Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2024.051633

ARTICLE

A Gaussian Noise-Based Algorithm for Enhancing Backdoor Attacks

Hong Huang, Yunfei Wang*, Guotao Yuan and Xin Li

School of Computer Science and Engineering, Sichuan University of Science & Engineering, Yibin, 644000, China

*Corresponding Author: Yunfei Wang. Email: 323085406124@stu.suse.edu.cn

Received: 11 March 2024 Accepted: 22 May 2024 Published: 18 July 2024

ABSTRACT

Deep Neural Networks (DNNs) are integral to various aspects of modern life, enhancing work efficiency. Nonetheless, their susceptibility to diverse attack methods, including backdoor attacks, raises security concerns. We aim to investigate backdoor attack methods for image categorization tasks, to promote the development of DNN towards higher security. Research on backdoor attacks currently faces significant challenges due to the distinct and abnormal data patterns of malicious samples, and the meticulous data screening by developers, hindering practical attack implementation. To overcome these challenges, this study proposes a Gaussian Noise-Targeted Universal Adversarial Perturbation (GN-TUAP) algorithm. This approach restricts the direction of perturbations and normalizes abnormal pixel values, ensuring that perturbations progress as much as possible in a direction perpendicular to the decision hyperplane in linear problems. This limits anomalies within the perturbations improves their visual stealthiness, and makes them more challenging for defense methods to detect. To verify the effectiveness, stealthiness, and robustness of GN-TUAP, we proposed a comprehensive threat model. Based on this model, extensive experiments were conducted using the CIFAR-10, CIFAR-100, GTSRB, and MNIST datasets, comparing our method with existing state-of-the-art attack methods. We also tested our perturbation triggers using various defense methods and further experimented on the robustness of the triggers against noise filtering techniques. The experimental outcomes demonstrate that backdoor attacks leveraging perturbations generated via our algorithm exhibit cross-model attack effectiveness and superior stealthiness. Furthermore, they possess robust anti-detection capabilities and maintain commendable performance when subjected to noise-filtering methods.

KEYWORDS

Image classification model; backdoor attack; gaussian distribution; Artificial Intelligence (AI) security

1 Introduction

In recent years, with the rise of deep learning, the Deep Neural Network (DNN) model has achieved remarkable results in classification tasks, including traffic signal recognition [1–3], disease analysis [4,5], face verification [6,7] and other application scenarios. However, these scenarios are facing the threat of backdoor attacks [8–10]. The backdoor attack is passing malicious samples with embedded backdoor triggers through the training model via data poisoning, causing the model to misclassify samples with the triggers into predetermined target classes, while still maintaining high accuracy on clean samples. Backdoor attacks are complex and deceptive techniques, with attackers often refining and evolving their methods to evade detection and enhance attack effectiveness. Given their serious security implications, it is necessary to conduct research on backdoor attacks to advance DNN development towards greater security.

Many advanced backdoor attack methods have been proposed, but updates in defense detection methods, model architectures, and other related technologies have posed new challenges for backdoor attack research. Firstly, in typical backdoor attacks, attackers usually employ patch-based backdoor patterns as triggers [11,12] but do not adequately consider data distribution and the requirements of different application scenarios. This leads to abnormal data distributions, causing abnormal activations during the inference process of deep neural networks. These abnormal activations can be easily detected by advanced detection techniques, rendering such methods ineffective. Tan et al.'s research [13] attempts to bypass detection but relies on complete control over the model's training process, limiting its practical applicability. Zhong et al. [14] proposed using Universal Adversarial Perturbations (UAP) to design triggers. By generating subtle perturbations through iterative adversarial attacks, they created a universal trigger that is difficult to detect and applicable across various scenarios. Building on this, Zhang et al. [15] introduced Targeted Universal Adversarial Perturbations (TUAP) for targeted backdoor attacks. The authors also verified that the trigger model based on universal adversarial perturbations makes the defense models proposed by Tran et al. [16] and Chen et al. [17] more challenging to detect. However, the perturbation magnitude was relatively high, and the stability across different sample categories was poor. Our research found that TUAP did not consider directionality when generating perturbations, leading to adversarial instability [18]. Consequently, the generated triggers exhibited high randomness and were easily recognizable by the human eye.

To overcome the challenges in backdoor attack research, we propose a new method called Gaussian Noise-Targeted Universal Adversarial Perturbation (GN-TUAP). GN-TUAP introduces Gaussian noise to constrain the direction and distribution of perturbations, thereby enhancing the effectiveness and stealthiness of backdoor attacks while increasing robustness against defense detection methods. The introduction of Gaussian noise brings several advantages. Gaussian noise satisfies the properties of Gaussian and uniform distributions, allowing us to optimize perturbations at the pixel level. Specifically, we enhance the effectiveness of the perturbations by directing them towards the optimal direction and improve their stability by uniformly handling anomalous pixels. This addresses the issue of anomalies being easily detected by defense methods. Our method simplifies the misclassification problem into a linear one—finding the shortest distance from the sample's origin to the hyperplane. The introduction of Gaussian noise enables perturbations to move as vertically as possible from the origin towards the hyperplane, thereby improving the attack's effectiveness. GN-TUAP shows significant improvements in effectiveness and stability compared to traditional TUAP methods. It not only generates high-quality perturbations but also effectively reduces the likelihood of being detected by defense methods. To demonstrate the general effectiveness, stealthiness, and resistance to defense detection, we proposed a more comprehensive threat model. In this threat model, we conduct backdoor attacks based on minimal oracle conditions, meaning our attack method only poisons a certain proportion of training samples in the dataset and does not rely on other data. Based on this stringent threat model, we conducted comparative experiments with several advanced backdoor attack methods, including Individual Sample Specific Backdoor Attack (ISSBA) [19], bit-per-pixel (BPP) [20], Subnet Replacement Attack (SRA) [9], and Adaptive Backdoor Attack [21]. Additionally, because our trigger includes Gaussian noise, which imparts certain noise characteristics, we conducted further experiments to test its resistance to noise filtering methods. Both theoretical explanations and experimental results demonstrate the robustness of the triggers generated by the GN-TUAP method.

In our study, the primary focuses include the following aspects: (1) How to achieve backdoor attacks while reducing assumptions. Current backdoor attacks often assume complete control by the attacker over the model, a condition challenging to meet in practical applications; (2) How to evade detection by advanced defense models. With the increasing volume of data, the complexity of Deep Neural Network models has risen, demanding substantial computational resources. Small and medium-sized enterprises, as well as individual users, often cannot afford the high training costs, outsourcing deep neural network training to third parties [22]. Due to the opacity of the training process, administrators often inspect models for potential malicious backdoors during acceptance testing; (3) Concerning the generation of backdoor triggers, we focus on how to effectively reduce perturbation and stably generate effective triggers. When inserting samples with triggers, it is necessary to consider the concealment of toxic samples, ensuring that they can evade both model detection and human visual recognition systems.

To achieve the objectives mentioned above, we have introduced a Gaussian Noise-Targeted Universal Adversarial Perturbation algorithm. This algorithm serves as a method for generating backdoor triggers. It optimizes the direction of perturbation by introducing Gaussian noise in each iteration, resulting in the creation of more stable and covert backdoor triggers. Additionally, we have proposed a threat model scenario to validate the effectiveness of our approach. The specific works include the following aspects:

(1) We have introduced an adversarial perturbation generation algorithm that adheres to a Gaussian distribution, aiming to create more stable and concealed triggers, thereby enhancing the feasibility of the attack.

(2) We have implemented cross-model attacks under different model structures, demonstrating that attackers can execute backdoor attacks even when they lack control over the training process, relying solely on access to the training dataset. This reduces the prerequisite knowledge required for backdoor attacks.

(3) We have designed a threat scenario that considers different roles with varying levels of permissions. This design closely mirrors real-world situations, providing substantial insights into the feasibility of attacks.

The rest of this paper is organized as follows. In Section 2, we will introduce relevant theoretical knowledge and discuss in detail the feasibility from three perspectives: the attacker, the attack scenario, and the attack target within the threat model. In Section 3, we will focus on the generation and injection of GN-TUAP. Then, in Section 4, we will present the experimental results and conduct a comprehensive analysis. Finally, in Section 5, we will provide conclusions and suggestions for future work.

2 Preliminaries

2.1 Definitions and Notions

For better understanding of the backdoor triggers generated via adversarial techniques, the following definitions and notions are introduced.

Adversarial Attack: Suppose we have a prediction function f and a clean sample x. The goal of adversarial attack is to find a perturbation v such that $f(x+v) \neq f(x)$. The sample $x+v$ is referred to as an adversarial sample. Typically, v is a very small perturbation that is imperceptible to the human eye.

Backdoor Attack: $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ represents a clean training dataset, where $(x_1, x_2, \ldots, x_n) \in X$ represents clean samples, and $(y_1, y_2, \ldots, y_n) \in Y$ represents the source label. To conduct a backdoor attack, during the toxic dataset generation phase, the attacker first selects a small subset of clean samples $D_C = \{(x_i, y_i) | x_i \in X, y_i \in Y\}$, among which $D_C \in D$, then adds a trigger to the clean samples x, and modifies the corresponding y values, ultimately obtaining toxic samples $(\tilde{x}, \tilde{y})$. These toxic samples are then inserted into the clean dataset, transforming it into a toxic dataset D_P. The generation of toxic samples is illustrated in formula (1).

$$D_P = \{(\widetilde{x_1}, \widetilde{y_1}), (\widetilde{x_2}, \widetilde{y_2}), \ldots, (\widetilde{x_i}, \widetilde{y_i})\} \tag{1}$$

In general, the toxic samples $\tilde{x}$ after triggers addition and the clean samples x_i will be very similar, making it difficult for the human visual recognition system to detect the difference. This achieves the effect of being covert. During the training phase, the attacker uses the generated toxic dataset to train the clean model f_c, resulting in a poisoned model f_p. The stealthiness of the backdoor attack lies in the fact that even after poisoning the model, it will not affect the normal predictions of the clean model, as illustrated in formula (2). When the clean sample x is input to the model, it will be correctly classified as y. However, when a sample $\tilde{x}$ carrying the trigger is input to the model, the poisoned model will result in a misclassification $\tilde{y}$.

$$f_p(x) = y, f_p(\tilde{x}) = \tilde{y} \tag{2}$$

Gaussian Noise: Noise is generally characterized by its frequency characteristics, and ideal noise is referred to as white noise. When the noise has amplitude distribution following a Gaussian distribution and its power spectral density follows a uniform distribution, it is called Gaussian white noise.

