Fragile Neural Network Watermarking with Trigger Image Set

Renjie Zhu[1], Ping Wei[1], Sheng Li[1], Zhaoxia Yin[2], Xinpeng Zhang[1](✉), and Zhenxing Qian[1]

[1] School of Computer Science and Technology, Fudan University, Shanghai, China
{19210240144,17110240026,lisheng,zhangxinpeng,zxqian}@fudan.edu.cn

[2] School of Computer Science and Technology, Anhui University, Anhui, China
Yinzhaoxia@ahu.edu.cn

Abstract. Recent studies show that deep neural networks are vulnerable to data poisoning and backdoor attacks, both of which involve malicious fine tuning of deep models. In this paper, we first propose a black-box based fragile neural network watermarking method for the detection of malicious fine tuning. The watermarking process can be divided into three steps. Firstly, a set of trigger images is constructed based on a user-specific secret key. Then, a well trained DNN model is fine-tuned to classify the normal images in training set and trigger images in trigger set simultaneously in a two-stage alternate training manner. Fragile watermark is embedded by this means while keeping model's original classification ability. The watermarked model is sensitive to malicious fine tuning and will produce unstable classification results of the trigger images. At last, the integrity of the network model can be verified by analyzing the output of watermarked model with the trigger image set as input. The experiments on three benchmark datasets demonstrate that our proposed watermarking method is effective in detecting malicious fine tuning.

Keywords: Neural network · Fragile watermarking · Model integrity protection · Malicious tuning detection · Data poisoning · Backdoor defense

1 Introduction

Deep neural networks (DNN) have been widely used in all areas, such as computer vision [1], natural language processing [2], etc. In addition, a variety of learning methods have been proposed, such as reinforcement learning [3,4], federated learning [5], and so on. Then comes into our sight not only the security of data [6], but also the security of models. These well-trained deep models are valuable assets to the owners. However, they may be possessed or tampered illegally. For example, customers who buy a DNN model might distribute it beyond the license agreement, or attackers may inject backdoor into the models.

H. Qiu et al. (Eds.): KSEM 2021, LNAI 12815, pp. 280–293, 2021.
https://doi.org/10.1007/978-3-030-82136-4_23

Some malicious attacks like adversarial examples [7], data poison [8] and backdoor attack [8] are very common in deep learning. Correspondingly, some measures have been taken to solve these security issues. Among them, neural network watermarking is a promising research area. Digital watermarking is a traditional technique used for copyright protection or integrity authentication of digital products. Neural network watermarking is the extension of traditional watermarking concept for neural networks. And neural network watermarking techniques can be classified into two types: robust and fragile watermarking.

So far, most of the published researches are robust watermarking techniques used for protecting the copyright of DNN models. The word robust means these methods are insensitive to changes that aim to remove the embedded watermark. The robust watermarking techniques can roughly be divided into two categories: weight-parameter-based methods [9–13] and trigger-set-based methods [14–18]. The former ones are white-box schemes, in which the details of network parameters are needed. While the latter ones are black-box watermarking methods requiring no inner parameters of the models. In these methods, a trigger image set is built in advance and these images may be assigned with false labels that are irrelevant to their contents. In the verification process, the watermarked model's classification results of trigger set can be used for authentication directly.

Fragile watermarking [19] is originally designed for multimedia authentication, which is sensitive to content modification and is usually transparent in terms of perception. Now we migrate the concept of traditional fragile watermarking to neural networks. For DNN fragile watermarking, the following properties should be considered. First, it should require low training cost and be easy to embed and extract from the model. Second, the embedded watermark should be imperceptible and has no much impact on model's original performance. Third, there should be some quantifiable metrics for malicious tampering authentication. Fourth, it should be extensible and can be widely applied to other networks and datasets.

