ICT: Invisible Computable Trigger Backdoor Attacks in Transfer Learning

Xiang Chen, Bo Liu, Shaofeng Zhao, Ming Liu, Hui Xu, Zhanbo Li, and Zhigao Zheng, *Member, IEEE*

***Abstract*—Transfer learning is a commonly used technique in machine learning to reduce the cost of training models. However, it is susceptible to backdoor attacks that cause models to misclassify data with specific triggers while behaving normally on clean data. Existing methods for backdoor attacks in transfer learning either do not consider attack stealthiness or require compromising attack effectiveness for trigger concealment. To overcome this challenge, we introduce the concept of Invisible and Computable Trigger (ICT), which involves two critical steps. First, we propose a new computable trigger obtained by training on input data to greatly increase the attack effect during inference. Second, we embed the trigger into an imperceptible perturbation, allowing poisoned data to appear indistinguishable from clean data. Our experimental results demonstrate that our approach outperforms state-of-the-art methods in both attack effect and stealthiness.**

***Index Terms*—Neural network, transfer learning, computer security, backdoor attack, stealthiness.**

I. Introduction

DEEP neural networks have been successfully applied in many fields, including object detection [1], [2], natural language processing [3], [4] and smart healthcare [5], [6]. The success of deep neural networks is widely attributed to their enormous model scale and vast training data, as they require training millions of parameters to learn valuable information from large volumes of data. However, this process is difficult for many resource-constrained users due to the significant human and computational resources required.

To reduce the training cost, federated learning [7] and transfer learning [8], [9] have been widely used in the training process. The main idea is to use a model trained on the source domain to accelerate the learning speed and improve the learning effect on the target domain by sharing its knowledge and features. This greatly reduces the annotation budget and training cost. A common transfer learning technique involves using a pre-trained model, where some of the parameters of the model are frozen, and the later layers of the model are fine-tuned for a new task. This reduces the number of parameters needed to train the model.

The security issues of deep learning models have been brought to light due to their sensitivity to training data [10]. When a user needs to train a model, they may have to download data from an untrusted third party, which could result in a backdoor attack [11], [12], [13]. The attacker chooses a trigger that activates the backdoor, causing the model to misclassify data marked with the trigger. If a model deployed on a Cyber-Physical System (CPS) or Consumer Electronics (CE) is attacked, the consequences could be severe [14], [15], [16]. For example, a consumer-facing electronic device [17] that has been implanted with a backdoor may recognize an expensive item as a cheap drink, causing problems when the user pays.

Common data poisoning attacks [18], [19] will completely destroy the accuracy of the model, so subsequent work [20] proposed targeted data poisoning attacks. However, the attack only makes the model misclassify a fixed amount of the data and lacks generalization. Backdoor attacks are more flexible, as shown in Figure 1, for any image, only the backdoor is activated when the trigger appears, causing the model to output the label specified by the attacker [21], [22]. Backdoor attacks are also different from adversarial sample attacks [10], [23], [24] that require the attacker to be able to control the inference process of the model. The generation of each adversarial sample requires backpropagation, which will make the attack easy to detect. Once the trigger in a backdoor attack is selected, it does not need to be changed. In the inference phase, any picture patched with a trigger will cause the model to misclassify.

In many existing attack methods, the choice of trigger is usually arbitrary, and it can be any meaningful or meaningless pattern [12], [14], [29]. But such triggers have two drawbacks: first, such triggers are detectable, and poisoned data can be easily identified. Then the user may refuse to use the data

Received 9 April 2024; revised 7 June 2024 and 3 August 2024; accepted 1 October 2024. Date of publication 8 October 2024; date of current version 31 December 2024. This work was supported in part by the National Key Research and Development Program of China under Grant 2023YFB4502704; in part by the National Natural Science Foundation of China under Grant 62402451 and Grant 62372333; and in part by the Henan Provincial Science and Technology Key Project Foundation under Grant 222102210334, Grant 222102210252, and Grant 222102210054. *(Corresponding author: Bo Liu.)*

Xiang Chen is with the College of Cyberspace Security, Zhengzhou University, Zhengzhou 450001, China.

Bo Liu and Zhanbo Li are with the Network Center, College of Cyberspace Security, Zhengzhou University, Zhengzhou 450001, China (e-mail: liubo@zzu.edu.cn).

Shaofeng Zhao is with the Information Management Center, Henan University of Economics and Law, Zhengzhou 450046, China.

Ming Liu is with the School of Information Technology, Deakin University, Geelong, VIC 3220, Australia.

Hui Xu is with the China Academy of Industrial Internet, Beijing 100020, China.

Zhigao Zheng is with the School of Computer Science, National Engineering Research Center for Multimedia Software, Institute of Artificial Intelligence, Hubei Key Laboratory of Multimedia and Network Communication Engineering, and the Hubei Luojia Laboratory, Wuhan University, Wuhan 430072, China.

Digital Object Identifier 10.1109/TCE.2024.3476313

TABLE I
THE COMPARISON OF DIFFERENT ATTACK METHODS ACROSS VARIOUS CHARACTERISTICS

Method	Trigger Type	Invisible	Consistent Label	Multi-Class	Transfer Learning
BadNets [12]	Static Trigger	✗	✗	✓	✗
Blended [21]	Static Trigger	✗	✗	✓	-
Rethink [25]	Static Trigger	✗	✗	✓	✗
Hidden Trigger [26]	Static Trigger	✓	✓	✗	✗
Trojan [14]	Computable Trigger	✗	✗	✓	✓
LIRA [27]	Computable Trigger	✓	✗	✓	-
ATTEQ-NN [28]	Computable Trigger	✓	✗	✓	-
ICT(Ours)	Computable Trigger	✓	✓	✓	✓

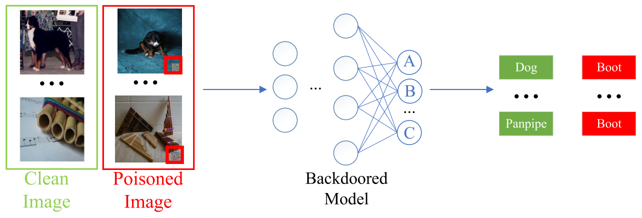


Fig. 1. An illustration of a backdoor attack, where the backdoor is activated only when the trigger is present, leading to misclassification by the model.

set. Second and more importantly, in the context of transfer learning, the use of such a common static trigger to implant a backdoor will lead to an unsatisfactory attack success rate. This is because fine-tuning the model only involves minor weight adjustments[1] to preserve the original classification capabilities of the model. However, these small weight changes are not enough to mitigate the model's inherited classification capabilities. This then inspires us to raise the following two questions: 1) for the invisible backdoor attack on the pre-trained model [26], can we use more targeted triggers according to the specific model, to improve the attack success rate? 2) while improving the success rate of the attack, can we ensure that the stealthiness of the attack is not affected, and can we use the improvement of the attack performance to design more stealthy attacks?

To answer these two questions, we propose a novel trigger design method. More specifically, we propose a new trigger generation algorithm that makes the triggers more targeted — different models would use different triggers, which greatly improves the success rate of invisible backdoors in transfer learning. In addition, in order to make the attack more stealthy, we will generate a tiny perturbation to replace the function of the trigger, and will not modify the label of the poisoned data. This makes the poisoned data not only appear the same as the clean data but also their label is not replaced. (Most existing attack methods require modifying the poisoned data's label [30], [31], [32]). Through the integration of these two phases, our methodology has secured the foremost performance on various attributes, as delineated in Table I.

In particular, the main differences between our approach and previous similar approaches [26] are: 1) this work investigates the behavior of dynamic model-dependent invisible triggers in backdoor attacks, whereas previous work used static triggers that were pre-specified by the attacker. 2) We try to attack the model under the setting of multi-class classification, which greatly increases the difficulty of attack for the attacks of invisible triggers and clean labels, while the previous attack methods can only be carried out under the condition of binary classification [26], which limits the generalization ability of the attack and reduces the risk of attack. We even tested the effect of the attack on the task of 1000 class image classification based on ImageNet(ILSVRC),[2] which significantly differs from the binary classification setting. 3) We further verify the effect of our attack under different defense strategies and find that our attack could be executed under different models with better generalization ability, while previous work was only tested on AlexNet [26].

