A Pragmatic Label-Specific Backdoor Attack

Yu Wang(✉), Haomiao Yang, Jiasheng Li, and Mengyu Ge

School of Computer Science and Engineering (School of Cyberspace Security),University of Electronic Science and Technology of China, Chengdu 610000, China
202021081812@std.uestc.edu.cn

Abstract. Backdoor attacks, as an insidious security threat to deep neural networks (DNNs), are adept at injecting triggers into DNNs. A malicious attacker can create a link between the customized trigger and the targeted label, such that the prediction of the poisoned model will be manipulated if the input contains the predetermined trigger. However, most existing backdoor attacks define an obvious trigger (eg: conspicuous pigment block) and need to modify the poisoned images' label, causing these images seems to be labeled incorrectly, which leads to these images can not pass human inspection. In addition, the design of the trigger always needs the information of the entire training data set, an extremely stringent experiment setting. These settings above remarkably restrict the practicality of backdoor attacks in the real world.

In our paper, the proposed algorithm effectively solves these restrictions of existing backdoor attack. Our Label-Specific backdoor attack can design a unique trigger for each label, while just accessing the images of the target label. The victim model trained on our poisoned training dataset will maliciously output attacker-manipulated predictions while the poisoned model is activated by the trigger. Meanwhile victim model still maintains a good performance confronting benign data samples. Hence, our proposed backdoor attack approach must be more practical.

Keywords: Deep neural network · Backdoor attack · Label-specific trigger

1 Introduction

Deep learning has shown increasing performance advantages in many real-world tasks with the development of computing resources and enrichment of training data [1,2], so many industries are more willing to make deep neural networks the foundational mean for solving practical challenges. Therefore, the security issue of deep neural networks is more remarkable than before. In common, a security system needs to consider three main aspects: availability, confidentiality, and integrity, where availability and confidentiality refer to the possible software vulnerabilities in the deep learning system framework as well as its dependent libraries, such as overflow attacks and DDos attacks, and the possibility

E. Ahene and F. Li (Eds.): FCS 2022, CCIS 1726, pp. 149–162, 2022.
https://doi.org/10.1007/978-981-19-8445-7_10

of the model's own parameters and data being stolen by an attacker during the inference phase [3,4], respectively. As for integrity-related security issues, they mainly refer to the interference in the process of model learning and prediction, which makes the output results not conform to the normal performance of the model, which can be divided into two categories: Evasion Attacks [5] and Data Poisoning Attacks [6,7]. Among them, data poisoning attacks occur during the model training phase, where attackers exploit vulnerabilities in data collection to manipulate training data so that downstream machine learning models contain exploitable behavior. Some attacks degrade the inference capability of the entire sample [8], while targeted data poisoning attacks induce false outputs on specific target samples [9].

Backdoor attacks, a classic representation of targeted data poisoning attacks, compromise the integrity of a model by poisoning a portion of the training data during the training phase of the model. Since deep learning systems often require large amounts of training data to support model training and high-quality training sets are expensive to collect and collate, practitioners seek larger and larger data sets to train their data-hungry models. Due to the surge in demand for training data and increased accessibility through the web, data curation efforts are also ongoing and the process of data curation is increasingly automated, which allows malicious attackers to control or poison training datasets.

A backdoor attack manifests itself in the form of a poisoned neural network model that outputs the same results as a normal model when confronted with benign data inputs but changes its output to a target label specified by the attacker when confronted with data with a trigger set by the attacker. Since a successful backdoor attack does not lead to any degradation in the overall task performance of the model, and the malicious attacker is free to select target labels and triggers, it is difficult to detect the backdoor attack. However, partial backdoor attack defense and backdoor removal techniques have been proposed and have yielded good results for partial backdoor attacks. So the concealment of backdoor attacks has received great questions.