Formally, image classifier $\mathcal{C}_\theta$ is a supervised learning task aiming at finding a classification function $\mathcal{F}$ to classify the images in training set Tra with the classification loss $\mathcal{L}_{cla}$, i.e., $\mathcal{C}_\theta = \mathcal{L}_{cla}(\mathcal{F}(Tra))$. Usually, the trigger-set-based watermarking methods need a trigger image set Tri apart from training set, where images in Tri are stamped with preset labels according to some rules. And watermarked classifier $\mathcal{C}_{\theta_w}$ tries to classify the images in both the training set and trigger set, i.e., $\mathcal{C}_{\theta_w} = \mathcal{L}_{cla}(\mathcal{F}(Tra \cup Tri))$. Watermarked models gain the ability to recognize both normal images and trigger images by training the network from scratch or fine tuning trained models. For unmodified watermarked models, $\mathcal{C}_{\theta_w}$ is supposed to output the predefined labels for input trigger images.

DNN models are vulnerable to malicious attacks like data poisoning attacking. Now many DNN models need multi-parties training, the participants might inject backdoor into the network while updating parameters. Typical data poisoning behaviors can be classified into three kinds as follows:

1) Simple data poisoning [8]: attackers attempt to reduce the model performance by introducing lots of mislabeled samples to the training set, which can be expressed as: $\mathcal{C}_{\theta p} = \mathcal{L}_{cla}(\mathcal{F}(Tra \cup Mislabeled\ Samples))$.
2) Backdoor data poisoning [8]: it is a more imperceptible way of model tampering by adding some poisoned samples with a fixed backdoor pattern to the training set, that is, $\mathcal{C}_{\theta b} = \mathcal{L}_{cla}(\mathcal{F}(Tra \cup Backdoor\ Samples))$. Through training, this backdoor pattern can be added to normal samples to obtain the expected output label y_b, i.e., $\mathcal{C}_{\theta b}(x \oplus Backdoor\ pattern) = y_b$. This data poisoning method is difficult to be found, for only a small number of backdoor samples are needed and the injected backdoor has little impact on model's performance.
3) Label-consistent data poisoning [20]: label consistency means there is no tampering with image labels. This kind of attacking method often use adversarial example or GANs to recreate the samples in training set, making DNN more difficult to learn the features of image content. So, the backdoor attacking can succeed because networks focus on the backdoor pattern more than often. And traditional backdoor samples with wrong labels are easy to be detected by checking the training set, therefore, the label-consistency data poisoning methods have drawn much attention.

Some approaches have been proposed to detect the malicious tampering of DNN models. In [21], a detection and mitigation system for DNN backdoor attacks is proposed, where the backdoor can be identified and even some mitigation techniques are proved to remove the embedded backdoors. And in [22], a black-box based backdoor detection scheme is presented with minimal prior knowledge of the model. Both of them are solutions of detecting the existence of backdoors afterwards, and no precautions are taken to prevent backdoor inserting. In [13], a reversible watermarking algorithm for integrity authentication is proposed, in which the parameters of the model can be fully recovered after extracting the watermarking and the integrity of the model can be verified by applying the reversible watermarking. However, it's a white-box method requiring the details of networks, which is inconvenient for watermarking embedding and extraction. Therefore, we proposed a black-box based integrity authentication method with fragile watermarking technique, in which a trigger set is used and no model inner parameter is revealed. Let's call it: fragile neural network watermarking with trigger image set.

The contributions of this paper are summarized as follows:

– We firstly proposed a black-box based fragile watermarking method for authenticating the integrity of DNN classifiers.
– A novel loss function, $\mathcal{L}_{var}$ and an alternate two-stage training strategy were put forward elaborately, with which fragile watermark can be embedded easily into the neural network. Meanwhile, two easily accessible metrics are designed for model authentication, which can be obtained quickly by only checking the classification outputs of trigger images.

- Our proposed watermarking method is of good compatibility and extensibility, and experiments on three benchmark datasets showed that the embedded watermark have little impact on the prior task of the network.

The rest of this paper is organized as follows: Sect. 2 describes the proposed fragile watermarking and authentication metrics. Then, the properties of our scheme are demonstrated in Sect. 3. At last, conclusion is given in Sect. 4.

2 Fragile Watermarking

2.1 Application Scenario

Before introducing our proposed watermarking method, let us describe the application scenario with the following scenes. Consider three parties: model trainer, consumer, and attacker. The training of a complex network requires multiple stages of adjustment, and an attacker among the trainers may use the convenience of accessing training data for poisoning, resulting in network backdoored or performing worse than expected. And attackers could even poison models delivered to consumers. To this end, we introduce a fragile watermarking method suitable for neural networks to verify the integrity of watermarked models.