The main contributions of this paper are as follows:

1) We propose a new trigger design method, which generates different triggers according to different models, and distributes the effect of the trigger to a small perturbation. It can increase the attack success rate from 29.61% to 96.65% while ensuring concealment.
2) Our attacks can be conducted at a smaller perturbation threshold, which results in the poisoned images being closer to the original images, thus bypassing some existing defenses. Our method can attack under a multi-class classification model, which greatly increases the capability of attack.
3) A comprehensive comparison has been conducted with both analogous and state-of-the-art works. Extensive experiments on four datasets are conducted, which verify the effectiveness of the proposed method.

II. RELATED WORK

A. Backdoor Attack

Visible Trigger-based Attack: Gu et al. [12] proposes the first backdoor attack method in deep neural networks, in which Gu believes that the essence of the existence of backdoor attacks is to establish a connection between a trigger and the target label. Liu et al. [14] proposes to dynamically generate a trigger according to the model's structure and parameters to inject the backdoor. The trigger generated in this way can make the trained backdoor model return a higher attack success rate in the transfer learning scenario. Wang et al. [33] proposes a work similar to ours, which also does not use static triggers in transfer learning, while the poisoned pictures

[1]Without loss of generality, we assume that the user fine-tunes only the last layer of the model due to resource constraints.

[2]Dataset link: https://image-net.org/download.php.

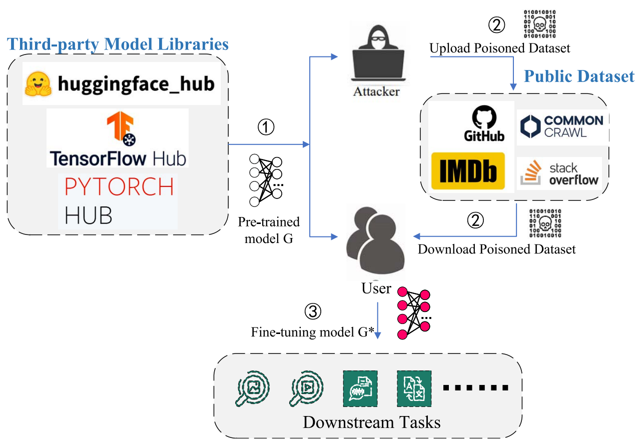


Fig. 2. Realistic scenarios of our attack method, and the capabilities of the attacker.

can be easily detected. Xie et al. [34] makes full use of the characteristics of federated learning and proposes a backdoor attack based on distributed triggers. Lv et al. [35] proposes an attack on transformer models, which have a completely different structure and principle compared with convolutional neural. These attacks are all based on patch-based triggers, as a result, the trigger is easily perceptible to human inspectors.

Invisible Trigger-based Attack: Chen et al. [21] first proposes backdoor attacks with invisible triggers, which makes backdoor attacks more difficult to detect. After that, Nguyen and Tran [36] hides the trigger by applying a warping transformation to the image. Doan et al. [27] treats the trigger generation process as a non-convex constrained optimization problem and uses a two-stage stochastic optimization process to learn a hidden noise as a trigger. Bagdasaryan and Shmatikov [29] eliminates the need to directly add triggers to the data by injecting malicious code into the training code. Gong et al. [37] changes the RGB value of every pixel in the picture, and the backdoor is activated when the model detects a specific change. For the multilabel classification task, Chen et al. [38] uses multiple labels as trigger.

Clean Label-based Attack: To further improve the concealment of poisoned images, Turner et al. [39] first proposes the idea of a clean-label attack. He uses adversarial perturbation or generative models to change the images of the target class so that the poisoned images do not need to change the label. Since then, a lot of related work has been proposed [22], [26]. Wang et al. [40] proposes a sequential trigger that injects hidden backdoors by changing the order of training samples without changing the samples. However, the above works cannot achieve efficient attacks under transfer learning and invisible triggers [33]. In this paper, we aim to achieve a satisfactory success rate of attack while ensuring stealthiness.

B. Backdoor Defence

In order to alleviate backdoor attacks, various defense methods have been explored, which can be mainly divided into three categories [41], including trigger-backdoor mismatch, backdoor elimination, and trigger elimination.

Trigger-Backdoor Mismatch works by breaking the connection between the trigger and the backdoor, usually adding a pre-processing phase between the data and the model so that the changed flip-flop cannot activate the backdoor hidden in the model. For example, Li et al. [25] proposes that in the test phase, small changes (e.g., left and right flip or a certain proportion of zoom) were made to each image, which will have no effect on the recognition of clean images, but could make the trigger no longer effective.

Backdoor Elimination tries to remove the backdoor in the model [42], [43]. Liu et al. [44] believes that the root cause of the backdoor's existence is the model's redundant classification ability. If the redundant neurons that are not activated by clean images can be pruned, then the backdoor can be removed. Wang et al. [45] showed that for a model with backdoors, the disturbance required to misclassify a picture into an attack label is much smaller than the disturbance required to misclassify a picture into another label. After the attack label is identified, the backdoor is eliminated through "Filter, Pruning, and Unlearning" techniques.

Trigger Elimination method occurs in the inference stage [46]. Gao et al. [47] proposes to filter the attacked samples by superimposing various images on the input samples. The less randomness of the perturbation input prediction, the higher the probability that the input sample will be attacked.

III. Threat Model

Attacker Ability: Let's assume that a user downloads a well-known model from a trusted source, such as AlexNet and ResNet, and then downloads a batch of data from an untrusted third party. The user uses limited training resources to train a desired model through transfer learning. Figure 2 presents the actual scenario and illustrates the specific attack environment. Under such conditions, the attacker only knows the structure of the pre-trained model (called G) and accomplishes the poisoning of the data based on it. However, the attacker does not know the new model structure (called G*) when the user fine-tunes it. This is the minimum required to carry out a backdoor attack [41] so that our attack has very good generalization ability.

Attacker Goal: In previous work, the user would download the entire training dataset $\mathcal{D}_{train} = \{(x_i, y_i)\}_{i=1}^{N}$, where $x_i \in \mathcal{X} = \{0, \ldots, 255\}^{C\times W\times H}, y_i \in \mathcal{Y} = \{1, \ldots, K\}$. C, W, H represent the number of channels, width, and height of a picture, respectively, and N $= |\mathcal{D}_{train}|$ represents the amount of data. The attacker selects a trigger t, a mask M, and a target class $\hat{y}$ in advance and uses the trigger injection function Eq. (1) to generate the poisoned data $\mathcal{D}_{poison} = \{(T(x_i), \hat{y})\}$

$$T(x) = (1 - M) \odot x + M \odot t \tag{1}$$

where M is a two-dimensional matrix of the same size as x, 1 at the trigger position and 0 everywhere else, $\otimes$ represents multiplication by elements. The attacker's goal is for the user to train a backdoor classification model $f_w : X \to [0, 1]^K$ so that f_w behaves normally on clean data, but when a trigger