Firstly, when white noise follows a Gaussian distribution, if a random variable X follows a normal distribution with a mean of μ and a variance of σ^2, it is denoted as $N(\mu, \sigma^2)$. The probability density function is shown in formula (3). The mean μ determines the center position, and the standard deviation σ determines the amplitude of the distribution.

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{3}$$

2.2 *Threat Model Scenario Design*

We have designed a reasonable threat model and conducted attack-defense adversarial experiments within it. The structure of this model is depicted in Fig. 1, which clearly illustrates the potential attack-defense scenarios in real-world settings and provides a comprehensive explanation, ensuring the feasibility of backdoor attacks.

Attacker: In the threat model, it is assumed that the attacker can manipulate the training data, but not the training process. For example, the attacker could be those who have access to the training data storage or the provider of the training data. Due to the large training dataset used by the learning model during the training phase, it is impractical for developers to manually check the security of the dataset, especially when the dataset is collected from multiple untrusted sources. Therefore, it is possible that the dataset was tainted before it was used, but the developer would not have noticed.

Attack Scenario: Attackers utilize training data and a general perturbation generation algorithm to create a backdoor trigger, embedding it within training samples to compromise the dataset. During this process, attackers may or may not possess knowledge of the target model's internal structure. If known, attackers can generate a more effective trigger using it as a reference model. Otherwise, they conduct cross-model attacks, leveraging available models to generate triggers. Both methods have proven effective. Typically, developers utilize public datasets from the internet due to cost constraints. Attackers exploit file upload vulnerabilities to substitute clean datasets with poisoned ones [23]. Consequently, poisoned data is used for model training, resulting in the backdoor lurking within the model. Post-verification, the trained model is deployed to the production environment. Upon inputting samples with triggers, the model activates abnormally, yielding abnormal outputs.

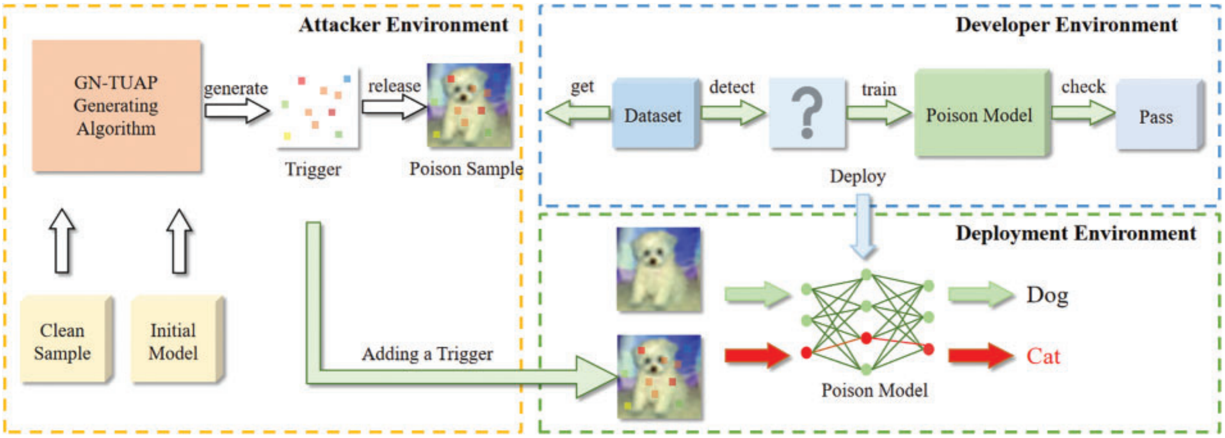

Figure 1: This figure demonstrates the capabilities of both attackers and developers in different scenarios

Attack Target: The primary aim of our adversarial perturbation backdoor attack is to inject a backdoor into a DNN model without compromising its accuracy on clean data. Once deployed, an attacker inputs trigger-laden samples, prompting the model to produce predetermined outputs aligned with the attacker's goals. Maintaining predictive accuracy on clean data is crucial for deployment; otherwise, developers may reject the model post-validation. Additionally, evading detection by backdoor detection tools is paramount. If detected, too much removal of poisoned data jeopardizes the attack's success.

Limitation: Although we have reasonably assumed such an attack-defense environment, we have also identified certain limitations in our approach. One of the shortcomings is that our algorithm heavily relies on the original dataset and model structure. If we cannot reliably access these two components, the generated trigger may exhibit low attack effectiveness. Additionally, different backdoor triggers often have corresponding detection methods. Up to now, there is no trigger that can completely evade all detection techniques, and our trigger is no exception.

3 Generate Trigger and Poison Training

The GN-TUAP backdoor attack is mainly based on data poisoning and the attack scheme consists of two parts. An adversarial attack algorithm is used to generate perturbations in the perturbation generation part. Gaussian noise is added in each iteration to optimize the generated perturbations, and GN-TUAP is generated after N iterations. In the data poisoning part, GN-TUAP is embedded into clean samples and the corresponding labels are changed to construct poison samples. Once the poison samples are used for model training, it will lead to backdoor attacks.

3.1 Generation Triggers

3.1.1 Adversarial Perturbation Based on Gaussian Distribution

From the research work of Moosavi-Dezfooli et al. [24], we can know that the perturbation has any direction, if not controlled, it will cause perturbation instability. As shown in Fig. 2, we first make a circle tangent to $H(x)$ with (x_0, y_0) as the center, and then make a triangle intersecting the circle with $H(x)$ as the hypotenuse, then any point on the circle is the smallest perturbation from (x_0, y_0) to the hyperplane $H(x)$, and the direction of orthogonal projection is the smallest perturbation, then the shaded part S is the additional perturbation to the decision hyperplane. From a probabilistic point of view, the phenomenon that every perturbation is the smallest perturbation is difficult to exist in nature, which leads to the instability of the perturbation. We find that the larger the area of S is, the more directions the perturbations have, while (x_0, y_0) and $H(x)$ are fixed, and we can only control the instability of the perturbations by decreasing the area of S. In order to make the perturbation direction more stable, we use to add the noise that satisfies the Gaussian distribution to stimulate the perturbation direction. In Fig. 3, we can see that μ represents the mean value, which controls the symmetry axis in the image, and σ^2 represents the variance, which controls the magnitude of the image. We take the point (x_0, y_0) as the mean so that the perturbation direction is centered on the perturbation, and then control the magnitude of the variance to make the perturbation direction as much as possible in the direction of the orthogonal projection, so as to make the generated perturbations reach the optimal value with maximum probability, and each time we generate different perturbations, we can make each generated perturbation reach the optimal value with maximum probability. The Gaussian noise we mentioned earlier not only meets the Gaussian distribution of amplitude, but also meets the uniform distribution of power spectrum, which means that our noise will also be more random in the sample distribution points, which has the advantage of reducing the abnormal distribution of samples and making the generated triggers more hidden. As shown in formula (4).

$$v_i = P(x_i, F_G, l_t) + GN(x_i, F_G(x_i)) \tag{4}$$

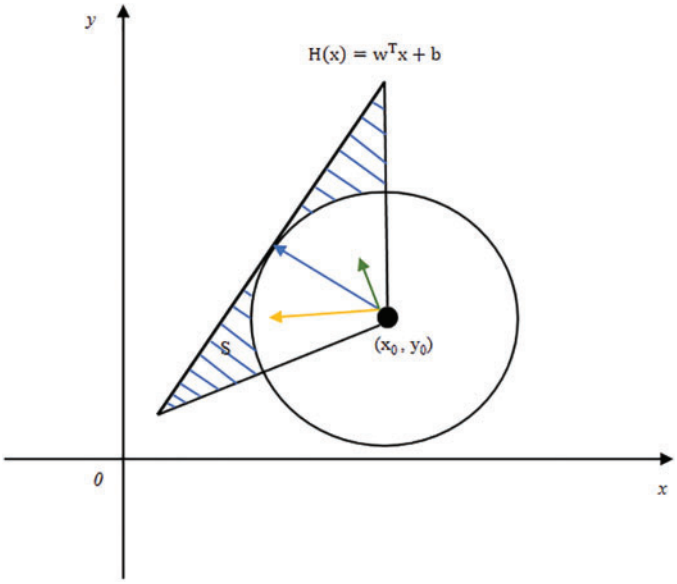

Figure 2: The presence of multiple different directions when moving from decision points to the decision plane

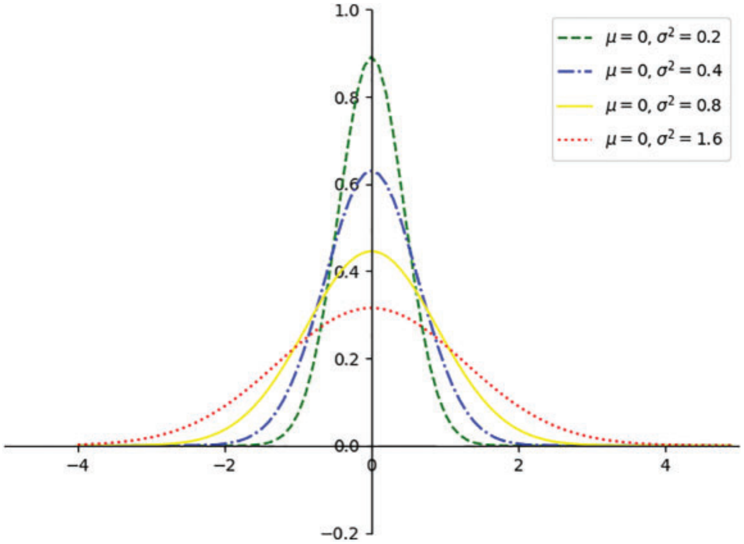

Figure 3: This figure illustrates that variance determines the steepness of the curve

The formula: $P(x_i, F_G, l_t)$ is to use the adversarial attack algorithm to generate the adversarial perturbation of the sample, and $GN(x_i, F_G(x_i))$ is calculating the Gaussian noise of $F_G(x_i)$ at the i-th sample.

3.1.2 Algorithm for Generating Adversarial Perturbations Based on Gaussian Distribution

The goal of the GN-TUAP algorithm is to find a universal perturbation. First, select the sample to enter the loop, then select an attack target l_t, use the method proposed in this paper to iterate with the adversarial attack algorithm, continuously superimpose the perturbations of the samples, and finally generate the final Perturbation V. Specifically our algorithm first selects a sample to enter the loop and chooses the target to attack based on the attack target class. Then iterate using adversarial generation algorithms and DNN (Deep Neural Networks) models to generate perturbations and superimpose them on the original sample until the maximum number of iterations is satisfied or a specified fooling rate threshold is reached. Finally, output the generated universal perturbation V, the details of the algorithm are shown in Algorithm 1.