A series of approaches to generating triggers have occurred in the existing works on backdoor attacks. The initial mean of backdoor attacks is to directly inject randomly generated triggers into the training data, so that the model trained on that data relies on triggers for inference, while the threat model of such attack methods usually includes label flipping, in which the poisoned images are often easily inspected because they belong to the incorrect class and contain a trigger that is placed in a visible location. In addition, there is a class of work that investigates invisible triggers, and to provide stealth in backdoor attacks, they make the trigger unobtrusive to the naked eye through different backdoor generation strategies, but the label-image inconsistency problem is still left behind. Subsequent work related to clean-label backdoor attacks emerged, which ensure that the labels of the poisoned data appear to human reviewers to be consistent with the category of the image itself.

In order to address these severe shortcomings and limitations in the above-mentioned backdoor attacks, we proposes a new backdoor attack scheme that is

more likely to be implemented in practical scenarios, and our contribution can be mainly concluded in the following three points:

- **Clean-label backdoor attack:** In our scheme, the user does not modify the label of the poisoned image as in a standard backdoor attack, so that the attack can effectively avoid the defense of human inspection and does not need to access data of different labels.
- **Less data information:** Compared with other backdoor attack methods and existing clean-label attack methods, the malicious attacker in our attack method needs extremely little information, only the data of the target label is indispensable in the trigger generation phase. The information about the structure of the victim model, and training process settings is not needed at all.
- **Low data poisoning rate:** Our attack method, as one of the clean-label backdoor attack, only need to poison a very small portion of the target label to achieve a very good attack effect. Concretely, we only need 0.5% of the overall dataset to achieve close to 100% of the attack effect, e.g.: 250 pictures in Cifar-10 dataset containing a total of 50000 pieces of images.

2 Related Work

2.1 Image Classification

Image Classification is a fundamental task that attempts to comprehend an entire image as a whole. The goal is to classify the image by assigning it to a specific label. Typically, Image Classification refers to images in which only one object appears and is analyzed. The backdoor attack scenarios considered in this thesis all take place in the image classification task, where the inputs $x \in X$ and label $y \in Y$ are mainly included, where the combination of (x, y) is the samples in the dataset (X, Y). The model can then be defined as the equation $f_\theta : X \rightarrow Y$, θ represents the parameters of the model (including the weights and bias in the neural network). To evaluate the performance of the model, a loss function $L(x, y, \theta)$ is often defined. The objective of the image classification task is to improve the accuracy of the model for image classification, which is equivalent to minimizing the loss function. Therefore, the model parameter θ^* corresponding to the smallest value of the loss function is what we want.

$$\theta^* = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} L(x, y, \theta)$$

Here the minimization of the loss function is usually done by using various gradient descent strategies, such as SGD, Adam, etc.

2.2 Data Poisoning Attack

Data poisoning attacks [10] mainly refers to a kind of attack where an attacker manipulates training samples or model architecture, aiming at impairing the

functionality of the model such as wrong predictions of data from all classes (an indiscriminate attack) or misclassification of subsequent input data associated with a specific label (a targeted attack).

As for untargeted data poisoning attacks, the goal of such attacks is to degrade the performance of a model so that it can no longer provide services. From the previous work presented [10], such attacks first act on simple machine learning models, like SVMs [8], logistic regression models [11], and linear classifiers [12], and then use methods like gradient descent strategy to reduce the overall effectiveness of the model. This series of attacks often render the final poisoned model incapable of completing its task and functionally paralyzed, which is discernible to anyone using it. As a result, this attack must be disposable, after which the poisoned model is quickly taken offline and modified. Moreover, the prerequisites required for the success of such an attack are stringent, which compel the attacker to have control over most of the training data. However, another collection of data poisoning attacks, targeting certain labels instead of the entire dataset, has substantial existing works and they can be divided into several categories. The following sections will specifically introduce Standard Backdoor Attack, Invisible Backdoor Attack, and Clean-Label Backdoor Attack and analyze their characteristics.