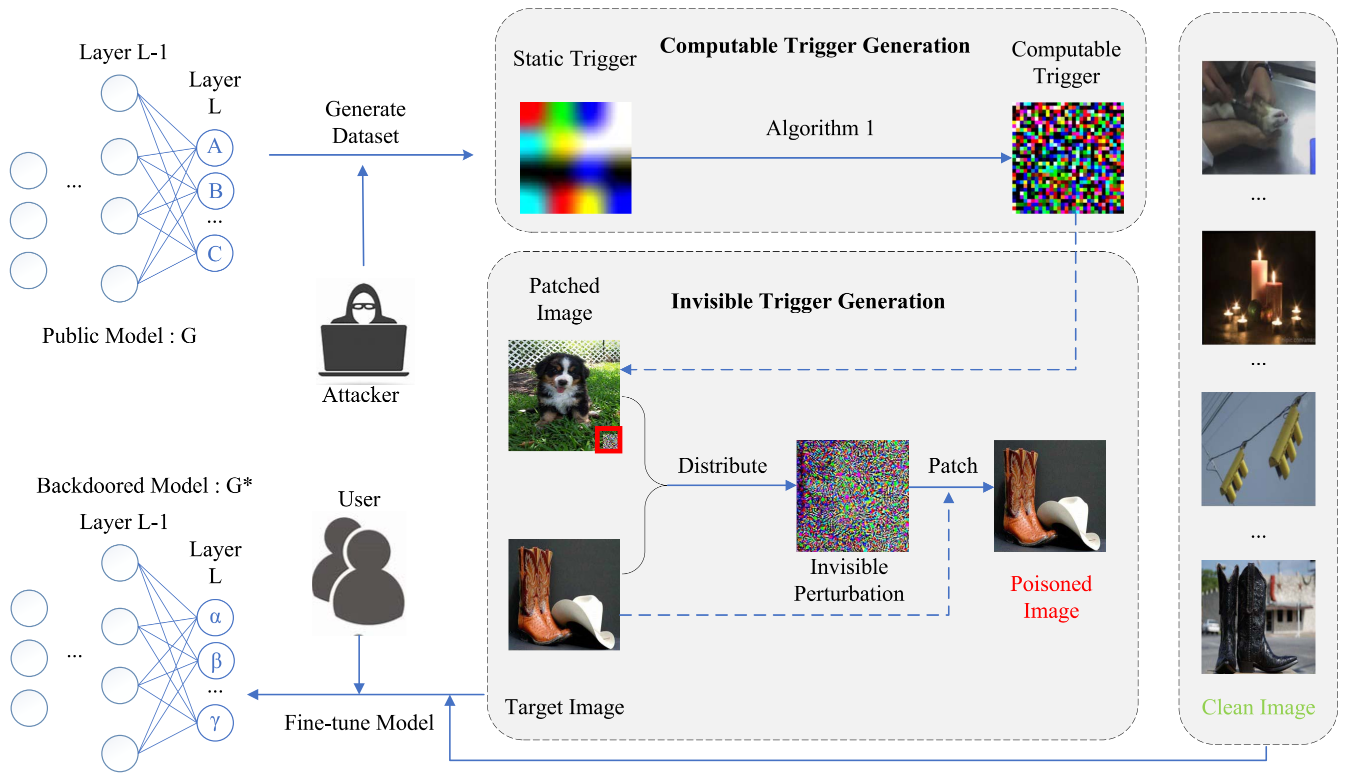


Fig. 3. The overall flow chart of the attack. After downloading the model, the attacker utilizes Algorithm 1 to generate a computable trigger. Subsequently, the functionality of the trigger is concealed by distributing it onto a minute perturbation. Following fine-tuning, the backdoor is implanted into the model. (For ease of demonstration, the added perturbation is amplified by 255 times; in the following, this will not be restated.).

is present, f_w will always output the class specified by the attacker, which is formally defined as follows:

$$w^* = \arg\max_{w} \left(\sum_{x_i \in \mathcal{D}_{poison}} I\big[f(T(x_i)) = \hat{y}\big] + \sum_{x_i \in \mathcal{D}_{clean}} I\big[f(x_i) = y_i\big] \right) \tag{2}$$

where $\mathcal{D}_{train} = \mathcal{D}_{clean} \cup \mathcal{D}_{poison}$, poison rate(r) $= \frac{|\mathcal{D}_{poison}|}{|\mathcal{D}_{train}|}$, and $I(\cdot)$ represents indicator function, $I(A) = 1$ if and only if A is true, otherwise $I(A) = 0$.

IV. Our Method

A. Attack Structure

In this section, we show the overall flow of our attack based on the description in Section III. As shown in Figure 3, the attacker and user can download a pre-trained public model G from the Internet. The attacker does not need to know either the hyperparameter settings or the structure and weights of the fine-tuned model. The process of generating the poisoned data by the attacker is independent of the last layer of the model. Then, based on the first L-1 layer of the model, the attacker generates a batch of poisoned data that the trigger is not visible. This process is mainly divided into two parts: the first part is the generation of a computable trigger; the second part is to hide the trigger by converting it into an invisible perturbation so that although the poisoned picture does not appear abnormal, it already has the characteristics of the trigger. The detailed process of these two parts will be described in the subsequent parts of this chapter.

Finally, users unconsciously download the poisoned dataset from the attacker, modify the structure of the last layer of the model, and fine-tune the model to get the injected backdoor model G^*. We would like to reiterate that when generating computable triggers and invisible poisoned data, attackers only know the parameters of the first L-1 layers of the model, and are unaware of the model structure used by users during fine-tuning.

B. Generation of Computable Trigger

For static triggers, if an attacker can only implant a backdoor in the last layer, no matter how to adjust the weight of the model in the last layer, it will not achieve satisfactory results. In other words, modifying only the last layer of the model makes it difficult for the model to learn the existence of triggers and establish associations with the target label. Thus, we use a computable trigger that maximizes the activation of certain neurons in the penultimate layer. When the trigger is present, the activation value of this neuron changes from a very small value to a very large value. We adopt this approach because this approach makes it easier for the model to perceive the trigger, and thus the backdoor can be implanted without large changes in the model weights.

As shown in Figure 4, we first determine the size, position, and shape of the trigger and randomly initialize the pixel values. In order to compare with the previous method, we

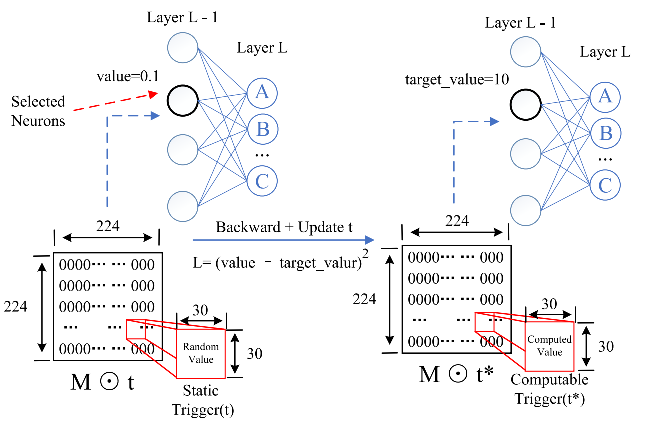


Fig. 4. Generation of a computable trigger. M is a mask with a value of 1 at the trigger position and 0 elsewhere. t represents a random static trigger. After updating the trigger values through backpropagation, t* causes the activation value of the targeted neuron to increase from 0.1 to 10.

Algorithm 1 Computable Trigger Generate Algorithm

```
Input: Model $G$, target_value, input $x$, Mask $M$
Output: trigger
1:  // Select from layer (1) to layer (L−1) of the model
2:  $SubNet\_G = G[: -1]$
3:  // Initialize the trigger according to the mask
4:  $trigger = Random\_Init(M)$
5:  // Patch the trigger on input data
6:  $T(x) = (1 - M) \odot x + M \odot trigger$
7:  // Compute activation value and define loss function
8:  $value = SubNet\_G(T(x))$
9:  $loss \overset{def}{=} MSE(value, target\_value)$
10: // The trigger's value is calculated based on the gradient
11: repeat
12:     $grad = \frac{\partial loss}{\partial trigger}$
13:     $grad = grad \odot M$
14:     $trigger = trigger - lr \cdot grad$
15: until $i > epochs$ or $loss < threshold$
16: return $trigger$
```

follow the previous method, making the trigger a $3 \times 30 \times 30$ pixel square. But in order to make the trigger fit into the model for computation, we use Eq. (1) to put the trigger into an all-zero matrix. We follow existing work [14] and select one neuron to control trigger generation. Subsequent experiments showed that selecting one neuron was enough to achieve desirable results. An important question now is which neuron is chosen to generate the trigger? [14] finds that for some neurons, no matter how the learning rate is adjusted, their activation values do not show obvious changes. Previous works [14], [33] select target neurons based on the scale of the weight sum or the scale of the activation sum. These methods, while seemingly quick to find a neuron, are often suboptimal and poorly interpreted. To this end, we propose an improved neuronal search method that iteratively traverses each neuron and applies the gradient descent algorithm at various learning rates. This approach aims to identify the global optimal solution more effectively.

After selecting the neuron, we use the Mean Squared Error (MSE) as the loss function, which is computed between the value of this neuron (the activation output of the second last layer before injecting the trigger) and the target value (which is pre-defined), as shown in Eq. (3).

$$\begin{aligned} t^* &= \text{Generator}(G, target\_value) \\ &= \arg\min_t MSE(value, target\_value) \end{aligned} \tag{3}$$

We update the pixel value of the trigger region through the gradient descent algorithm so that the activation value of the neuron gradually approaches the target value. The detailed process of generating the trigger is shown in Algorithm 1.