Algorithm 1: GN-TUAP algorithm

```
Input: $Xs$ indicates the original target category, $xi$ indicates the $i$-th sample, $vi$ indicates the perturbation after the $i$-th iteration, $lt$ indicates the attack target category, $\delta$ indicates the threshold of the fooling rate, $l$ indicates the maximum number of iterations, $FG$ indicates the DNN model that generates the perturbation, $r$ indicates the radius of the projected sphere, and $P$ indicates the confrontation generation algorithm.
Output: $V$ represents the generated GN-TUAP.
$k = 0$
$p = \Sigma x \in Xs[FG\ (x) = lt]$
$V = 0$
while $k < l$ and $p < (1 - \delta) \times |Xs|$ do
    $i = 0$
```

(Continued)

Algorithm 1 (continued)

```
6.    while $i < n$ do
7.        if $FG\ (xi) \neq lt$ then
8.            $vi\ = P\ (xi, FG, lt) + \mathrm{GN}\ (xi, FG\ (xi))$
9.        $V\ = V\ + vi$
10.       $Vsign = sign(V)$
11.       $Vmin = minimum\ (|V|, r)$
12.       $V\ = Vsign \times Vmin$
13.       $i\ = i\ + 1$
14.   $k\ = k\ + 1$
15.   $p\ = \Sigma x \in Xs\ [FG\ (x\ + V) = lt]$
16.   Return $V$
```

The k and P in the outer loop are to ensure that the loop can exit normally, and in the inner loop P is an adversarial attack algorithm. This algorithm uses a Deep Fool attack. Line 8, P can be selected as arbitrary adversarial attack according to different situations, the input of an adversarial attack usually includes clean samples, an adversarial attack algorithm, attack target, GN is Gaussian noise, using the size of mean and variance (mean is 0, variance is 0.01) to generate a small perturbation to disturb the perturbation The function of the direction, the input is the sample and the adversarial perturbation; in line 9, each iteration superimposed to calculate the perturbation of the sample, and then the perturbation of each sample will be superimposed together by the algorithm, and finally generate a V; lines 10–13 optimize the size of the overall perturbation by limiting the overall perturbation; lines 14–15 count the number of samples to ensure that the loop can exit normally; finally line 16 will return a perturbation that contains all v_i and has been concealed. The returned perturbation has the same pixel size as the input sample x. GN-TUAP is generated using a clean dataset and a clean model. The clean data set is the data set that needs to be polluted in the backdoor attack; the clean model is the target model to be attacked. This model can choose the same model structure as the target or a similar model structure. The poisoning attack does not need to be too strong on the ability to master the model. The algorithm pays more attention to how these generated poisonous samples can escape the detection of the defense model. If the generated poisonous samples cannot escape detection, then no matter how advanced the poisonous samples are, they still cannot attack successfully.

3.2 Data Poisoning Training

During the data poisoning training process, some toxic samples with backdoor triggers will be created first, which will make it difficult for the deep neural network model to learn the features of the samples, resulting in a very strong dependence of the model on the triggers, and thus the attacker can manipulate the output of the model to get the desired results, and then add these toxic samples to the normal dataset for training; eventually, the model will learn backdoor features in the poisoned samples. In this paper, the process of injecting the backdoor into the model by generating perturbations using the GN-TUAP method proposed in Section 3.1 is shown in Fig. 4. First, some input samples are randomly selected from the clean dataset, the triggers generated by the GN-TUAP algorithm were then added to these samples. These samples after adding triggers are called poisoned samples. Second, the labels of these poisoned samples are changed to the label of the attack targets, and the poisoned samples labeled with the attack target label are mixed into the clean dataset, thus constructing the poisoned dataset; finally, the model with the backdoor is trained using these poisoned datasets.

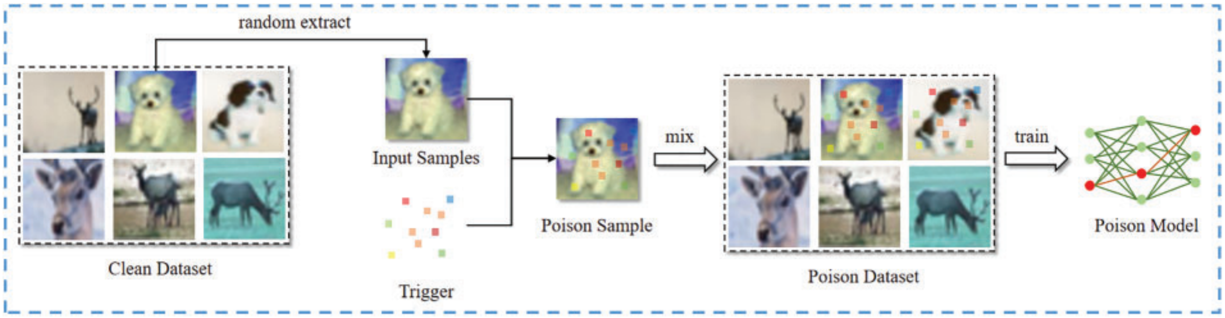

Figure 4: The process of data poisoning training in a backdoor attack

4 Experiment and Result Analysis

4.1 Evaluation Index

Considering that this paper uses the gray box model to simulate the backdoor attack in the real environment, combined with the threat model scenario proposed in Section 2.2, the proposed backdoor attack method is to generate triggers and make poison samples through other models, and then use data The poisoning method focuses on evading the detection of developers and backdoor defenses so that poisonous data sets can participate in model training to poison the model and achieve the purpose of the attack. Therefore, the backdoor attack is mainly evaluated from four aspects: effectiveness, concealment, adversarial detection, noise filtration resistance performance, as follows:

(1) Effectiveness: The effectiveness of deep neural network backdoor attacks can be measured by the Attack Success Rate (ASR). The model classifies any sample with a backdoor trigger as the target class to define the ASR, such as formula (5), which means that the value is 1 when both sides of the equal sign are equal, and the value is 0 when they are not equal.

$$ASR = \frac{1}{N}\sum\nolimits_{i=1}^{N}\prod(f(\tilde{x}) = \tilde{y}) \tag{5}$$

(2) Stealthiness: The concealment of the deep neural network backdoor attack is to measure the performance difference between the deep neural network implanted with the backdoor and the original network in the face of benign samples, that is, the difference between the embedded trigger and the clean sample can be avoided. Developer detection. And when the model with the backdoor is faced with clean input samples, the accuracy rate is still similar to that of the clean model.

(3) Adversarial detection: The adversarial detection of backdoor attacks generally refers to whether the trigger embedded in the sample is easy to be detected by manual recognition or defense algorithms. Defense detection uses the detection method of activation clustering to detect whether there is a backdoor. In the backdoor detection, three indicators of Precision, Recall, and F1 value are used to evaluate the backdoor situation.

$$Precision = \frac{TP}{TP + FP} \times 100\% \tag{6}$$

$$Recall = \frac{TP}{TP + FN} \times 100\% \tag{7}$$

$$F1 = \frac{2Precision * Recall}{Precision + Recall} \times 100\% \tag{8}$$

In the formula: *TP* is the positive class is judged as the positive class, *FP* is the negative class is judged as the positive class, *TN* is the negative class is judged as the negative class, *FN* is the positive class is judged as the negative class.

(4) Noise filtration resistance performance: The triggers generated by GN-TUAP have some characteristics of noise. To explore the impact of noise filtering methods on the triggers, we designed experiments to evaluate the resistance performance of the triggers to noise filtering. By applying noise filtering to the entire dataset, we simulated the scenario in which developers denoise the dataset without knowing which samples are toxic.

4.2 Effectiveness Evaluation Results

In order to verify the attack effectiveness of the GN-TUAP algorithm for generating perturbations, we first conduct experiments based on the CIFAR-10 dataset, and we will conduct comparative experiments on the attack success rate with the TUAP method, BPP method and the Adaptive method on the ResNet and the VGGNet models. Table 1 shows the specific structure of the victim model we use.

Table 1: Schematic table of ResNet18 and VGGNet16 model structures

Layer name	Output size	ResNet18	VGGNet16	
Conv1	112 × 112	7 × 7, 64, stride 2	Conv1	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$
Conv2_x	56 × 56	3 × 3 max pool, stride 2		Pool
		$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	Conv2	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$
Conv3x	28 × 28	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$		Pool
Conv4x	14 × 14	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$	Conv3	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$
Conv5x	7 × 7	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 2$		Pool
	1 × 1	Average pool, 1000-d fc, softmax	Conv4	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$ Pool

(Continued)

Table 1 (continued)

Layer name	Output size	ResNet18	VGGNet16
FLOPs		1.8×10^9	Conv5 $\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$ Pool Fc 4096 Fc 4096 Fc 1000 softmax

The BPP backdoor attack method [20] is image color quantization and dithering are used to generate triggers as well as poisoned samples, which are injected using comparative learning and adversarial training methods, while the Adaptive method [21] is an adaptive backdoor attack method proposed by Qi et al. The method categorizes toxic samples into two groups: one with normal labels and the other with target labels. During training, it employs various triggers with some weakening. To maintain fairness despite different attack methods, we set a uniform target label and a 30% poisoning rate. We conducted one-to-one and many-to-one labeled attacks. In one-to-one attacks, only samples of a specific label type were poisoned, while in many-to-one attacks, all samples except the target label were poisoned. Tables 2 and 3 present cross-model attack success rates for each backdoor method in these tasks, with "Begin Models" indicating victim model types. Table 4 illustrates the change in clean sample accuracy pre- and post-attack on the victim model. Notably, the Adaptive method struggles with one-to-one attacks due to its constrained poisoning sample range, resulting in maladaptation despite several attempts.

