.: Masterkey: practical backdoor attack against speaker verification systems. In: Proceedings of the 29th Annual International Conference on Mobile Computing and Networking, ACM MobiCom 2023. pp. 1-15. ACM (Oct 2023). https://doi.org/10.1145/3570361.3613261
11. Nguyen, T.A., Tran, T.A.: Input-aware dynamic backdoor attack. In: Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS 2020, Curran Associates Inc., Red Hook, NY, USA (2020)
12. Hayase, J., Kong, W., Somani, R., Oh, S.: Spectre: defending against backdoor attacks using robust statistics. In: Meila, M., Zhang, T. (eds.) Proceedings of the 38th International Conference on Machine Learning. Proceedings of Machine Learning Research, 18–24 Jul, vol. 139, pp. 4129–4139. PMLR (2021). https://proceedings.mlr.press/v139/hayase21a.html
13. Ma, W., Wang, D., Sun, R., Xue, M., Wen, S., Xiang, Y.: The "beatrix" resurrections: robust backdoor detection via gram matrices. In: 30th Annual Network and Distributed System Security Symposium, NDSS (2023)
14. Pan, M., Zeng, Y., Lyu, L., Lin, X., Jia, R.: ASSET: robust backdoor data detection across a multiplicity of deep learning paradigms. In: 32nd USENIX Security Symposium (USENIX Security 23), pp. 2725–2742. USENIX Association, Anaheim, CA (Aug 2023). https://www.usenix.org/conference/usenixsecurity23/presentation/pan

15. Wang, N., Xiao, Y., Chen, Y., Hu, Y., Lou, W., Hou, Y.T.: Flare: defending federated learning against model poisoning attacks via latent space representations. In: Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security, ASIA CCS 2022, pp. 946-958. ACM (May 2022). https://doi.org/10.1145/3488932.3517395
16. Li, Y., Lyu, X., Koren, N., Lyu, L., Li, B., Ma, X.: Neural attention distillation: erasing backdoor triggers from deep neural networks. In: International Conference on Learning Representations (2021). https://openreview.net/forum?id=9l0K4OM-oXE
17. Zheng, R., Tang, R., Li, J., Liu, L.: Pre-activation distributions expose backdoor neurons. In: Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A. (eds.) Advances in Neural Information Processing Systems, vol. 35, pp. 18667–18680. Curran Associates, Inc. (2022)
18. Gao, K., Bai, Y., Gu, J., Yang, Y., Xia, S.T.: Backdoor defense via adaptively splitting poisoned dataset. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE (Jun 2023). https://doi.org/10.1109/cvpr52729.2023.00390
19. Guo, J., Li, Y., Chen, X., Guo, H., Sun, L., Liu, C.: SCALE-UP: an efficient black-box input-level backdoor detection via analyzing scaled prediction consistency. In: The Eleventh International Conference on Learning Representations (2023). https://openreview.net/forum?id=o0LFPcoFKnr
20. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. In: MHS 1995. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, MHS 1995, pp. 39-43. IEEE. https://doi.org/10.1109/mhs.1995.494215
21. Nguyen, T.A., Tran, A.T.: Wanet - imperceptible warping-based backdoor attack. In: International Conference on Learning Representations (2021). https://openreview.net/forum?id=eEn8KTtJOx
22. Wang, Z., Zhai, J., Ma, S.: Bppattack: stealthy and efficient trojan attacks against deep neural networks via image quantization and contrastive adversarial learning. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 15054-15063. IEEE (Jun 2022). https://doi.org/10.1109/cvpr52688.2022.01465