2.3 Backdoor Attack

Due to the uniqueness of backdoor attacks in terms of attack purpose, the comprehensive performance of backdoor attacks can be concluded with three aspects: attack success rate (ASR), attack stealthiness, and the model performance on the benign dataset. More specifically, the attack stealthiness can be assessed from two perspectives: whether the trigger is conspicuous and whether the label of the poisoned data is modified.

As shown in Table 1, we summarize the characteristics of some classical backdoor attack algorithms here, mainly comparing the adversary capabilities assumed by different methods and the visibility of generated triggers. 'Obvious trigger' refers to whether its trigger is visible, and 'label flipping' refers to whether an attacker needs label poisoning operations. The “low poisoning rate" also refers to whether the poisoning rate of attackers is low as well as the “entire dataset information" indicates whether the attacker needs information on the complete training dataset.

Table 1. Backdoor attack features

	BadNets [13]	SSBA [14]	HTBA [15]	Ours
Obvious Trigger	✓	×	×	×
Label Flipping	✓	×	×	×
Low Poisoning Rate	✓	✓	✓	×
Entire dataset information	✓	✓	✓	×

Standard Backdoor Attack. The original backdoor attack scheme against DNNs is BadNets [13], the representative of Standard Backdoor Attack. In this attack scheme, the attacker would specify an attack target label t, then implant conspicuous triggers in parts of the image data with non-target labels, after which modify the labels of these images to target label t. The combination of poisoned data, image data injected with trigger, and benign dataset will be fed into the victim model, which will be successfully injected with the backdoor after a normal training procedure. Meanwhile, there are many similar backdoor attacks, [16] uses the combination of sample features belonging to non-target labels in the training dataset to generate a trigger, [17] generates a separate trigger for each input image, and [18] reduce the possibility of detection by backdoor detection techniques through adversarial regularization. In general, all these approaches require conspicuous triggers and label flipping to conduct the connections between target label and trigger, which greatly reduces the stealthiness of backdoor attacks. In addition, backdoor attack methods that require label flipping operations always assume that the attacker has information about the complete dataset and can make changes to the entire dataset, which is unrealistic in practical scenarios.

Invisible Backdoor Attack. In order to improve the conspicuousness of backdoor attacks, [19] mentions the conspicuousness of the trigger for the first time, specifically that the trigger set in a backdoor attack should not be too visible, or the image data with the trigger will be easily recognized in front of the human inspector. Therefore, [19] proposes a backdoor attack method based on a blended strategy. Meanwhile, [20] used image-scaling attack to hide the trigger and SSBA [21] proposed a sample-specific backdoor attack method. Although such trigger generation approaches can help the backdooring attacks evade human inspector detection, these methods still require the attacker to do label-flipping operations on non-target data and to maintain information about the entire dataset.

Clean-Label Backdoor Attack. The most distinctive feature of this type of backdoor attack is just like its name itself, 'clean-label' means that there is no label poisoning in the method. In this series of approaches, even the image data provided by the malicious attacker remains consistent with their semantic label, and the label-flipping operation no longer exists in the attack procedures. The HTBA [15] minimizes the distance between the non-target label data and the target-label data in the feature space by using the PGD algorithm to add the trigger so that the victim model will misclassify the data with the trigger. So at the same time, a feature extractor obtained on the complete training dataset has to be known to the attacker. Similar work has been done on [21–23], all of these attack methods assume that the attacker has access to the entire dataset information.

3 Method

We first briefly introduce the threat model and the problem setup parts and then show the detailed procedure of the proposed method.