Table 2: The ASR of each backdoor attack method when performing many-to-one cross-model attacks

Begin models	GN-TUAP	TUAP (2021)	BPP (2022)	Adaptive (2023)
VGGNet	93.00%	92.72%	96.44%	97.10%
ResNet	92.43%	89.23%	94.17%	95.30%

Table 3: The ASR of each backdoor attack method when performing one-to-one cross-model attacks

Begin models	GN-TUAP	TUAP (2021)	BPP (2022)	Adaptive (2023)
VGGNet	94.67%	93.32%	94.54%	6.25%
ResNet	96.73%	90.44%	88.76%	6.11%

Table 4: The change in accuracy of the victim model on clean samples before and after being attacked

Models	ResNet			VGGNet		
		Many-to-One	One-to-One		Many-to-One	One-to-One
Backdoor attack methods	Initial clean sample accuracy	Poisoned model clean sample accuracy	Poisoned model clean sample accuracy	Initial clean sample accuracy	Poisoned model clean sample accuracy	Poisoned model clean sample accuracy
GN-TUAP	93.95%	91.30%	90.24%	92.89%	88.56%	91.33%
TUAP	93.95%	90.12%	89.37%	92.89%	91.47%	90.26%
BPP	93.95%	88.74%	89.04%	92.89%	88.70%	89.57%
Adaptive	93.95%	87.04%	93.68%	92.89%	89.40%	91.83%

From the data, we can see that our attack is effective, especially under the labeled one-to-one attack task, which is related to our perturbation algorithm, our proposed GN-TUAP algorithm is generated for the labeled one-to-one task, and at the same time, the damage of our attack on the model's performance on the clean samples is acceptable, which means that the attacked model will not be abandoned to be deployed because of the bad effect.

4.3 Stealth Assessment Results

In the stealth evaluation, this paper compares with the TUAP method proposed by Li et al. [11] on the stability of trigger generation and the diversity of poison samples. As shown in Fig. 5, Fig. 5a Noise shows that the triggers generated by the GN-TUAP method are very uniform in the noise distribution point, which is more in line with natural noise, and the triggers generated under different sample categories are also relatively stable. Fig. 5a Add Noise shows the poison samples after adding GN-TUAP, it is difficult to identify the shape with abnormal distribution with human eyes. Fig. 5b Noise shows the triggers generated using TUAP. It can be seen that the noise points of the triggers are uneven, the noise is very unnatural, and the triggers are under different sample categories There is a big difference between the detectors. From Fig. 5b Add Noise, it can be seen that the poison samples after adding TUAP also have the obvious abnormal distribution of data, which is very easy to distinguish even by the human eye.

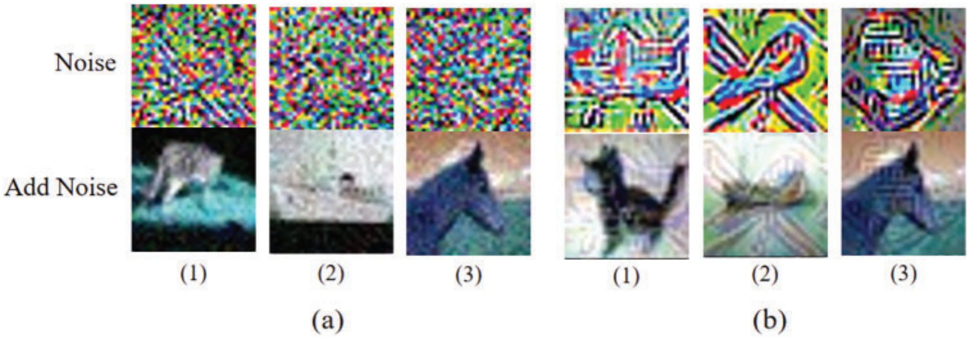

Figure 5: Comparison of the effect of triggers based on the CIFAR-10 dataset

Meanwhile, we conducted a comparison test on the CIFAR-100 dataset and the GTSRB dataset, Figs. 6 and 7 show the comparison of the effect of the triggers generated by GN-TUAP and the triggers generated by TUAP on the two datasets, respectively, and we can see that the triggers generated by the GN-TUAP algorithm will be more natural and more covert.

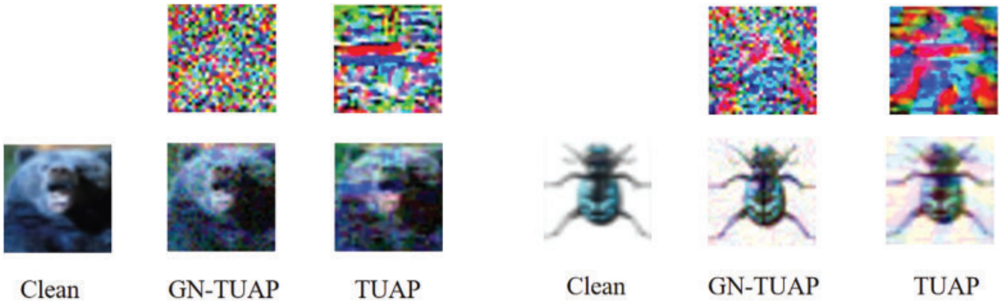

Figure 6: Comparison of the effect of triggers based on the CIFAR-100 dataset

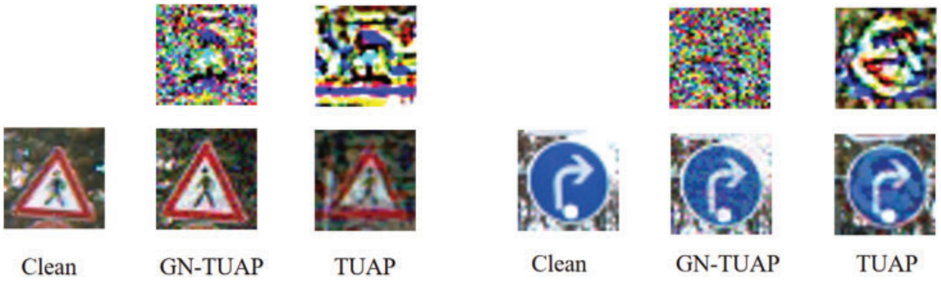

Figure 7: Comparison of the effect of triggers based on the GTSRB dataset

In addition, we also verify the difference between poison and clean samples generated using GN-TUAP on the Fashion MNIST and MNIST datasets, as shown in Fig. 8. In the figure, we compare the toxic samples generated using the GN-TUAP and TUAP algorithms on the FASHION-MNIST and MNIST datasets. Overall, the GN-TUAP method has a stable and hidden trigger generation advantage. Using GN-TUAP cannot only quickly construct a poison sample to quickly deploy an attack plan, but also easily evade screening even if a developer reviews the data set manually.

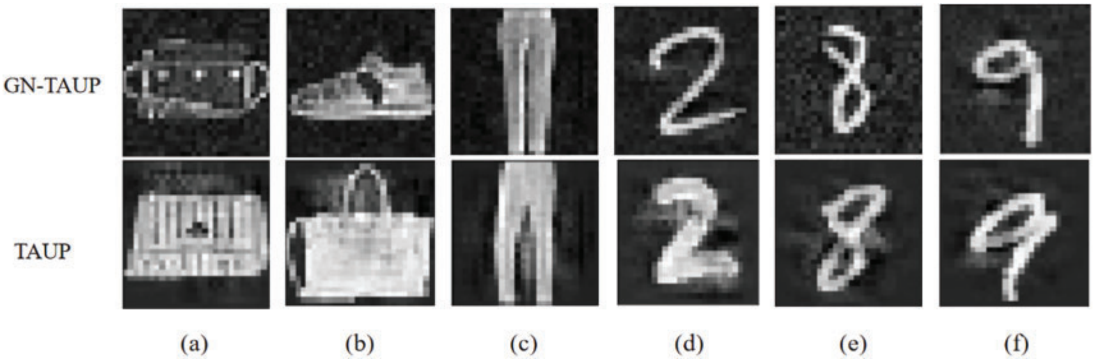

Figure 8: Poison samples (up) after adding GN-TUAP and poison samples (low) after adding TUAP in Fashion MNIST

4.4 Backdoor Defense Detection and Evaluation Results

In the threat model, when developers use unknown data sets, they often pass the detection of the defense model before they can be used. There we adopt the Activation Clustering [1,2,15] method to detect whether a sample contains a backdoor trigger. First, we conducted experiments based on the CIFAR-10 dataset. we will provide all the inputs to the backdoor model, collect their activation values separately, and then use the K-means algorithm to cluster the activation values into two clusters after dimensionality reduction, if the number of activations in a cluster is lower than A certain value, the cluster will be identified as poisoned, once the cluster is determined to be poisoned, the current model will be marked as poisoned, and the corresponding activation data in the poisoned cluster will be deleted. When using the above defense model for detection experiments, 1500 poison samples added with GN-TUAP were input for each category to evaluate its adversarial detection performance. Therefore, when the values of these three indicators are lower, it means that GN-TUAP is more resistant to detection and less likely to be detected. This paper selects 10 pairs of samples to compare with other works and uses Precision, Recall, and F1 to evaluate the detection results. The specific data are shown in Table 5. The table compares the adversarial detection performance of GN-TUAP, TUAP, and patch triggers on the defense model. It can be seen that the triggers based on the patch are almost all detected in the backdoor detection model. This is because the trigger is based on a patch The sensor is not an adversarial perturbation, but a fixed pattern, so it leads to abnormal distribution and cannot escape detection. Using TUAP adversarial perturbation and GN-TUAP proposed in this paper solves this distribution anomaly. The highest detection rate of poison samples generated by using the GN-TUAP method is to identify the Deer category as the Bird category, and the detection indicators Precision, Recall, and F1 are 31.53%, 22.93%, and 26.55%, respectively, and there are three categories in the TUAP method The detection rates of toxic samples are higher than the highest index in GN-TUAP, respectively, Airplane is recognized as Deer, Bird is recognized as Dog, and Deer is recognized as Bird; although six outliers are detected using the GN-TUAP method, the average abnormality The value is relatively low, the average detection rate of the three detection indicators is only 7.4%, and three outliers are detected based on the TUAP method, all of which are marked as backdoor models, with an average detection rate of about 20.53%. GN-TUAP can effectively prevent the detection and removal of poison data, and still, save a large amount of toxic data in the data set. It can be seen in Fig. 9 that only an average of less than 10% of toxic samples will be detected as abnormal, which is about 13% lower than the detection rate of traditional TUAP.