C. Generation of Invisible Triggers

Computable triggers make it easier to inject backdoors into pre-trained models, but they do not take into account the stealth of triggers. In order to make the trigger invisible, we propose to add a small perturbation to the picture, so that it has the same functionality as the trigger. The term "same functionality" can be understood from two perspectives. From the neural network's viewpoint, it refers to the similarity in activation values at the penultimate layer of the model. From the practical application perspective, it denotes that when an image is patched with this perturbation, it becomes poisoned data that can be injected into the backdoor. However, the poisoned data does not look any different from normal data, and in order not to change the label of the poisoned data, we decided to add perturbations only to the target label. That is, if $\mathcal{A}$ is the target label specified by the attacker, then the attacker will only add perturbation to the picture labeled as $\mathcal{A}$, so there is no need to change the label of the poisoned data to the label of the target label. Unlike a computable trigger, which sets a loss function based on the activation value, the invisible trigger enhances the loss function. Here we denote the loss function as the mean square error of the activation values of two poisoned pictures (one picture attached with a traditional trigger, another one with an invisible trigger). Its formal definition is as follows:

$$\begin{aligned} T(x) &= \big(\text{Poison G,x,t}^*\big) \\ &= \arg\min MSE(\text{sub\_G}(p), \text{sub\_G}(\tilde{s})) \\ &\quad st. \; || \, p - x \, ||_\infty < \epsilon \end{aligned} \tag{4}$$

where sub_G$(\cdot)$ is the activation value of the deep model at the penultimate layer, x is any picture of the target class, and p is its image after adding the perturbation. Attackers select an image from outside the target class and patch the trigger t* to it as $\tilde{s}$. The specific generation process is illustrated in Figure 5. When $\tilde{s}$ and p are inputted into the model, they produce activation values A1 and A2, respectively. Following the approach presented in previous work [26], we employ the standard Projected Gradient Descent (PGD) algorithm [48] to optimize Eq. (4). This process is iteratively repeated until the value of the loss function is below the threshold set by the attacker. At this stage, the effect of the trigger is distracted into an imperceptible perturbation. In most experiments in this paper, we use AlexNet as the test model. ϵ is used to control

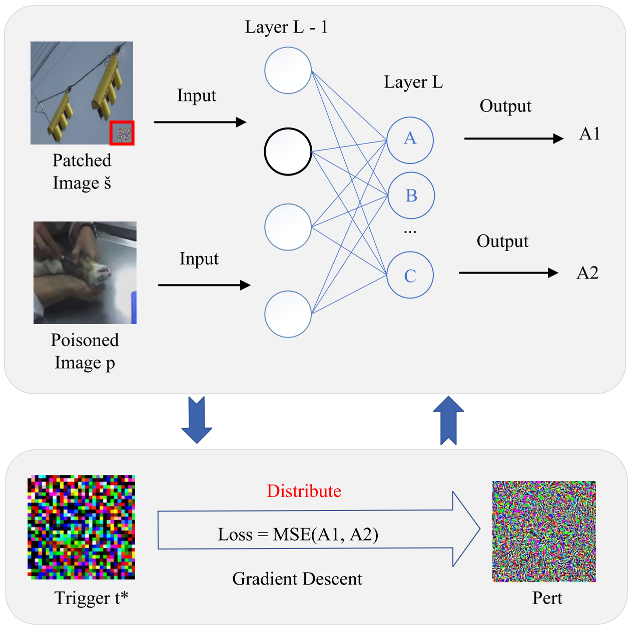


Fig. 5. Generation of invisible triggers. Where š is the picture with the trigger attached and p is the poisoned picture with the perturbation added. Using the mean squared error of their activations as a loss function, a decrease in the loss value means that the activations of š and p are getting closer, so the function of the trigger is distracted over the perturbation.

the order of magnitude of perturbations, and our experiments show that our method can use a lower ϵ than previous methods.

V. Experiment

A. Experiment Setup

Data Sets and Models: We conducted our experiments on the CIFAR-10, CIFAR-100, GTSRB and DTD. We used 4%, 10%, 20%, and 20% of each class as poisoning data, respectively. As mentioned in Section IV-C, since only the poisoned data of one class is added to the training set, for a ten-class task, the poisoning rate should be multiplied by 0.1. Meanwhile, in order to reduce the consumption of training data, the poisoned data generation process will be repeated twice. That is, we will generate two batches of similar poisoned data. In order to compare with the previous methods, we use the AlexNet model in the main experimental part, and then ResNet-18 and ResNet-34 in the later ablation experiments. These models are all pre-trained on the ImageNet dataset.

Baseline Selection: We compare this with previous backdoor attack methods, including Trojan [14], which uses reverse engineering to generate triggers, and Hidden Trigger [26], which uses hidden triggers. The reason we chose these two baselines both of them use hidden and computable triggers, which is comparable to our setting. In addition, we provide another baseline for reference with models trained on benign datasets (called standard training). At the same time, we selected STRIP [47], Neural Cleanse [45], SentiNet [49] and NAD [50] to evaluate the robustness of our method.

Attack Setup: For each attack method, we crop the size of the picture to $3 \times 224 \times 224$ and set the trigger to a 30×30 size color matrix, fixed to the bottom right corner of the image. When generating the computable trigger, we only iterated 20 rounds, and the learning rate was fixed at 1.0; In generating the hidden trigger, we performed 400 iterations with a batch size of 100, the learning rate was initially 0.01, and the learning rate decayed to 1/2 of the original after every 200 rounds. To ensure the stealthiness of perturbations, we initially set ϵ to 16. In subsequent experiments, we will test different ones to demonstrate the superiority of our proposed method. In the model fine-tuning phase, we set the learning rate to 0.001 and use the SGD algorithm to fine-tune the model. All the experiments were implemented with PyTorch and performed on a server with 2 Intel Xeon CPUs, 1 NVidia Tesla V100 GPU, 376GB RAM, and Ubuntu 18.04LTS OS.

Evaluation Metric: We use Attack Success Rate (ASR) and Benign Accuracy (ACC) to evaluate the effectiveness of different attack methods. Specifically, ASR refers to the ratio of the amount of poisoned data misclassified by the model into the label specified by the attacker to the amount of all poisoned data. ACC is defined as the accuracy of the benign sample on the test set. The higher the ASR and ACC, the better the attack method.

B. Main Result

We evaluate the task of 10-class image classification. We find that even for the same dataset, different labels have a great impact on Hidden Trigger. So, we repeated the experiment three times for the latter three datasets. We use the same experiment setting and try to poison the data on different labels. As shown in Table II, our proposed approach can successfully implant backdoors in the context of transfer learning and achieve a very high ASR. In particular, our attack method can maintain 100% ASR in most cases, however, Hidden Trigger's ASR is generally lower than 30%. Under the four data sets, the ASR of our method is nearly 70% higher than that of Hidden Trigger on average, which is because our method can be used in multi-class classification scenarios. Nearly 30% higher than Trojan, but our trigger is invisible.

In later ablation experiments, we will test the effects of different ϵ, different poisoning rates, and different model structures on our method. Based on these experimental results, we modified the model and compared it with the advanced methods to show the superiority of our method, including Invisible Trigger [30] and ATTEQ [28].

C. Ablation Experiment

Order of ϵ: The value of ϵ (defined in Eq. (4)) directly affects the stealthiness of poisoned images and thus affects the possibility of the attack being discovered by users. Figure 6 shows the original image and the effect of different ϵ values. It can be found that the perturbation in the poisoned image can be obviously seen when ϵ=16 is set by predecessors. With the continuous reduction of ϵ, the poisoning picture is gradually approaching the original picture, so the ϵ should be as small as possible. The authors of Hidden trigger [26] tested the performance of the attack at an ϵ of 8,16 and 32 and found no significant difference in the effect of the attack, so they

TABLE II
PERFORMANCE OF THREE ATTACK METHODS ON FOUR DATASETS WITH ALEXNET MODEL. FOR THE LATTER THREE DATASETS, 10 CLASSES WERE SELECTED EACH TIME AND THE EXPERIMENT WAS REPEATED THREE TIMES