3.1 Threat Model

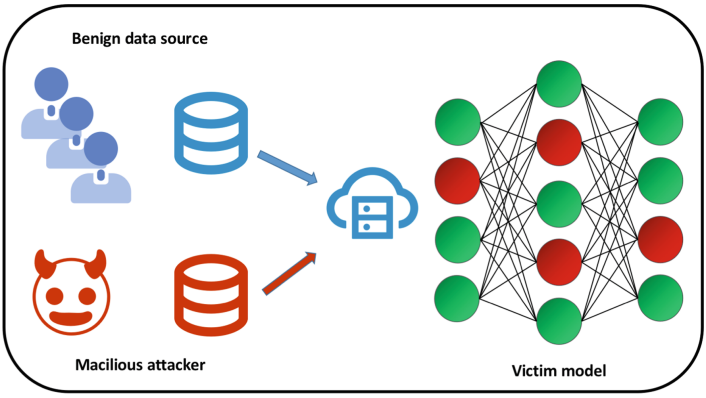

Fig. 1. Threat model

We consider a practical scenario: a model owner would like to use the data collected from multiple sources to assist in training their model and only the data collection phase is open to the public. Hence throughout all model training phases, any malicious attacker can only interfere with the model during the data collection phase. Therefore, we will define two main parts in Fig. 1: the attacker and the victim model, where the attacker is one of the many data providers and the victim model is the model with the backdoor trigger. As a result, the attacker can control only a limited part of the training dataset while having no information about the data from other providers.

The essential goal of the attacker is to successfully implant the trigger of the backdoor attack into the victim model. More concretely, the attacker wants to manipulate the model's output when confronting data patched with the trigger, whose output label will be the attacker's pre-specified label instead of the original label of the image data, and ensure the model's performance on clean data at the same time (classification accuracy).

The attacker's knowledge and capabilities: We divide the practical scenario into three phases: data collection phase, model training phase, and model deployment phase. In the data collection phase, the attacker is one of the many data providers. We assume that the attacker is responsible for providing data for a certain label and has only limited knowledge about the model training task, not the specific information about the whole training task, such as the number of

labels and the meaning of each label for the multi-category task. However, the attacker can find some relevant additional publicly available datasets to assist in training the alternative model based on the basic information available, but we assume here that there is no overlap of sample labels between the publicly available alternative datasets and the training dataset. As for the attacker's manipulation of the poisoned data, our assumptions are more stringent. The attacker is required not to do label flipping (poisoning) on the data, and the percentage of poisoned data is also limited to a small range. Based on previous work settings, the attacker to modify the data range is also limited in the l norm range, making the trigger to add will not cause a sharp change of the original image.

In the training phase of the model, the attacker has no information about the parameters used in the training phase of the victim model and the structure of the model, and cannot directly modify the model parameters of the model training to control the model training process. In the deployment phase of the model, it is also impossible to do any modification to the trained model, but in this phase, it is possible to control the output of the model by putting data with a trigger into the victim model. This adversary capability is common and feasible in realistic scenarios and is weaker than the adversary capability mentioned in the existing work [13].

3.2 Proposed Attack

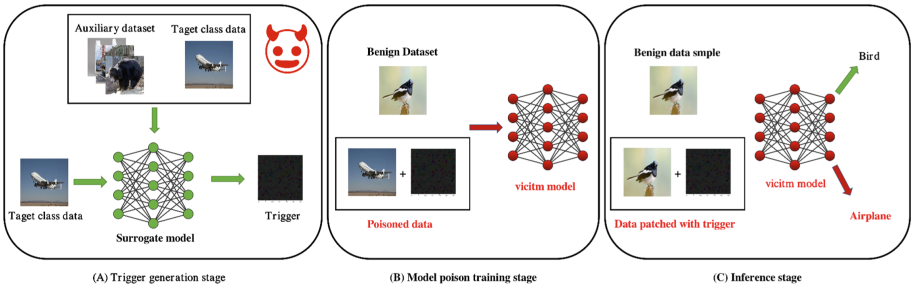

Fig. 2. Backdoor attack workflow

In this section, we will mainly introduce our attack method with a complete illustration(label-specific backdoor attack). The exact attack workflow will be divided into three stages: trigger generation stage, model poisoning stage, and inference stage (model deployment stage) (Fig. 2).