Table 5: Three evaluation indexes of GN-TUAP, TUAP, and patch in detection resistance

Attack category	GN-TUAP			TUAP			Patch		
Category A to Category B	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Horse→Truck	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	99.68%	83.33%	90.78%
Ship→Airplane	2.06%	0.80%	1.15%	0.00%	0.00%	0.00%	100%	85.40%	92.13%
Ship→Frog	16.07%	12.66%	14.16%	0.00%	0.00%	0.00%	100%	85.87%	92.40%
Deer→Bird	31.53%	22.93%	26.55%	57.38%	81.60%	67.38%	99.43%	81.47%	89.56%
Dog→Deer	21.41%	11.53%	14.99%	0.00%	0.00%	0.00%	99.37%	84.73%	91.47%
Airplane→Deer	0.00%	0.00%	0.00%	58.20%	78.47%	67.24%	99.25%	79.93%	88.55%
Horse→Vehicle	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	100%	89.27%	94.33%
Bird→Dog	10.32%	4.00%	5.70%	57.53%	80.73%	67.18%	99.92%	82.47%	90.36%
Cat→Airplane	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	100%	81.86%	90/03%
Cat→Frog	10.60%	7.73%	8.94%	0.00%	0.00%	0.00%	100%	74.27%	85.23%
Average	9.19%	5.88%	7.14%	17.35%	24.09%	20.17%	99.77%	82.80%	90.48%

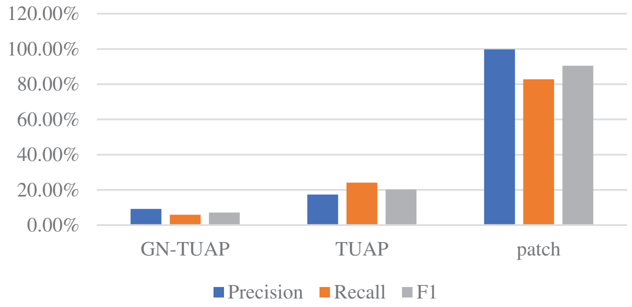

Figure 9: The three evaluation indexes average detection values based on the CIFAR-10 dataset

We also conducts experiments using the CIFAR-100 dataset, where, in addition to employing the aforementioned Activation Clustering method for detection, the Spectral detection method [16] is also utilized. Ten groups are selected from the dataset categories to verify the detection of backdoor attacks based on the GN-TUAP and TUAP algorithms. Tables 6 and 7 display the relevant detection data. From the tables, we can see that when experimenting with the CIFAR-100 dataset, the average detection rate of the GN-TUAP method using the Activation Clustering detection method is significantly lower than that of the TUAP algorithm, with a reduction of approximately 44% in detection rate. When using the Spectral detection method, the GN-TUAP method reduces the detection rate by about 16%. consistent with previous experiments on the CIFAR-10 dataset. Fig. 10 displays the average values of relevant indicators for two algorithms, where "AC" represents the Activation Clustering detection method, and "SP" represents the Spectral detection method.

Table 6: Precision, Recall, and F1-score of two detection methods: Activation Clustering and Spectral on GN-TUAP based on the CIFAR-100 dataset

Attack classes (Source→Target)	GN-TUAP					
	Activation clustering			Spectral		
	Precision	Recall	F1-score	Precision	Recall	F1-score
3→6	0.00%	0.00%	0.00%	2.78%	3.33%	3.03%
3→8	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
5→6	0.00%	0.00%	0.00%	37.50%	46.15%	41.38%
5→8	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
6→9	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
6→21	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
7→11	31.58%	40.00%	35.29%	0.00%	0.00%	0.00%
8→5	100%	60.00%	74.99%	0.00%	0.00%	0.00%
8→6	22.73%	38.46%	28.57%	12.50%	15.38%	13.79%
66→77	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Average	15.43%	13.85%	13.89%	5.28%	6.49%	5.82%

Table 7: Precision, Recall, and F1-score of two detection methods: Activation Clustering and Spectral on TUAP based on the CIFAR-100 dataset

Attack classes (Source→Target)	TUAP					
	Activation clustering			Spectral		
	Precision	Recall	F1-score	Precision	Recall	F1-score
3→6	54.76%	76.67%	63.89%	55.56%	66.67%	60.61%
3→8	100%	40.74%	57.89%	0.00%	0.00%	0.00%
5→6	88.89%	61.53%	72.72%	68.75%	86.62%	75.86%
5→8	100%	71.43%	83.33%	0.00%	0.00%	0.00%
6→9	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
6→21	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
7→11	38.46%	41.67%	40.00%	0.00%	0.00%	0.00%
8→5	90.00%	94.74%	92.30%	0.00%	0.00%	0.00%
8→6	90.90%	76.92%	83.33%	75.00%	92.31%	82.76%
66→77	95.45%	65.62%	77.78%	0.00%	0.00%	0.00%
Average	65.85%	52.93%	57.12%	19.93%	24.56%	21.92%

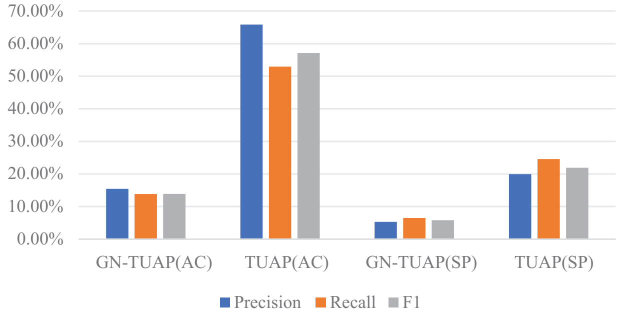

Figure 10: The three evaluation indexes average detection values based on the CIFAR-100 dataset

From the experimental data above, we find that there are some differences in the detection results between different class pairs, which is due to the different gaps between different class pairs. If the two classes are very different, then the perturbations generated by the GN-TUAP algorithm need to be of higher magnitude, which will generate more anomalies in the activation of the poisoning model, and thus be more easily detected. We verified this phenomenon on the GTSRB dataset. The data for defense detection of the GN-TUAP method on the GTSRB dataset is shown in Table 8. We classify the degree of difference between class pairs as Level 1, Level 2, and Level 3 based on three indicators: color, shape, and pattern. If there is only a difference in pattern, we classify it as Level 1. If there are differences in shape or color in addition to pattern, we classify it as Level 2. If there are differences in pattern, color, and shape, we classify it as Level 3. Fig. 11 shows an example of a clean sample used in the table. Analysis of the tabulated data reveals a positive correlation between the level of dissimilarity among class pairs and the probability of detection, with a notable increase in detection accuracy corresponding to higher dissimilarity levels. Notwithstanding, there are outliers; specific class pairs exhibit high visual differentiation yet evade detection. This phenomenon is consistent with our detection data derived from the CIFAR dataset. Despite these exceptions, the overarching trend remains evident: a greater disparity between class pairs generally translates to an elevated likelihood of detection. Of course this does not mean that a large difference will necessarily be detected, just more probable.

Table 8: Experimental datasheet for defense detection based on GTSRB dataset

Variance level	Attack classes (Source→Target)	GN-TUAP					
		Activation clustering			Spectral		
		Precision	Recall	F1-score	Precision	Recall	F1-score
1	21→22	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
1	26→27	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
1	22→27	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
1	33→34	10.63%	7.93%	9.09%	0.00%	0.00%	0.00%
2	5→6	34.82%	55.55%	42.81%	41.89%	49.21%	45.26%
2	5→26	37.68%	41.27%	39.39%	0.00%	0.00%	0.00%
2	34→5	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
2	33→5	31.94%	48.02%	38.36%	0.00%	0.00%	0.00%
3	26→34	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
3	21→6	100%	42.85%	60.00%	35.14%	41.27%	37.96%
3	6→27	56.34%	88.09%	68.73%	3.52%	6.67%	4.60%
3	22→34	48.20%	74.60%	58.57%	0.00%	0.00%	0.00%

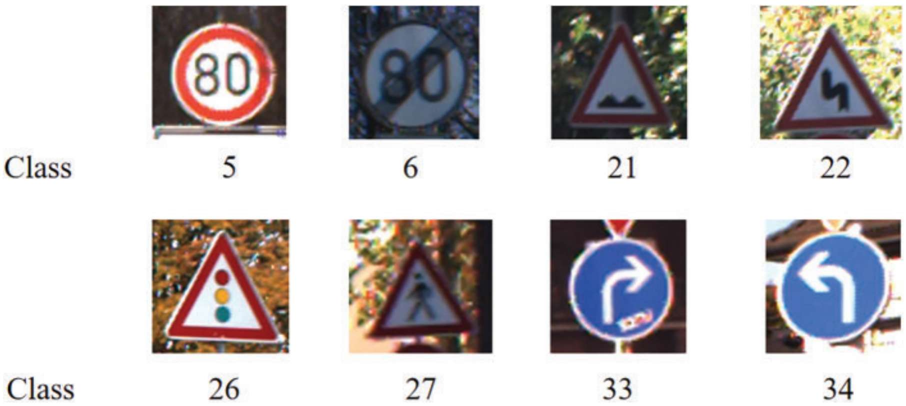

Figure 11: Comparison chart of example categories of the GTSRB dataset

In order to better test our method, we use the ResNet model as the victim model, based on the CIFAR-10 dataset and the GTSRB dataset, and use the Super-Fine-Tuning (SFT) [25], Implicit Bacdoor Adversarial Unlearning (I-BAU) [26], and SCALE-UP [27] defense methods to test our attack method, and do comparison experiments between the detection results and the other backdoor attack methods, in order to facilitate the control and the fairness of the experiment, we still use the setup of the comparison experiment in 4.2. Tables 9 and 10 show the relevant data of the experiments based on the CIFAR-10 dataset, where ASR is the attack success rate after defense, and ACC denotes the accuracy of the model on clean samples after using the defense method.

Table 9: Defense detection datasheet for Many-to-One attacks on the CIFAR-10 dataset

Methods of defense	GN-TUAP (Ours)		BPP (2022)		Adaptive (2023)	
	ASR	ACC	ASR	ACC	ASR	ACC
SFT (2022)	27.54%	54.10%	7.11%	59.76%	1.68%	81.86%
IBAU (2022)	72.48%	27.59%	15.61%	25.48%	4.03%	84.90%
ScaleUp (2023)	58.54%	74.48%	33.34%	75.20%	9.66%	74.38%

Table 10: Defense detection datasheet for One-to-One attacks on the CIFAR-10 dataset

Methods of defense	GN-TUAP (Ours)		BPP (2022)		Adaptive (2023)	
	ASR	ACC	ASR	ACC	ASR	ACC
SFT (2022)	21.06%	51.14%	6.50%	74.45%	2.85%	74.19%
IBAU (2022)	11.01%	26.64%	2.44%	79.02%	1.56%	85.26%
ScaleUp (2023)	49.93%	76.83%	36.40%	75.28%	0.44%	76.03%

From the experimental results based on the CIFAR-10 dataset, it can be seen that our algorithm has excellent performance in defense resistance, in order to further validate the effectiveness and applicability of the method, we conducted experiments based on the GTSRB dataset, we still use the Resnet model as the victim model to conduct experiments using the same experimental setup, and at the same time, we add the ISSBA [19] and SRA [9] backdoor attack method and add another defense detection method called MOTH (Model OrTHogonalization) [28] is used, taking into account that not all methods are adapted to one-to-one attack scenarios, so here we only perform labeled many-to-one attack experiments. Tables 11 and 12 show the data related to the defense detection experiments based on the GTSRB dataset, where ASR is the success rate of the attack after the defense, and ACC denotes the accuracy rate of the model on the clean samples after the use of the defense method.