	Trojan [14]			Hidden Trigger [26]			Our Method		
	Benign Model	Backdoor Model		Benign Model	Backdoor Model		Benign Model	Backdoor Model	
Datasets	ACC	ACC	ASR	ACC	ACC	ASR	ACC	ACC	ASR
CIFAR-10	81.3	82.7	100	82.1	82.2	32.2	82.5	81.6	**100**
CIFAR-100	87.9	87	100	87.7	87.1	47.3	87.4	86	**100**
	90	90.8	100	90.6	90.4	45.6	90	91.2	**100**
	90.2	90.6	100	89.4	88.8	18.5	90	90.3	**100**
GTSRB	88.1	88.1	98.7	88	87.3	46.1	88.4	88.2	**99.5**
	83	83	98.2	82	82.6	25.2	83.5	83.2	**100**
	77.5	77.3	65.6	78.1	78.3	21.5	78.9	77.1	**100**
DTD	80	78	12.7	80.2	80.7	16.1	79.7	80.2	**81.3**
	65.5	64.7	3.8	64.7	65.2	13.6	64.7	64.5	**90.5**
	64.7	65.7	16.3	67	66.7	30	65.7	66.7	**95.2**
Average	80.82	80.79	69.53	80.98	**80.93**	29.61	**81.08**	80.9	**96.65**

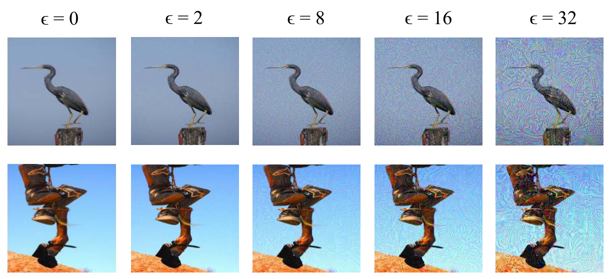
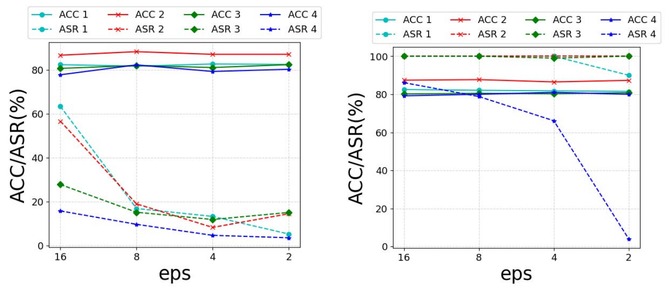
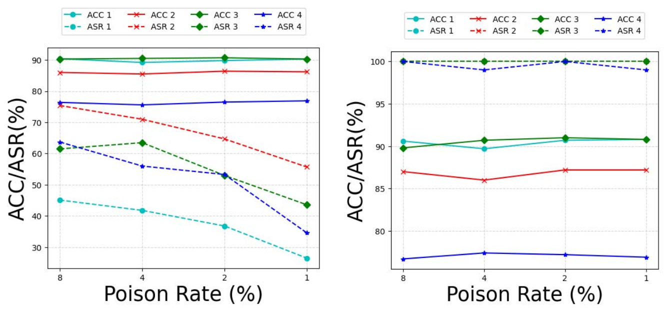


Fig. 6. When ϵ is 2,8,16,32, the poisoned picture is generated. The smaller the ϵ, the closer the poisoning image is to the original image. ϵ equals 0 representing the clean image.



(a) Hidden Trigger [26] (b) Our Method

Fig. 7. (a) and (b) are respectively the changes of ACC and ASR under different ϵ values of Hidden Trigger and Our Method. 1, 2, 3, and 4 represent different datasets, for example, ACC1 means accuracy under CIFAR-10 and ASR1 means attack success rate under CIFAR-10.



(a) Hidden Trigger [26] (b) Our Method

Fig. 8. (a) and (b) are respectively the changes of ACC and ASR under different numbers of poisoned images of Hidden Trigger and Our Method.

concluded that ϵ did not affect the experiment. But we argue this is wrong, and a decrease in ϵ usually leads to a decrease in ASR. The reason why predecessors concluded that ϵ had no effect on the experimental results was because ϵ was set too large. When ϵ=16, it is big enough to produce a suitable perturbation so that the poisoned picture has a similar output at the specified layer as the picture with the trigger added, so the ASR does not improve when the ϵ increases.

We test the impact of the value of ϵ on four datasets. As shown in Figure 7(a), where 1, 2, 3, and 4 represent four different datasets, respectively. When the ϵ is reduced to 8, there is a significant drop in ASR, and even in the first and last set of experiments, ASR is down below 10 percent. In contrast, the ASR of our method is not affected by the reduction of ϵ. After the ϵ is reduced to 2, it can still have a high attack success rate, and the ACC is not reduced, remaining above 90%, as shown in Figure 7(b). The ASR of our method is reduced significantly when ϵ is set as 2 in the fourth set of experiments, but in general, it outperforms the previous methods.

Number of Poisons: In actual attacks, the attacker prefers using less poisoned data to reduce the risk of detection. To do this, we test the performance of our method and baseline under different numbers of poisoned images. Since there are fewer pictures per label in the latter two datasets, we only conduct experiments on the CIFAR-100 dataset. To exclude the influence of data labels, we change the label of the training data and conduct four independent repeat experiments, which correspond to the four groups of data in Figure 8. As shown in Figure 8 (a), when the number of poisoned pictures decreased, the ASR of Hidden trigger show an obvious downward trend. When the poisoning rate is reduced to 1%, the ASR is generally less than 50%, and only the ASR of the second group of experiments is greater than 50%. We believe this is because the label is more susceptible to establish a connection with the trigger. Then the trigger activates the backdoor in the model, resulting in a higher ASR. As can be seen from Figure 8 (b), with the number of poisoned images decreasing, the ASR of our attack is not particularly affected, and generally remained at about 99%, only the fourth set of experiments shows a small fluctuation.

TABLE III
RESULTS OF OUR ATTACK METHOD UNDER DIFFERENT MODELS. THE NUMBERS IN PARENTHESES REPRESENT THE TARGET ACTIVATION VALUES OF THE SELECTED NEURONS. FOR EXAMPLE, "RESNET-18(10)" MEANS THAT THE TRIGGER WILL CHANGE THE ACTIVATION OF A NEURON IN THE MODEL TO 10

	ResNet-18(10)			ResNet-18(20)			ResNet-34(30)		
	Benign Model	Backdoor Model		Benign Model	Backdoor Model		Benign Model	Backdoor Model	
Datasets	ACC	ACC	ASR	ACC	ACC	ASR	ACC	ACC	ASR
CIFAR-10	80.3	81	62.8	80.3	80.7	99.8	80.6	81.5	**100**
CIFAR-100	83.2	83.3	59.3	82.6	84.8	78.8	85	86	**100**
	80.7	82.9	43.8	80.6	82.9	83.7	88.9	90.5	**100**
	88.9	89.8	30.5	89.3	89.5	81.6	88	88.4	**100**
GTSRB	81.6	84.1	66.5	82.7	84.8	99.8	83.9	84.3	**100**
	87.9	88.9	69.5	88	88.6	100	79.8	80.5	**100**
	89.1	89.6	79.8	88.8	89.4	100	78.5	79.5	**100**
DTD	67	70.5	95.8	67.2	66.7	100	75.5	75.2	**97.5**
	72	72.7	85.8	71	75.5	96.1	72.7	73.7	**100**
	62.5	61.7	82.5	61.7	62.7	100	61.5	63.2	**100**
Average	79.32	80.45	67.63	79.22	**80.56**	93.98	**79.44**	80.28	**99.75**

TABLE IV
ACCURACY (ACC) AND ATTACK SUCCESS RATE (ASR) OF DIFFERENT NUMBER OF CLASSIFICATIONS UNDER ALEXNET MODEL

	Hidden Trigger [26]				Our Method			
num_class	2	100	500	1000	2	100	500	1000
ACC	97	74	57	50.41	99	74	57.84	50.28
ASR	60	8.32	2.32	1.27	100	98.4	98.35	98.87

TABLE V
ACCURACY (ACC) AND ATTACK SUCCESS RATE (ASR) OF DIFFERENT NUMBER OF CLASSIFICATIONS UNDER RESNET-18 MODEL

	Our method		Invisible Trigger [30]		ATTEQ [28]
num_class	200	10	200	200	10
PR	0.1	3.8	10	0.1	5
ACC	79	**97**	**85**	85	91.95
ASR	**99.5**	**99.5**	**99.5**	60	88.82
PSNR	**43.1673**	27.311	27.195	27.195	-
$\mathcal{L}^{\infty}$	**2**	16	83.198	83.198	-

Different Models: To test the generalization ability of our method, we also conduct tests on ResNet-18 and ResNet-34. Experiments show that the model with residual structure is more difficult to inject backdoors, and the ASR of the model decreases significantly. As shown in Table III, ASR decreases on all four datasets, and the average ASR is also reduced to 67.63%. This is because ResNet is more robust than AlexNet. To enhance the effect of our attack, We increased the target_value in Eq. (3) from 10 to 20 and 30, respectively, and then performed the experiment again. As shown in the second and third columns of Table III, the experiment found that the ASR of ICT returned to a high level, and the ACC was not affected.