Trigger Generation Stage: At this stage, our main goal is to generate the target-label-specific trigger and the specific algorithm for this part is shown in ALGORITHM 1. Because of the characteristics of the clean-label backdoor and the limitation of our ability to the adversary, our method is different from

Algorithm 1. Trigger generation algorithm

Input: Auxiliary dataset D_{aux}, targeted label data D_t, target label t, *limited pixal range of trigger* ρ, *total train epoch* N_{train}, *trigger generation epoch* $N_{trigger}$;
Output*: Pretrained surrogate model* $f_{surrogate}$, *Trigger* τ;
Begin:
1. Surrogate dataset $D_{surrogate} \in (X, Y) \leftarrow$ *combine the Auxiliary dataset* D_{aux} *and targeted label data* D_t
2. $f_{surrogate} \leftarrow$ *random initialization*
For $i \in range(0, N_{train})$: ***do***
  $loss = CrossEntropy(f_{surrogate}(X), Y)$
  compute gradient and update $f_{surrogate}$
For*i* $\in range(0, N_{trigger})$: ***do***
  $loss = CrossEntropy(f_{surrogate}(\rho), t)$
  compute gradient and update trigger image ρ
Output *trigger image* ρ *and Pretrained surrogate model* $f_{surrogate}$

the previous work, which directly establishes a connection between the trigger and the target label through label reversal operation. Rather, by reinforcing the connection between the victim Model and the characteristics of the target label data itself.

Therefore, we first need an extractor that can extract the features of the target label data, and a well-trained target classification model is an ideal feature extractor because a high-performance classification model must be a good feature extractor. However, due to the attacker can not access the complete training dataset, only target label data sets cannot complete the feature extractor (image classification model), so the attacker needs to find the appropriate auxiliary dataset to facilitate the target label data to train a surrogate feature extractor (image classification model).

Then, the surrogate model is used to extract the features of the target label data. Here, we first initialize a noise image and then use the ADAM optimizer to continuously update the noise image iteratively to improve the confidence of the prediction of the noise image as the target label. In addition, the noise image after each round of update is scaled to ensure that its maximum pixel value does not exceed the established range, and to ensure that the poisoned data will not change too much from the original image. Otherwise, the model will be overfitted to the trigger image on the target label, which will not only affect the overall performance of the model, especially the accuracy of the clean data of the target label but also lead to the reduction of the concealability of the backdoor attack.

After several rounds of iterative optimization, the noise image finally obtained is the trigger corresponding to our ideal target label, because it will be superimposed on any image to make it possess certain characteristics of the target label. The subsequent Model poisoning training phase will have Victim Model strengthen the connection between the trigger and the target label.

Model Poisoning Stage: After the completion of the trigger generation stage, a very small portion of the target label dataset can be selected with a trigger

as the poisoned data. It should be noted here that we only add a trigger to the image data but do not change any label corresponding to the data, and then fuse it together with the benign dataset as the poisoned dataset. The subsequent model training process is the same as the normal machine learning model training process, and the attacker does not exercise any control over the training process. The final trained model is the poisoned model with the backdoor implanted.

Inference Stage: This stage is the stage when the backdoor attack works after the actual deployment of the victimization model on the ground. The attacker can add the triggers generated in the previous stage to any non-target labeled data to generate test data with triggers, which feeds the victim model and outputs the target labels specified by the attacker.

4 Experiment

4.1 Experiment Settings

This experiment mainly uses 2 GTX3090 Ti GPUs, 12th Gen Intel(R) Core(TM) i9-12900K and 128G RAM as the hardware base. As for the machine learning framework, we use Pytorch for all experiments, including comparison (ablation) experiments.