Table 11: Defense detection datasheet for backdoor attacks on the GTSRB dataset

Methods of defense	GN-TUAP (Ours)		ISSBA (2021)		SRA (2022)		Adaptive (2023)	
	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
None	95.67%	90.68%	81.63%	91.00%	97.92%	95.47%	92.50%	89.26%
SFT (2022)	1.3%	97.41%	0.6197%	98.07%	0.07%	98.78%	0.15%	98.45%
Moth (2022)	97.72%	73.26%	92.68%	63.60%	90.88%	87.14%	68.53%	83.99%
ScaleUp (2023)	47.88%	60.87%	27.16%	58.20%	50.62%	58.15%	15.35%	60.01%

Table 12: The effect of the three noise filtering methods on the ASR metrics and ACC metrics at different denoising strengths

Attack Classes	None		Gaussian		NL_means		Wiener		Wavelet	
	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
0→6	0.9984	0.9343	0.7822	0.3143	0.8735	0.9317	0.8567	0.9082	0.7203	0.9300
0→4	0.9302	0.9389	0.9297	0.2957	0.7152	0.9323	0.8367	0.9205	0.7256	0.9220
1→6	0.9643	0.9363	0.7221	0.2152	0.7918	0.9301	0.8567	0.8336	0.7980	0.9232
3→9	0.9729	0.9370	0.8539	0.2940	0.9523	0.9323	0.8533	0.8298	0.8752	0.9286
2→6	0.9289	0.9363	0.8064	0.5763	0.9103	0.9316	0.9467	0.8113	0.7440	0.9158
3→6	0.9633	0.9207	0.7477	0.3508	0.6741	0.9323	0.8800	0.8181	0.7795	0.9252
4→6	0.9476	0.9443	0.8613	0.4510	0.8654	0.9301	0.7300	0.8973	0.8053	0.9291
4→7	0.9988	0.9418	0.9276	0.3503	0.9429	0.9337	0.7943	0.9032	0.8310	0.9288
2→3	0.9355	0.9331	0.8374	0.2717	0.8420	0.9301	0.7674	0.9233	0.8854	0.9281
Average	0.9599	0.9358	0.8298	0.3465	0.8408	0.9315	0.8357	0.8717	0.7960	0.9256

From the experimental data, we can see that the perturbations generated by the GN-TUAP algorithm have excellent performance in terms of anti-detectability, especially on the CIFAR-10 dataset, where the defense method often needs to disrupt the model's performance on clean samples to weed out our toxic samples, while on the GTSRB dataset the SFT (super-fine-tuning) defense method has good results, which is related to the composition of the dataset as well as to the mechanism of action of the defense method. The GTSRB dataset is highly similar under a certain category, it is different from the CIFAR-10 dataset, the CIFAR dataset has a greater disparity of images under a certain category, the SFT defense method is based on the method of adjusting the learning rate during the training so as to allow the model to forget the features of the triggers, for a single category of similarity in the GTSRB dataset in the case of a toxic number of samples does not dominate the situation, the features of the triggers are easy to be forgotten thus resulting in the decline in the effectiveness of the attack is obvious.

4.5 Noise Filtration Resistance Performance

From the previous experiments, we can see that the trigger generated by the GN-TUAP algorithm exhibits better detection evasion capability when facing defense detection methods. This is mainly due to the restriction of the direction and distribution of perturbations using noise satisfying the Gaussian distribution in the method, which reduces the presence of anomalous pixels. So, can the trigger survive under noise filtering methods? To further demonstrate the robustness of triggers generated by the GN-TUAP method, we conducted additional experiments using some typical and effective noise filtering methods that perform well on Gaussian noise. The experiments were conducted using VGGNet as the victim model, CIFAR-10 dataset, with the attack strategy set to the one-to-one label attack strategy, and the poisoning rate set to 3% of the training samples. The noise filtering methods used included Gaussian Filter, Non-Local Means Filter, Wiener filtering, and Wavelet Transform Filter, with the intensity of noise filtering adjusted by adjusting the convolution kernel size during the experiments. Fig. 12 illustrates the effects of the four noise filtering techniques on clean samples, whereas Fig. 13 depicts their impact on toxic samples.

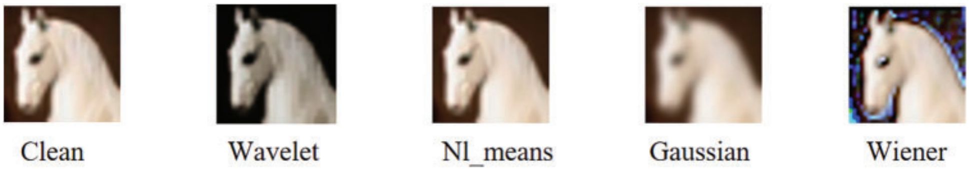

Figure 12: Effectiveness of three noise filtering means on clean samples

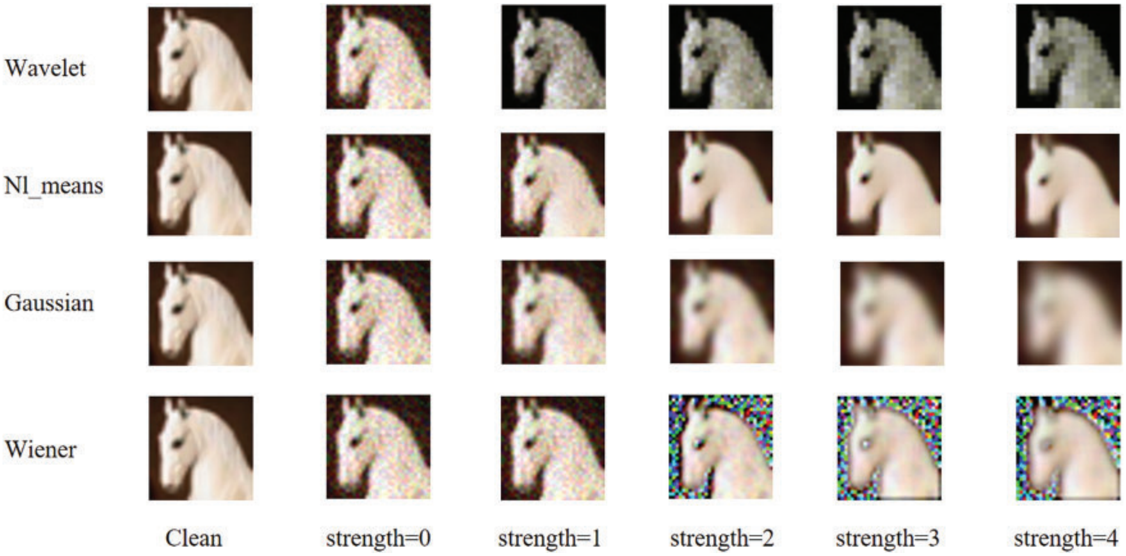

Figure 13: Effectiveness of three noise filtering means on toxic samples

Gaussian Filter is based on the Gaussian function itself. It can smooth images, reduce noise, but also blur image edges. The key to Gaussian filtering lies in selecting the appropriate kernel size and standard deviation to achieve a balance between denoising and preserving details. Non-Local Means Filter is based on repeated texture information in the image. It replaces the current pixel value with the weighted average of similar pixels by comparing each pixel in the image with its surrounding pixels. This method effectively removes noise while preserving image details and structures, especially performing well in low-texture areas. Wavelet Transform Filter decomposes the image into wavelet coefficients of different scales, and then applies thresholding to these coefficients to remove noise. Choosing the appropriate wavelet basis and threshold is crucial. Wavelet Transform Filter is particularly suitable for processing images with multi-scale features. Wiener filtering is a statistical-based linear filtering method that adapts to the denoising needs of different regions by estimating the local variance of the image. Wiener filtering can remove random noise while preserving image details as much as possible.

From the experimental data in Table 12, we can see that the four noise filtering methods have different effects on the effectiveness of attacks and the performance of the model. Overall, the wavelet transform noise filtering method performs the best, reducing the attack success rate by approximately 18% while having minimal impact on the model. The non-local means denoising method reduces the attack success rate by around 15%, with the least impact on the model's performance on clean samples. Gaussian blur and Wiener filtering methods have relatively larger impacts on the model, which can affect the model's performance in practical use. To further explore the effects of noise filtering methods on the perturbation triggers generated by the GN-TUAP algorithm, we conducted additional experiments by varying the intensity of noise filtering methods and adding more noise filtering methods based on previous experimental settings. Detailed experimental data can be found in Table 13.

Table 13: The effect of the three noise filtering methods on the ASR metrics and ACC metrics at different denoising strengths

Noise filtering methods	S1		S2		S3		S4		S50	
	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC
Median	0.9031	0.9092	0.8306	0.8045	0.9505	0.6133	0.7024	0.4225	0.3191	0.2350
Mean	0.9087	0.9103	0.8808	0.5313	0.6676	0.2496	0.2689	0.1288	0.0619	0.0575
Gaussian	0.7822	0.3143	0.3221	0.2007	0.1390	0.1431	0.1055	0.1782	0.1597	0.1393
NL_means	0.8735	0.9317	0.7295	0.9008	0.7644	0.8955	0.7483	0.7911	0.5372	0.3218
Wiener	0.9240	0.9226	0.8348	0.8482	0.5133	0.4808	0.1928	0.3086	0.1573	0.1048
Wavelet	0.7203	0.9300	0.7815	0.9150	0.7645	0.9130	0.8285	0.9173	0.7997	0.8897

From the data in Table 13, we can observe that as the intensity of noise filtering increases, the impact on the attack success rate also increases. Meanwhile, the impact on the model also increases. However, the impact on the triggers remains within an acceptable range. The wavelet transform method can achieve a good balance between image quality and denoising level. Therefore, during the process of increasing denoising intensity, the impact of the wavelet transform method on the model performance is relatively smaller compared to other methods. However, its impact on the attack success rate is not fatal enough.

Based on the experimental data in this section, it is evident that noise filtering methods have an impact on triggers generated by the GN-TUAP algorithm, but this impact is acceptable. Furthermore, noise filtering methods also affect the model's performance. Therefore, to maintain the model's performance, researchers have to use lower filtering intensity and employ methods that preserve image details, such as the wavelet transform filtering method. From the experimental data, we can see that the wavelet transform noise filtering method reduces the attack success rate by around 20%, which is excellent compared to other noise filtering methods. However, this impact is not fatal enough for attackers, as an attack success rate of approximately 75% already poses a serious security threat to the model.