Number of Classes: According to the above experiments, it is sufficient to prove the superiority of our method in all aspects. But Hidden Trigger can only be attacked in the setting of binary classification, so we are extremely curious about the performance of our method in multi-class tasks. However, since the datasets we previously used had a maximum of only 100 labels, we conducted tests using the ImageNet dataset in this case. We are aware that this does not strictly adhere to the conditions of transfer learning, but we believe it still holds persuasive power as we only modified the weights of the final layer. We divided the experiment into two groups. We first tested our method against Hidden Trigger [26] using the AlexNet model, and the second group was compared with the current excellent methods, Invisible Trigger [30] and ATTEQ [28] with ResNet-18 model.

As shown in Table IV, with the continuous increase of the number of classes, the ACC of the model continues to decline, which is in line with common sense. However, because Hidden trigger [26] could not complete the attack under multiple classification tasks, the ASR of the model was very low, with 8.32% ASR for the task with 100 classification tasks and only 1.27% ASR for the task with 1000 classification tasks. Our approach can perform well on a variety of classification tasks, with ASR achieving 98.4%, 98.35%, and 98.87% on the 100/500/1000 class classification tasks, respectively. This is a breakthrough step up from previous work.

As shown in Table V, by comparing the second and fourth columns (num_class=200), it can be found that our method and Invisible Trigger [30] have similar ASR, both reaching 99.5%. But the ACC for our method is 6% lower, and we think there are two main reasons for this. First, although they are all ImageNet's 200 labels, the data sets are not exactly the same, which can still affect the results. Second, the Invisible Trigger method trains the entire model, and we only train the last layer of the model. When the Invisible Trigger's poisoning rate(PR) also dropped to 0.1% (num_class=200), the ASR of the poisoned model dropped sharply, to 60%. Compared to this, our ASR is 99.5% under the same poisoning rate which means that we can complete the attack successfully. When we compare ATTEQ in ten labels (num_class=10), our poisoning rate at this point is about 3.8%. Our method can maintain the ASR of 99.5%, while the ASR of ATTEQ can only achieve 88.82% under 5% PR.

We attribute the excellent performance of ICT to the use of computable triggers and the generation of unseen perturbations. A computable trigger can cause a neuron to produce a larger activation value, and an invisible perturbation can make the trigger more hidden. Both techniques make backdoor samples more covert and efficient.

D. Defense Experiment

A qualified attack method not only has excellent attack effect but also needs to be able to successfully bypass

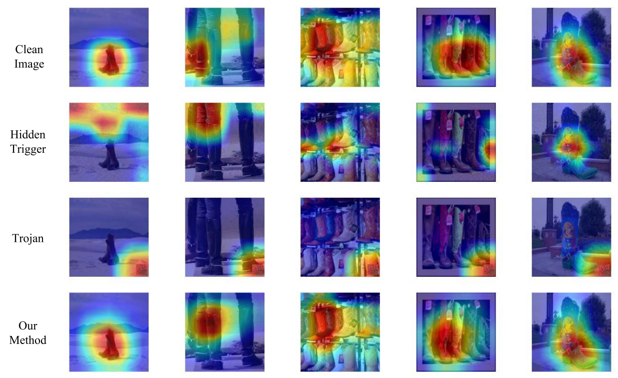


Fig. 9. The output results under the heatmap of Grad-CAM. Note that in our method and Hidden Trigger, the target class and poison class are the same class.

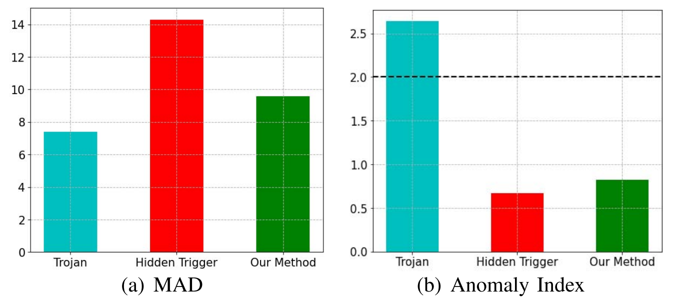


Fig. 10. (a) MAD values of the three methods, and (b) anomaly index of the three methods under the setting of threshold 2.

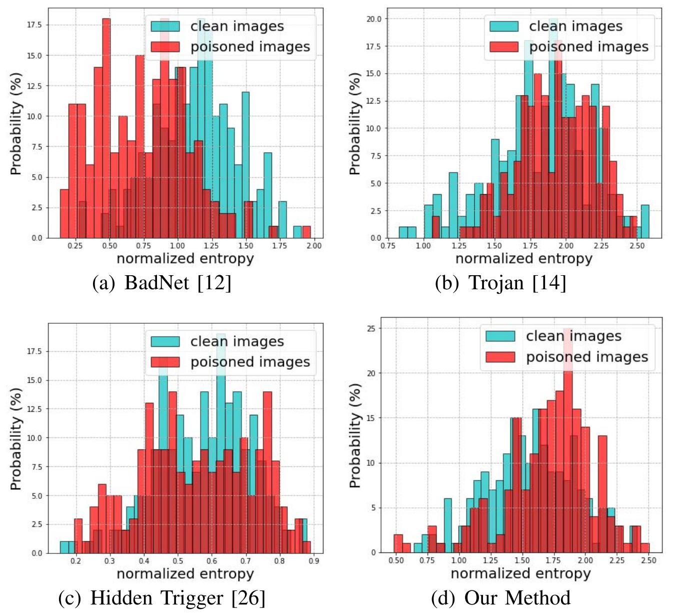


Fig. 11. The distribution of the entropy prediction results for the clean data and the poisoned data for the four attack methods.

advanced backdoor defense strategies. In this section, we test whether ICT can bypass these detection methods, including SentiNet [49], Neural Cleanse [45], STRIP [47], and NAD [50]. In addition to these detection methods, we also use two indicators used in an invisible trigger, PSRN [51] and infinite norm, to visually evaluate the concealment of poisoned images. We randomly select 10 classes for testing.

SentiNet [49]: It takes advantage of the neural network's sensitivity to attacks - when clean and poisoned images are fed into the model, different heat maps are obtained to defend against backdoor attacks. We used the common visualization tool Grad-CAM [52] to generate heat maps to help us understand which areas the model is more concerned about when poisoned data is fed into the model. Among them, the heat maps of clean pictures, trojan, and our method are obtained under the ten-class model, while the heat maps of Hidden triggers are obtained under the two-class model. As shown in Figure 9, the trigger generated by trojan was successfully identified by the Grad-CAM. Although the trigger of Hidden Trigger is invisible and cannot be identified by the heat map, there is a big gap between the heat map of the poisoned picture generated by it and that of the clean picture, which increases the possibility of the attack being discovered. In contrast, our method shows that not only the trigger is invisible and cannot be recognized by the heat map, but also there is a high similarity between the heat map of the poisoned image and that of the clean image.

Neural Cleanse [45]: For a backdoor model, assume that class A is the target class of the attacker. The order of magnitude of disturbance required for all classes to be misclassified as Class A is much smaller than that required for all classes to be classified as other classes. The defender iterates over each label, calculates the disturbance required for each label to be a target label, and then detects if there is a disturbance order of magnitude that is significantly smaller than the others (threshold 2) based on Median Absolute Deviation(MAD) [53]. If yes, the label is considered to be attacked, that is, the network has a backdoor. As shown in Figure 10 (a), Trojan has the lowest MAD value, followed by our method, and Hidden trigger has the highest. In the end, only the Trojan attack was detected, and the "Anomaly Index" for both Hidden Trigger and ICT falls below the threshold., as shown in Figure 10 (b). Our method has a higher value than the Hidden Trigger, which we think is because we use a lower order of magnitude perturbation, which makes the poisoned picture look more similar to the clean picture.