Datasets and Models: The experiments here mainly focus on the classification task of image datasets. We use CIFAR-10 dataset containing 10 classes [24], and Tiny ImageNet dataset containing 200 classes, a subset of ImageNet dataset [25], which are both classical datasets for image classification tasks. CIFAR-10 dataset is used as the training dataset of the victim model. Tiny Imagenet is a helper dataset used by attackers to generate Surrogate Models. It is worth noting that the two datasets do not coincide in the setting of labels, which is consistent with our setting in the threat model.

As for the model selection, all experiments have been conducted on Resnet-18 and GoogleNet models, and the backdoor attack effect is evaluated when the structures of the surrogate model and victim model are inconsistent.

Attack Setup: Our experiment is mainly divided into the trigger generation stage, poison model stage, and Inference stage. In Trigger Generation Stage, our experimental process is mainly to train surrogate models and generate triggers for target label datasets. Therefore, in this part, we specify that the target label is Label 0 in the CIFAR-10 dataset – the size of the airplane and the corresponding trigger, which is equal to the size of the original image(3×32x32). The 5000 images corresponding to the target label Airplane were then merged with the auxiliary dataset Tiny ImageNet into the surrogate dataset, which was used to train the surrogate Model.

In the subsequent model poisoning stage, the data poison rate γ in backdoor attack varies in different experiments. More specifically, we set the highest data poisoning rate as $\gamma = 0.5\%$, 200 training rounds, and the way of implanting

backdoor as directly stacking the trigger with the original image. Meanwhile, we also tested the experimental effects of different data poisoning rates.

Here we define three datasets: poison training data set, poison test data set, and clean test dataset. The poison training dataset is used to implant a backdoor into the victim Model. Therefore, 250 images randomly selected from data belonging to the airplane label are superimposed with trigger images to generate backdoor data and put into the original dataset as the poison training dataset.

The poison test dataset and clean test dataset are used to measure the success rate of backdoor attacks and the Benign accuracy of models, respectively. Therefore, the clean test dataset is the original test dataset in the CIFAR-10 dataset (10000 images with 10 classes), and the poison test dataset is generated by modifying the dataset in the clean test dataset with non-target labels.

During the training process, we also recorded the attack success rate and Benign accuracy of the model. It is worth noting that to prove that our backdoor attack method does not require information from the Victim Model Architecture, we also tested the experimental effects of the Surrogate Model when it is different from the Victim Model.

In the Inference stage, we test the attack success rate of the final victim model, the Benign accuracy of the model, and other indicators (Fig. 3).

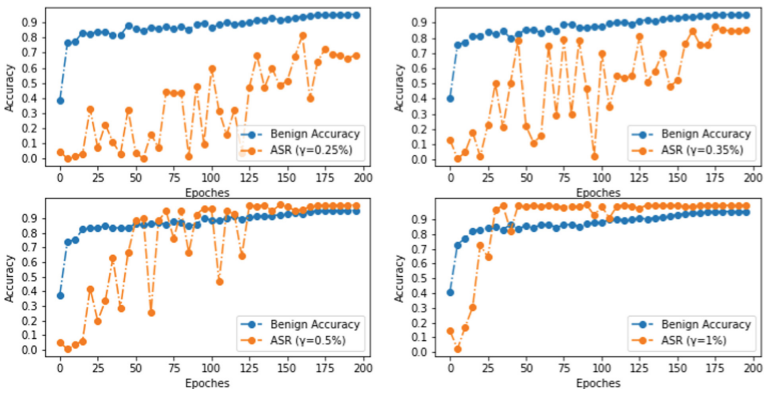

Fig. 3. Benign accuracy and attack success rate via epoch under different poisoning rate

Evaluation Metric: Here, we use two indexes, attack success rate and Benign accuracy to jointly measure the effect of the backdoor attack.

Attack success rate (ASR): recorded here is the proportion of the data of non-target labels that are successfully misclassified as target labels after being put into the victim model. This index directly reflects the effectiveness of the backdoor attack.

Benign accuracy refers to the performance of the victim model on Benign data sets. This index can also measure the stealthiness of backdoor attack methods to a certain extent.