From a theoretical perspective, the reason why noise filtering methods cannot cause fatal damage to trigger noise is that the perturbation triggers generated by the GN-TUAP algorithm are essentially highly correlated and directional perturbations related to the content of the samples. These perturbations have differences in features compared to noise. In Fig. 14, we show triggers generated by the GN-TUAP and TUAP methods, as well as some common noises. We can see that compared to the TUAP method, the triggers generated by the GN-TUAP method have some characteristics of noise, but they are still very different from noise. The distribution of noise is more uniform, and each trigger generated for different categories has its unique features. In order to demonstrate the differences in filtering effects on perturbations and noise by noise filtering means, we applied four different noise filtering methods to images with perturbation triggers generated by TUAP, GN-TUAP, and pure Gaussian noise, respectively. The specific effects are shown in Fig. 15.

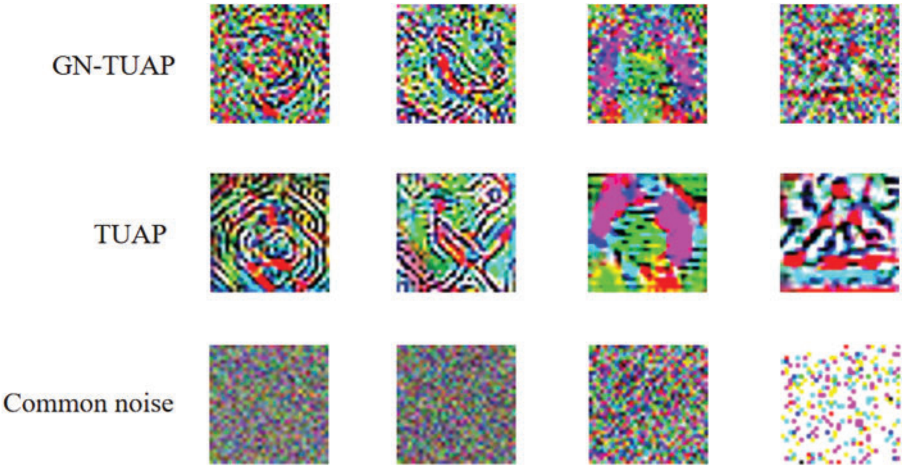

Figure 14: The difference between the triggers generated by GN-TUAP and TUAP and some common noise

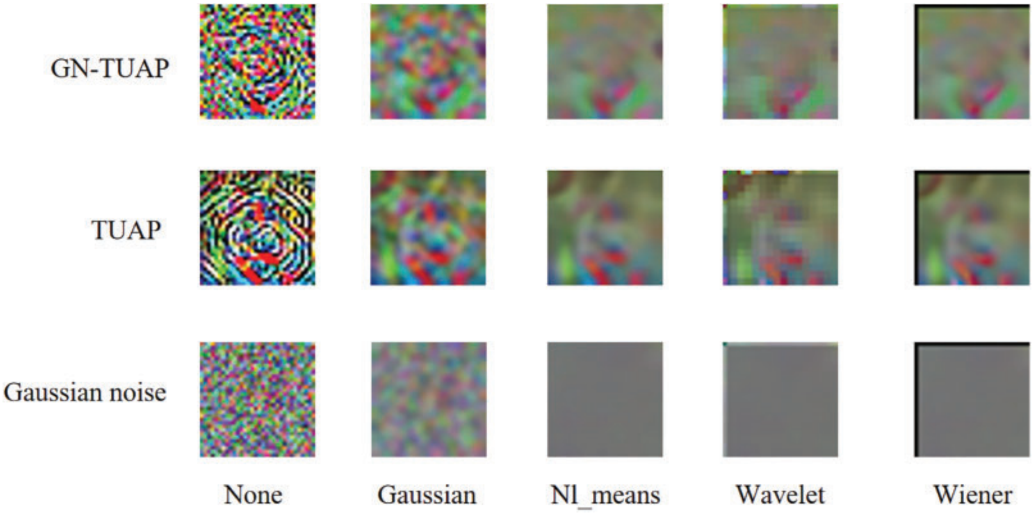

Figure 15: The differences in the filtering effects on perturbation and noise for four types of noise filtering methods

Noise filtering methods are designed based on known noise characteristics, so they cannot filter perturbations with unknown features. Instead, they filter out the part of the triggers that have noise characteristics, similar to how all images are affected to some extent by noise filtering methods. Noise filtering methods filter out what they perceive as noise. Next, we will provide a more detailed explanation from a theoretical perspective.

Firstly, we need to understand how the GN-TUAP algorithm works. GN-TUAP is an algorithm for generating adversarial samples, with the goal of finding the minimal perturbation that causes a slight shift in the input sample in the feature space, resulting in a change in the classification result of the neural network. From the perspective of the neural network, this perturbation causes the input sample to “jump” from the range of one class to the range of another class, while from the human perspective, this perturbation is almost imperceptible. The GN-TUAP algorithm calculates the distance from the input sample to each classification hyperplane, finds the nearest one, and then adds perturbation in that direction to move the sample across this hyperplane. This process is repeated until the sample is classified into a new category.

On the other hand, noise filtering is typically designed to remove random noise from images. This type of noise is often caused by various factors such as electronic noise from devices, interference during signal transmission, etc., and it is randomly distributed in the image, independent of the content and structure of the image. Therefore, noise filtering methods, such as median filtering, mean filtering, Gaussian filtering, etc., can be used to remove this type of noise and restore the original content of the image.

However, the perturbation generated by the GN-TUAP algorithm is not random noise; it is purposeful and directional. This perturbation is carefully designed based on the characteristics of the input sample, the structure and weights of the neural network, and the target category. This perturbation is not randomly distributed in the image but is concentrated on the features that have the greatest impact on the classification result. Therefore, this perturbation is closely related to the content and structure of the image, and noise filtering methods cannot effectively remove it.

Secondly, perturbations designed by the GN-TUAP algorithm are typically very small and almost imperceptible. These small perturbations may be below the detection threshold of noise filtering methods, making them undetectable by these methods. Even if detected, noise filtering methods may not effectively remove such small perturbations due to their size. Additionally, noise filtering is usually based on statistical methods, assuming that noise and signal are independent, and it removes pixels inconsistent with the surrounding area by analyzing the local characteristics of the image. However, in the case of adversarial samples, perturbations and sample content are correlated, and perturbations are carefully designed based on the characteristics of the sample content. Therefore, noise filtering methods struggle to effectively remove such perturbations.

In summary, because the perturbations generated by the GN-TUAP algorithm are purposeful, directional, closely related to the content and structure of the image, and typically very small, noise filtering methods cannot effectively remove them.

4.6 Experimental Results Analysis

Our experiments examine the performance of GN-TUAP algorithm from three aspects: attack effectiveness, stealthiness, and resistance to defense detection, we test our proposed method through multiple perspectives, from the experimental data, we can learn that GN-TUAP algorithm is able to achieve the effectiveness of cross-model attack, under the experimental conditions we set up both one-to-one attack task and many-to-one task, the success rate of the attack is higher than 92%, which means that the attacker can reduce the precondition needed for the attack, meanwhile, the damage of our attack to the model is acceptable, and the change of the initial model's performance on clean samples after being poisoned is less than 5%, which is also commendable in the comparison with other state-of-the-art methods. In terms of covertness, our trigger is more natural and covert than the same type of TUAP method, and less destructive to the original image, which makes it easier to evade manual inspection. In terms of resistance to defense detection, our method performs well, and on the CIFAR-10, CIFAR-100, and GTSRB datasets, GN-TUAP faces defense detection and exhibits generally better anti-detection performance than the other methods undergoing comparison. We also performed noise filtering experiments for our triggers. The experimental results show that our trigger, can survive the noise filtering means.

And we find that the magnitude required for GN-TUAP to generate perturbations is different due to the different variability among different classes, so different attack tasks are detected with different results, but our triggers are difficult to be removed in most cases, and the defense detection methods often fail to achieve a good defense even in the case of the accuracy of the loop-breaking model.

5 Conclusion

The main contributions of this research lie in the in-depth exploration of the practicality of backdoor attacks in real-world applications and the proposal of a Gaussian Noise-Targeted Universal Adversarial Perturbation (GN-TUAP) algorithm, significantly enhancing the concealment and stability of backdoor triggers. By designing a threat model and adopting appropriate evaluation metrics, we have provided a more rational and accurate analytical approach for the study and defense against backdoor attacks.

However, we acknowledge that the field of backdoor attacks still faces some challenges. Firstly, the concealment and stability of triggers are crucial for successful attacks. Although methods like adversarial perturbations and TUAP have improved trigger concealment to some extent, they still rely on the original dataset and model structure, leading to unstable attack performance in different environments. To address this, more research is needed to design trigger generation methods with greater generality and applicability, reducing their dependency on the original dataset and model structure. Secondly, the issue of anti-detection is another significant challenge. Although some detection methods can identify certain backdoor triggers, there is no trigger that can entirely evade all detection techniques. Thus, enhancing the anti-detection capability of backdoor triggers is a pressing matter. In the study of backdoor triggers, it is essential to consider the characteristics of different detection methods and design more concealed triggers or use adversarial attacks to bypass detection mechanisms. Furthermore, backdoor attacks pose a threat to data privacy and security. Attackers may inject toxic samples to tamper with the dataset, causing the trained model to perform well under normal conditions but exhibit misjudgments under specific backdoor conditions. Therefore, it is essential to strengthen data protection measures against tampering and ensure the integrity and trustworthiness of datasets.

To tackle these challenges, we propose the following recommendations: Firstly, enhance research on trigger generation algorithms, particularly by reducing their dependency on the original dataset and model structure, and improving the generality and stability of triggers. Secondly, conduct more in-depth research to enhance the anti-detection capabilities of backdoor triggers, exploring more adversarial attack techniques to bypass detection mechanisms. Simultaneously, strengthen data security measures to prevent malicious data tampering. Additionally, fostering collaboration with the industry to jointly research defense methods against backdoor attacks can enhance system security and credibility. Through sustained research and collaboration, we can achieve more groundbreaking progress in the field of backdoor attacks, ensuring the security and reliability of artificial intelligence systems.

Acknowledgement: The authors wish to express their appreciation to the reviewers for their helpful suggestions which greatly improved the presentation of this paper.