STRIP [47]: For a model with a backdoor, it is highly sensitive to the trigger, and once the trigger appears, it will be misclassified regardless of which label the original image belongs to. STRIP uses this to detect whether an input sample is a poisoned sample. It makes several copies of an input sample, overlays each copy with other labels of pictures, and the predicted distribution of these copies is used to detect backdoor samples.[3] If the input sample is a clean image, then the prediction results of the copied images should have a high entropy, because the copied images will be randomly classified into various labels. For poisoned images, the existence of triggers will cause these copied images to be mainly classified into the label specified by the attacker, so the predicted results will have a lower entropy. As shown in Figure 11, the poisoned

[3]We have repeatedly confirmed with the author that the pixel value of the overlapped image needs to be limited to the interval of [0, 1]/[0, 255].

TABLE VI
ACCURACY AND ATTACK SUCCESS RATE (ASR) FOLLOWING THE UTILIZATION OF NAD ON THREE DATASETS

Dataset	Defence w/o	Our Method	
		ACC	ASR
CIFAR-10	Original	80.7	99.8
	NAD	10.2	1.7
CIFAR-100	Original	89.5	81.6
	NAD	15.9	22.83
GTSRB	Original	89.4	100
	NAD	24.6	6.8

samples of BadNet were successfully distinguished by STRIP, because the entropy distribution of the predicted results of these poisoned samples was obviously different from that of the clean samples. For Hidden Trigger, there was a lot of overlap between poisoned data and clean data. We think the reason may be that Hidden Trigger itself is only a binary classification model, so the entropy of the prediction result will not change greatly. For Trojan and our method, the entropy distribution of the predicted results of poisoned and clean samples is similar, so our method can successfully bypass this detection strategy.

NAD [50]: In addition to defensive testing on the input side, we further investigate the efficiency of the NAD, a method for eliminating backdoor, across three distinct datasets, as shown in Table VI. We find that although NAD can reduce the ASR of our poisoning model, the accuracy of the model also shows a very significant decline, from 80% to about 10% (on the ResNet-18 model), which is unacceptable. This is because traditional backdoors are embedded throughout the entire model. When performing knowledge distillation, the adjustment of each parameter will not be too large, thus ensuring the stability of ACC. However, for our method, the backdoor focuses on the backdoor layers of the model. If we want to erase the backdoor, the parameters of the second half of the model will be changed across orders of magnitude, resulting in the model also performing poorly on the clean data set. Therefore, it is reasonable to believe that our method bypasses NAD detection.

VI. CONCLUSION

In this paper, we show that static triggers are difficult to inject backdoors into transfer learning, with only very low ASR. To this end, we propose an efficient and covered backdoor attack method - an invisible computable Trigger Backdoor attack (ICT). In order to inject backdoors into the last layer, we propose to use a neuron selection algorithm to generate more targeted computable triggers. It can be more easily perceived by the model, thus reducing the difficulty of injecting backdoors. In order to be invisible, we use the PGD algorithm to carefully calculate a tiny disturbance, so that after it is added to the image data, the change can not be detected by people or common defense methods. We have conducted extensive experiments to verify the effectiveness of our approach against attacks and robustness against defense methods. The work in this paper contributes to the study of invisible triggers of clean labels in transfer learning. It is also hoped that the work of this paper can provide new ideas and ideas for the defense of backdoor attacks. In future work, we will try to evaluate our approach to more real-world data in the physical world.

ACKNOWLEDGMENT

The author sincerely thanks Dr. Zhen Li from the School of Cyberspace Security, Huazhong University of Science and Technology. She is thanked for her comments and suggestions on the first draft of this article.

REFERENCES

[1] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2015, pp. 815–823.
[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 779–788.
[3] L. Qi, Y. Zhang, and T. Liu, "Bidirectional transformer with absolute-position aware relative position encoding for encoding sentences," *Front. Comput. Sci.*, vol. 17, no. 1, 2023, Art. no. 171301.
[4] H. Zhao, W. Min, J. Xu, Q. Wang, Y. Zou, and Q. Fu, "Scene-adaptive crowd counting method based on meta learning with dual-input network DMNet," *Front. Comput. Sci.*, vol. 17, no. 1, 2023, Art. no. 171304.
[5] Z. Guo, J. Chen, T. He, W. Wang, H. Abbas, and Z. Lv, "DS-CNN: Dual-stream convolutional neural networks-based heart sound classification for wearable devices," *IEEE Trans. Consum. Electron.*, vol. 69, no. 4, pp. 1186–1194, Nov. 2023.
[6] P. Chanak and I. Banerjee, "Congestion free routing mechanism for IoT-enabled wireless sensor networks for smart Healthcare applications," *IEEE Trans. Consum. Electron.*, vol. 66, no. 3, pp. 223–232, Aug. 2020.
[7] X. Zhou et al., "Hierarchical federated learning with social context clustering-based participant selection for Internet of Medical Things applications," *IEEE Trans. Comput. Social Syst.*, vol. 10, no. 4, pp. 1742–1751, Aug. 2023.
[8] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 199–210, Feb. 2011.
[9] X. Zhou et al., "Spatial–temporal federated transfer learning with multi-sensor data fusion for cooperative positioning," *Inf. Fusion*, vol. 105, May 2024, Art. no. 102182.
[10] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2015, pp. 427–436.
[11] Y. Chen, X. Gong, Q. Wang, X. Di, and H. Huang, "Backdoor attacks and defenses for deep neural networks in outsourced cloud environments," *IEEE Netw.*, vol. 34, no. 5, pp. 141–147, Sep./Oct. 2020.
[12] T. Gu, B. Dolan-Gavitt, and S. Garg, "BadNets: Identifying vulnerabilities in the machine learning model supply chain," 2017, *arXiv:1708.06733*.
[13] X. Qi, J. Zhu, C. Xie, and Y. Yang, "Subnet replacement: Deployment-stage backdoor attack against deep neural networks in gray-box setting," 2021, *arXiv:2107.07240*.
[14] Y. Liu et al., "Trojaning attack on neural networks," in *Proc. 25th Annu. Netw. Distrib. System Secur. Symp. (NDSS)*, 2018, pp. 1–15.
[15] J. Xu, K. Xue, Q. Yang, and P. Hong, "PSAP: Pseudonym-based secure authentication protocol for NFC applications," *IEEE Trans. Consum. Electron.*, vol. 64, no. 1, pp. 83–91, Feb. 2018.
[16] D. Sethia, D. Gupta, and H. Saran, "NFC secure element-based mutual authentication and attestation for IoT access," *IEEE Trans. Consum. Electron.*, vol. 64, no. 4, pp. 470–479, Nov. 2018.
[17] P. Paikrao, S. Routray, A. Mukherjee, A. R. Khan, and R. Vohnout, "Consumer personalized gesture recognition in UAV-based industry 5.0 applications," *IEEE Trans. Consum. Electron.*, vol. 69, no. 4, pp. 842–849, Nov. 2023.
[18] F. A. Yerlikaya and Ş. Bahtiyar, "Data poisoning attacks against machine learning algorithms," *Expert Syst. Appl.*, vol. 208, Nov. 2022, Art. no. 118101.
[19] S. Alfeld, X. Zhu, and P. Barford, "Data poisoning attacks against autoregressive models," in *Proc. AAAI Conf. Artif. Intell. (AAAI)*, 2016, pp. 1452–1458.