4.2 Main Results

Table 2. Comparison between existing clean-label backdoor attack performance with ours under different poison budgets(the proportion of images that are poisoned). PB: Poison Budget, BA: Benign Accuracy, ASR: Attack Success Rate, TA: Target-label Accuracy

Approach	Ours					HTBA	SSBA
PB	**125**	**150**	**175**	**250**	**500**	**500**	**500**
BA (%)	95.36	95.35	95.41	95.33	95.32	94.52	94.41
ASR(%)	66.11	75.53	84.57	99.08	99.53	41.37	83.31
TA (%)	93.33	93.41	93.54	93.37	93.74	96.31	95.87

Attack Effectiveness

As shown in Table 2, our attack can successfully implant the backdoor into the Victim Model as the backdoor attack success rate is more than 99% with only 250 samples (0.5% of the training dataset). More specifically, we record experimental effects on different poison budgets in backdoor attacks. When poison samples occupy only 0.25% of the overall training dataset (125 pieces of poisoned images), the attack success rate can reach above 66%. In the meantime, during the improvement of the poisoning rate, Not only is the attack success rate on the rise, but the other two indexes that we pay attention to in the experiment are Benign accuracy and target-label accuracy, which have been kept around 95%. Benign accuracy remains a high effect. It can be seen that our scheme does not cause model performance degradation on the main task. Target-label accuracy also maintaines at a high level, indicating that our trigger generation algorithm can actually extract the intrinsic features of the target label data. This is because only by strengthening the connection between the label and its inherent features can the model's performance on the clean data of the target label not be degraded.

In addition, Table II also shows the attack performance of two classical clean-label backdoor attack schemes under the same experimental settings. Benign accuracy and target-label accuracy of HTBA and SSBA programs are almost consistent with ours, but it is obvious that when the poisoning rate has reached 1%, The attack success rate of our scheme is almost 100%, but the effect of HTBA and SSBA is far worse than our scheme.

In addition, to verify that our attack scheme does not depend on the structural knowledge of the victim model, we conduct corresponding experiments on the different pairs of surrogate model and victim model, and it can be concluded

from the results that no matter whether the structure of surrogate Model and victim Model is constant or not, our attack maintain a high attack success rate (Table 3).

Table 3. The ASR with pairs of surrogate model and victim model on Cifar-10 dataset and data poisoning rate $\gamma = 0.5\%$

Surrogate model	Victim model	
	ResNet-18	GoogLeNet
ResNet-18	99.08	95.46
GoogLeNet	96.35	99.26

Time Analysis
Although the generation method of our trigger is different from the random setting of the standard backdoor attack method, which needs to be generated by the surrogate model, our approach can iteratively generate the trigger of the corresponding category within 10s using the trained surrogate model while this trigger can complete the poisoning training and output manipulation for a target label.

5 Conclusion

In this paper, we propose a backdoor attack scheme that is most likely to be implemented in real scenarios. As a kind of clean-label backdoor attack, the attacker does not need to perform any label poisoning operation and has no information on non-target label data. As far as we know, the attacker's ability and knowledge are the weakest among these existing works. At the same time, we only need the data poisoning rate of 0.5% of the overall training dataset to achieve 99.53% success rate of backdoor attacks. Therefore, we believe that such a real-world backdoor attack scheme is enough to reveal a remarkable hidden danger in DNNs. Our work wants to indicate that relevant workers need to research the defense scheme against this kind of attack to protect the deep learning system.

Besides, to urge the development of the backdoor attack defense method, we will concentrate on research about backdoor attack techniques, which can escape the existing defense method, including human inspection and AI-based approaches.