Funding Statement: This research was funded by National Natural Science Foundation of China under Grant No. 61806171, The Sichuan University of Science & Engineering Talent Project under Grant No. 2021RC15, Sichuan University of Science & Engineering Graduate Student Innovation

Fund under Grant No. Y2023115, The Scientific Research and Innovation Team Program of Sichuan University of Science and Technology under Grant No. SUSE652A006.

Author Contributions: The authors confirm contribution to the paper as follows: project supervision: Hong Huang, research conception and design: Hong Huang, Yunfei Wang; initial draft writing: Hong Huang, Yunfei Wang, Guotao Yuan; manuscript review and editing: Hong Huang, Yunfei Wang, Guotao Yuan, Xin Li; experimental data collection and organization: Yunfei Wang, Guotao Yuan, Xin Li; experimental results analysis and interpretation: Xin Li; graph design: Guotao Yuan, Xin Li. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The three datasets used in this study can be found in references [29–31], respectively.

Conflicts of Interest: The authors declare that there is no conflict of interest regarding the publication of this paper.

References

[1] Y. Li, Y. Jiang, Z. Li, and S. T. Xia, “Backdoor learning: A survey,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 1, pp. 5–22, Jan. 2024. doi: 10.1109/TNNLS.2022.3182979.

[2] H. Ali *et al.*, “A survey on attacks and their countermeasures in deep learning: Applications in deep neural networks, federated, transfer, and deep reinforcement learning,” *IEEE Access*, vol. 11, pp. 120095–120130, 2023. doi: 10.1109/ACCESS.2023.3326410.

[3] H. He, Z. Zhu, and X. Zhang, “Adaptive backdoor attack against deep neural networks,” *Comput. Model. Eng. Sci.*, vol. 136, no. 3, pp. 2617–2633, 2023. doi: 10.32604/cmes.2023.025923.

[4] R. Jin and X. Li, “Backdoor attack and defense in federated generative adversarial network-based medical image synthesis,” *Med. Image Anal.*, vol. 90, no. 1, pp. 102965, Dec. 2023. doi: 10.1016/j.media.2023.102965.

[5] Y. Feng, B. Ma, J. Zhang, S. Zhao, Y. Xia and D. Tao, “FIBA: Frequency-injection based backdoor attack in medical image analysis,” in *2022 IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, New Orleans, LA, USA, IEEE, Jun. 2022, pp. 20844–20853. doi: 10.1109/CVPR52688.2022.02021.

[6] M. Xue, C. He, J. Wang, and W. Liu, “Backdoors hidden in facial features: A novel invisible backdoor attack against face recognition systems,” *Peer-to-Peer Netw. Appl.*, vol. 14, no. 3, pp. 1458–1474, May 2021. doi: 10.1007/s12083-020-01031-z.

[7] I. Masi, Y. Wu, T. Hassner, and P. Natarajan, “Deep face recognition: A survey,” in *2018 31st SIBGRAPI Conf. Graphics, Patterns and Images (SIBGRAPI)*, Parana, Brazil, IEEE, Oct. 2018, pp. 471–478. doi: 10.1109/SIBGRAPI.2018.00067.

[8] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg, “BadNets: Evaluating backdooring attacks on deep neural networks,” *IEEE Access*, vol. 7, pp. 47230–47244, 2019. doi: 10.1109/ACCESS.2019.2909068.

[9] X. Qi, T. Xie, R. Pan, J. Zhu, Y. Yang and K. Bu, “Towards practical deployment-stage backdoor attack on deep neural networks,” in *2022 IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, New Orleans, LA, USA, IEEE, Jun. 2022, pp. 13337–13347. doi: 10.1109/CVPR52688.2022.01299.

[10] D. M. Alghazzawi, O. B. J. Rabie, S. Bhatia, and S. H. Hasan, “An improved optimized model for invisible backdoor attack creation using steganography,” *Comput. Mater. Contin.*, vol. 72, no. 1, pp. 1173–1193, 2022. doi: 10.32604/cmc.2022.022748.

[11] S. Li, M. Xue, B. Zhao, H. Zhu, and X. Zhang, “Invisible backdoor attacks on deep neural networks via steganography and regularization,” *IEEE Trans. Dependable Secure Comput.*, vol. 12, no. 5, pp. 2088–2105, 2020. doi: 10.1109/TDSC.2020.3021407.

[12] Y. Yao, H. Li, H. Zheng, and B. Y. Zhao, “Latent backdoor attacks on deep neural networks,” in *Proc. 2019 ACM SIGSAC Conf. Comput. Commun. Secur.*, London, UK, ACM, Nov. 2019, pp. 2041–2055. doi: 10.1145/3319535.3354209.

[13] T. J. L. Tan and R. Shokri, "Bypassing backdoor detection algorithms in deep learning," in *2020 IEEE Eur. Symp. Secur. Priv. (EuroS&P)*, Genoa, Italy, IEEE, Sep. 2020, pp. 175–183. doi: 10.1109/EuroSP48549.2020.00019.

[14] H. Zhong, C. Liao, A. C. Squicciarini, S. Zhu, and D. Miller, "Backdoor embedding in convolutional neural network models via invisible perturbation," in *Proc. Tenth ACM Conf. Data and Appl. Secur. Priv.*, New Orleans, LA, USA, ACM, Mar. 2020, pp. 97–108. doi: 10.1145/3374664.3375751.

[15] Q. Zhang, Y. Ding, Y. Tian, J. Guo, M. Yuan and Y. Jiang, "AdvDoor: Adversarial backdoor attack of deep learning system," in *Proc. 30th ACM SIGSOFT Int. Symp. Softw. Testing and Anal.*, Denmark, ACM, Jul. 2021, pp. 127–138. doi: 10.1145/3460319.3464809.

[16] B. Tran, J. Li, and A. Madry, "Spectral signatures in backdoor attacks," in *Adv. Neural Inf. Process. Syst. 31: Annual Conf. Neural Inf. Process. Syst. 2018, NeurIPS 2018*, Montréal, QC, Canada, Curran Associates, Inc., 2018, pp. 8011–8021. Accessed: May 13, 2024. [Online]. Available: https://proceedings.neurips.cc/paper/2018/hash/280cf18baf4311c92aa5a042336587d3-Abstract.html.

[17] B. Chen *et al.*, "Detecting backdoor attacks on deep neural networks by activation clustering," in *Workshop on Artif. Intell. Safety 2019 Co-Located with the Thirty-Third AAAI Conf. Artif. Intell. 2019 (AAAI-19)*, Honolulu, HI, USA, 2019. Accessed: May 13, 2024. [Online]. Available: https://ceur-ws.org/Vol-2301/paper_18.pdf

[18] D. J. Miller, Z. Xiang, and G. Kesidis, "Adversarial learning targeting deep neural network classification: A comprehensive review of defenses against attacks," *Proc. IEEE*, vol. 108, no. 3, pp. 402–433, Mar. 2020. doi: 10.1109/JPROC.2020.2970615.

[19] Y. Li, Y. Li, B. Wu, L. Li, R. He and S. Lyu, "Invisible backdoor attack with sample-specific triggers," in *2021 IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Montreal, QC, Canada, IEEE, Oct. 2021, pp. 16443–16452. doi: 10.1109/ICCV48922.2021.01615.

[20] Z. Wang, J. Zhai, and S. Ma, "BppAttack: Stealthy and efficient trojan attacks against deep neural networks via image quantization and contrastive adversarial learning," in *2022 IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, New Orleans, LA, USA, IEEE, Jun. 2022, pp. 15054–15063. doi: 10.1109/CVPR52688.2022.01465.

[21] X. Qi, T. Xie, Y. Li, S. Mahloujifar, and P. Mittal, "Revisiting the assumption of latent separability for backdoor defenses," presented at the Eleventh Int. Conf. Learn. Rep., ICLR 2023, Kigali, Rwanda, Sep. 2022. Accessed: May 13, 2024. [Online]. Available: https://openreview.net/forum?id=_wSHsgrVali

[22] S. Kaviani and I. Sohn, "Defense against neural trojan attacks: A survey," *Neurocomputing*, vol. 423, no. 2, pp. 651–667, Jan. 2021. doi: 10.1016/j.neucom.2020.07.133.

[23] I. Riadi and E. I. Aristianto, "An analysis of vulnerability web against attack unrestricted image file upload," *Comput. Eng. Appl. J.*, vol. 5, no. 1, pp. 19–28, Feb. 2016. doi: 10.18495/comengapp.v5i1.161.

[24] S. M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A simple and accurate method to fool deep neural networks," in *2016 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, IEEE, Jun. 2016, pp. 2574–2582. doi: 10.1109/CVPR.2016.282.

[25] Z. Sha, X. He, P. Berrang, M. Humbert, and Y. Zhang, "Fine-tuning is all you need to mitigate backdoor attacks," in *The Twelfth Int. Conf. Learn. Rep., ICLR 2024*, Vienna, Austria, 2024. Accessed: May 13, 2024. [Online]. Available: https://openreview.net/forum?id=ywGSgEmOYb

[26] Y. Zeng, S. Chen, W. Park, Z. Mao, M. Jin, and R. Jia, "Adversarial unlearning of backdoors via implicit hypergradient," presented at the The Tenth Int. Conf. Learn. Rep., ICLR 2022, Virtual Event, Oct. 2021. Accessed: May 13, 2024. [Online]. Available: https://openreview.net/forum?id=MeeQkFYVbzW

[27] J. Guo, Y. Li, X. Chen, H. Guo, L. Sun and C. Liu, "SCALE-UP: An efficient black-box input-level backdoor detection via analyzing scaled prediction consistency," in *The Eleventh Int. Conf. Learn. Rep., ICLR 2023*, Virtual Event, Sep. 2022. Accessed: May 13, 2024. [Online]. Available: https://openreview.net/forum?id=o0LFPcoFKnr

[28] G. Tao *et al.*, "Model orthogonalization: Class distance hardening in neural networks for better security," in *2022 IEEE Symp. Security and Privacy (SP)*, San Francisco, CA, USA, IEEE, May 2022, pp. 1372–1389. doi: 10.1109/SP46214.2022.9833688.

[29] A. Krizhevsky and G. Hinton, *Learning multiple layers of features from tiny images*. Toronto, ON, Canada: University of Toronto, 2009. Accessed: May 13, 2024. [Online]. Available: http://www.cs.utoronto.ca/~kriz/learning-features-2009-TR.pdf

[30] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Netw*, vol. 32, no. 1, pp. 323–332, Aug. 2012. doi: 10.1016/j.neunet.2012.02.016.

[31] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998. doi: 10.1109/5.726791.