[20] A. Shafahi et al., “Poison frogs! targeted clean-label poisoning attacks on neural networks,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 31, 2018, pp. 6106–6116.
[21] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, “Targeted backdoor attacks on deep learning systems using data poisoning,” 2017, *arXiv:1712.05526*.
[22] M. Barni, K. Kallas, and B. Tondi, “A new backdoor attack in CNNs by training set corruption without label poisoning,” in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, 2019, pp. 101–105.
[23] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” 2014, *arXiv:1412.6572*.
[24] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, “Practical black-box attacks against machine learning,” in *Proc. ACM Asia Conf. Comput. Commun. Secur. (CCS)*, 2017, pp. 506–519.
[25] Y. Li, T. Zhai, B. Wu, Y. Jiang, Z. Li, and S. Xia, “Rethinking the trigger of backdoor attack,” 2020, *arXiv:2004.04692*.
[26] A. Saha, A. Subramanya, and H. Pirsiavash, “Hidden trigger backdoor attacks,” in *Proc. AAAI Conf. Artif. Intell. (AAAI)*, 2020, pp. 11957–11965.
[27] K. Doan, Y. Lao, W. Zhao, and P. Li, “Lira: Learnable, imperceptible and robust backdoor attacks,” in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2021, pp. 11966–11976.
[28] X. Gong, Y. Chen, J. Dong, and Q. Wang, “ATTEQ-NN: Attention-based QoE-aware evasive backdoor attacks,” in *Proc. Annu. Netw. Distrib. System Secur. Symp. (NDSS)*, 2022, pp. 1–18.
[29] E. Bagdasaryan and V. Shmatikov, “Blind backdoors in deep learning models,” in *Proc. 30th USENIX Secur. Symp. (USENIX Security)*, 2021, pp. 1505–1521.
[30] Y. Li, Y. Li, B. Wu, L. Li, R. He, and S. Lyu, “Invisible backdoor attack with sample-specific triggers,” in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2021, pp. 16463–16472.
[31] E. Wenger, J. Passananti, A. N. Bhagoji, Y. Yao, H. Zheng, and B. Y. Zhao, “Backdoor attacks against deep learning systems in the physical world,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, pp. 6206–6215.
[32] P. Xia, Z. Li, W. Zhang, and B. Li, “Data-efficient backdoor attacks,” 2022, *arXiv:2204.12281*.
[33] S. Wang, S. Nepal, C. Rudolph, M. Grobler, S. Chen, and T. Chen, “Backdoor attacks against transfer learning with pre-trained deep learning models,” *IEEE Trans. Services Comput.*, vol. 15, no. 3, pp. 1526–1539, Jun. 2020.
[34] C. Xie, K. Huang, P.-Y. Chen, and B. Li, “DBA: Distributed backdoor attacks against federated learning,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–19.
[35] P. Lv et al., “A data-free backdoor injection approach in neural networks,” in *Proc. 32nd USENIX Secur. Symp. (USENIX Security)*, 2023, pp. 2671–2688.
[36] A. Nguyen and A. Tran, “WaNet–imperceptible warping-based backdoor attack,” 2021, *arXiv:2102.10369*.
[37] X. Gong, Z. Wang, Y. Chen, M. Xue, Q. Wang, and C. Shen, “Kaleidoscope: Physical backdoor attacks against deep neural networks with RGB filters,” *IEEE Trans. Depend. Secure Comput.*, vol. 20, no. 6, pp. 4993–5004, Dec. 2023.
[38] K. Chen, X. Lou, G. Xu, J. Li, and T. Zhang, “Clean-image backdoor: Attacking multi-label models with poisoned labels only,” in *Proc. 11th Int. Conf. Learn. Represent. (ICLR)*, 2023, pp. 1–18.
[39] A. Turner, D. Tsipras, and A. Madry, “Clean-label backdoor attacks,” in *Proc. ICLR*, 2018, pp. 1–20.
[40] Y. Wang, K. Chen, Y.-a. Tan, S. Huang, W. Ma, and Y. Li, “Stealthy and flexible trojan in deep learning framework,” *IEEE Trans. Depend. Secure Comput.*, vol. 20, no. 3, pp. 1789–1798, Jun. 2023.
[41] 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.
[42] Y. Li et al., “Reconstructive neuron pruning for backdoor defense,” in *Proc. 40th Int. Conf. Mach. Learn. (ICML)*, 2023, pp. 1–18.
[43] J. Xia, T. Wang, J. Ding, X. Wei, and M. Chen, “Eliminating Backdoor triggers for deep neural networks using attention relation graph distillation,” in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2022, pp. 1481–1487.
[44] K. Liu, B. Dolan-Gavitt, and S. Garg, “Fine-pruning: Defending against backdooring attacks on deep neural networks,” in *Proc. Int. Symp. Res. Attacks, Intrusions, Defenses*, 2018, pp. 273–294.
[45] B. Wang et al., “Neural cleanse: Identifying and mitigating backdoor attacks in neural networks,” in *Proc. IEEE Symp. Security Privacy (SP)*, 2019, pp. 707–723.
[46] W. Chen, B. Wu, and H. Wang, “Effective backdoor defense by exploiting sensitivity of poisoned samples,” in *Proc. 36th Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2022, pp. 1–11.
[47] Y. Gao, C. Xu, D. Wang, S. Chen, D. C. Ranasinghe, and S. Nepal, “Strip: A defence against trojan attacks on deep neural networks,” in *Proc. 35th Annu. Comput. Secur. Appl. Conf. (ACSAC)*, 2019, pp. 113–125.
[48] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” 2017, *arXiv:1706.06083*.
[49] E. Chou, F. Tramer, and G. Pellegrino, “Sentinet: Detecting localized universal attacks against deep learning systems,” in *Proc. IEEE Security Privacy Workshops (SPW)*, 2020, pp. 48–54.
[50] Y. Li, N. Koren, L. Lyu, X. Lyu, B. Li, and X. Ma, “Neural attention distillation: Erasing backdoor triggers from deep neural networks,” 2021, *arXiv:2101.05930*.
[51] Q. Huynh-Thu and M. Ghanbari, “Scope of validity of PSNR in image/video quality assessment,” *Electron. Lett.*, vol. 44, no. 13, pp. 800–801, 2008.
[52] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-CAM: Visual explanations from deep networks via gradient-based localization,” in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2017, pp. 618–626.
[53] F. R. Hampel, “The influence curve and its role in robust estimation,” *J. Am. Statist. Assoc.*, vol. 69, no. 346, pp. 383–393, 1974.

Xiang Chen received the B.S. degree in computer science and technology from the Nanyang Institute of Technology, Nanyang, China, in 2022.

He is currently pursuing the master’s degree in cyberspace security with Zhengzhou University, Zhengzhou, China. His research interests include neural networks, AI security, and accelerated computing.

Bo Liu received the M.S. degree in computer science and engineering from Zhengzhou University, Zhengzhou, and the Ph.D. degree in computer architecture from the Huazhong University of Science and Technology, Wuhan.

She is currently a Lecturer with Zhengzhou University. Her research interests focus on deep learning, AI security, and AI accelerator design. She has served as a Reviewer for top conferences or journals, such as IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS and *ACM Transactions on Architecture and Code Optimization*.

Shaofeng Zhao received the B.S. and M.S. degrees in computer science and engineering from Zhengzhou University, Zhengzhou, and the Ph.D. degree in computer architecture from the Huazhong University of Science and Technology, Wuhan.

He is currently a Lecturer with the Henan University of Economics and Law. His current research interests focus on Internet security and nonvolatile memory systems.

Ming Liu received the Ph.D. degree from Monash University in 2019. He joined the School of IT, Deakin University as a Research Fellow later and became an Assistant Professor in 2022. His main research area includes natural language processing, machine learning, and continual learning. He is a member of the China Computer Federation Technical Committee of Natural Language Processing and a Program Member of the Association of Computational Linguistics.

Hui Xu received the Ph.D. degree in computer architecture from the Huazhong University of Science and Technology, Wuhan.

She is currently an Assistant Researcher with the China Academy of Industrial Internet, Beijing, China. Her research interests focus on deep learning, data privacy, and industrial Internet security. She has served as a Reviewer for international conferences or journals, such as HPCC, ICPADS, and *Cybersecurity*.

Zhanbo Li received the M.S. degree from Sun Yat-sen University in 1993. He has participated in more than ten provincial science and technology research projects and has undertaken several network engineering planning and design, provincial and municipal information construction planning, and software development projects, with rich experience in system design, development, and engineering implementation. His research interests include network security, e-government, and urban informatization.

Zhigao Zheng (Member, IEEE) received the Ph.D. degree from the Huazhong University of Science and Technology, Wuhan, China. He published more than 20 peer-reviewed publications (such as the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, and IEEE TRANSACTIONS ON COMPUTERS). He joins research projects from various governmental and industrial organizations, such as the National Science Foundation of China, the Ministry of Science and Technology, and the Ministry of Education. His current research interests focus on cloud computing, big data processing, and AI systems. He is also an Editorial Board Member of some high quality journals, such as the IEEE TRANSACTIONS ON CONSUMER ELECTRONICS and *Mobile Networks and Applications*. He is a PC member of several TOP conferences, such as TheWebConf (formally WWW) and NeurIPS, and the Vice PC Chair of CPSCom 2023. He is an Executive Committee Member of the Technical Committee on Distributed Computing and Systems and Embedded Systems of CCF. He is a member of ACM and CCF.