References

1. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)

2. Li, X., Ma, C., Wu, B., He, Z., Yang, M.H.: Target-aware deep tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1369–1378 (2019)
3. Tramèr, F., Zhang, F., Juels, A., Reiter, M.K., Ristenpart, T.: Stealing machine learning models via prediction {APIs}. In: 25th USENIX Security Symposium (USENIX Security 16), pp. 601–618 (2016)
4. Shokri, R., Stronati, M., Song, C., Shmatikov, V.: Membership inference attacks against machine learning models. In: 2017 IEEE Symposium on Security and Privacy (SP), pp. 3–18. IEEE (2017)
5. Moosavi-Dezfooli, S.M., Fawzi, A., Frossard, P.: DeepFool: a simple and accurate method to fool deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2574–2582 (2016)
6. Li, B., Wang, Y., Singh, A., Vorobeychik, Y.: Data poisoning attacks on factorization-based collaborative filtering. In: Advances in Neural Information Processing Systems, vol. 29 (2016)
7. Alfeld, S., Zhu, X., Barford, P.: Data poisoning attacks against autoregressive models. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 30 (2016)
8. Biggio, B., Nelson, B., Laskov, P.: Poisoning attacks against support vector machines. arXiv preprint arXiv:1206.6389 (2012)
9. Shafahi, A., et al.: Poison frogs! targeted clean-label poisoning attacks on neural networks. In: Advances in Neural Information Processing systems, vol. 31 (2018)
10. Goldblum, M., et al.: Dataset security for machine learning: data poisoning, backdoor attacks, and defenses. IEEE Trans. Pattern Anal. Mach. Intell. (2022)
11. Muñoz-González, L., et al.: Towards poisoning of deep learning algorithms with back-gradient optimization. In: Proceedings of the 10th ACM Workshop On Artificial Intelligence and Security, pp. 27–38 (2017)
12. Steinhardt, J., Koh, PW., Liang, P.S.: Certified defenses for data poisoning attacks. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
13. Tianyu, G., Liu, K., Dolan-Gavitt, B., Garg, S.: BadNets: evaluating backdooring attacks on deep neural networks. IEEE Access **7**, 47230–47244 (2019)
14. Souri, H., Goldblum, M., Fowl, L., Chellappa, R., Goldstein, T.: Sleeper agent: scalable hidden trigger backdoors for neural networks trained from scratch. arXiv preprint arXiv:2106.08970 (2021)
15. Saha, A., Subramanya, A., Pirsiavash, H.: Hidden trigger backdoor attacks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 11957–11965 (2020)
16. Lin, J., Xu, L., Liu, Y., Zhang, X.: Composite backdoor attack for deep neural network by mixing existing benign features. In Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, pp. 113–131 (2020)
17. Nguyen, T.A., Tran, A.: Input-aware dynamic backdoor attack. In: Advances in Neural Information Processing Systems, vol. 33, pp. 3454–3464 (2020)
18. Shokri, R., et al. Bypassing backdoor detection algorithms in deep learning. In: 2020 IEEE European Symposium on Security and Privacy (EuroS&P), pp. 175–183. IEEE (2020)
19. Chen, X., Liu, C., Li, B., Lu, K., Song, D.: Targeted backdoor attacks on deep learning systems using data poisoning. arXiv preprint arXiv:1712.05526. (2017)
20. Quiring, E., Rieck, K.: Backdooring and poisoning neural networks with image-scaling attacks. In: 2020 IEEE Security and Privacy Workshops (SPW), pp. 41–47. IEEE (2020)

21. Li, Y., Li, Y., Wu, B., Li, L., He, R., Lyu, S.: Invisible backdoor attack with sample-specific triggers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 16463–16472 (2021)
22. Barni, M., Kallas, K., Tondi, B.: A new backdoor attack in CNNs by training set corruption without label poisoning. In: 2019 IEEE International Conference on Image Processing (ICIP), pp. 101–105. IEEE (2019)
23. Liu, Y., Ma, X., Bailey, J., Lu, F.: Reflection backdoor: a natural backdoor attack on deep neural networks. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12355, pp. 182–199. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58607-2_11
24. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
25. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L. ImageNet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255. IEEE (2009)