2.2 Watermarking Methodology

Figure 1 shows the overview of the proposed watermarking method. The whole process is divided into three steps: the first step is to construct a trigger image set using a secret key; and the second step is to embed a fragile watermark in a two-stage training procedure; the last step is the authentication process determining whether a neural model is tampered through two proposed metrics.

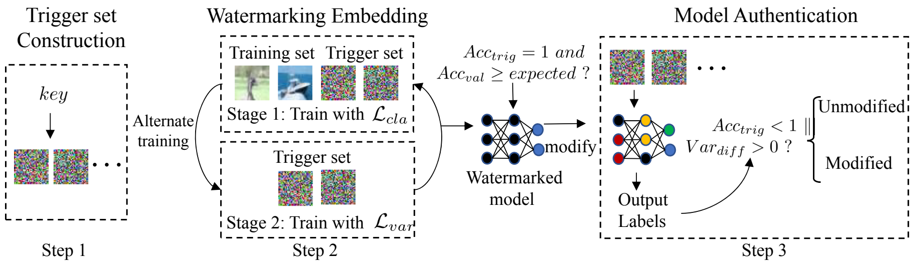

Fig. 1. An overview of the proposed fragile neural network watermarking method. The process of our scheme is divided into 3 steps. The fragile watermark is embedded by fine tuning the well-trained models in a two-stage alternate training manner, until the fine-tuned models satisfy the following condition: $Acc_{trig} = 1$ and $Acc_{val} \geq expected\ value$. At last, the integrity of fragile watermarked model can be authenticated by evaluating two metrics, namely $Acc_{trig} < 1$ or $Var_{diff} \geq 0$.

The first step of our scheme is to generate L pseudo-random trigger images with the secret key key specified by the model trainer. Each trigger image is assigned with a fix label, which can be expressed as follows:

$$\{Image_i, Label_i\} \leftarrow \{(I_i,\ i\%C) \mid i = 0, 1, \cdots, L-1\}, \tag{1}$$

where $Image_i$ or I_i represents a trigger image and $Label_i$ is its preset label $i\%C$. Here, % is the Mod operation of math, and C is the total number of classes in training dataset. Figure 2 shows three trigger samples used in our experiments.

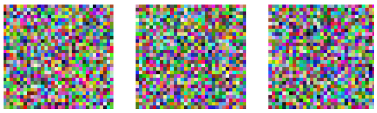

Fig. 2. Three examples of trigger images.

The second step of our method is watermarking embedding process. Our proposed fragile watermark is usually embedded in the well-trained models by a two-stage fine tuning procedure, with the classification loss $\mathcal{L}_{cla}$ and fragile watermarking loss $\mathcal{L}_{var}$, which are defined as follows:

$$\mathcal{L}_{cla} = -\sum_{j=0}^{C-1} y_j \log(p_j), \tag{2}$$

$$\mathcal{L}_{var} = \mathcal{L}_{cla} + \alpha \cdot \mathrm{Var}(\mathcal{P}), \tag{3}$$

where $\mathcal{L}_{cla}$ is the cross entropy loss for multi-class classification. For fragile watermarking, the watermarked model should be sensitive to modification. Thus, a regularization term $\mathrm{Var}(\mathcal{P})$ is added in $\mathcal{L}_{var}$. Here α is a weight coefficient, and $\mathcal{P}$ is a vector of classified results for each trigger image after Softmax operation, i.e., $\mathcal{P} = \{p_j \mid j = 0, 1, \cdots, C-1\}$ $(\sum p_j = 1)$. Here, p_j is the predicted probability of each class which falls into (0, 1), and $\mathrm{Var}(\mathcal{P})$ is the variance of $\mathcal{P}$.

The watermark embedding process can be divided into two stages. In the first stage, training set (Tra) along with the whole trigger set (Tri) are used for model fine-tuning with loss $\mathcal{L}_{cla}$, where training set (Tra) is made up of partial images from the raw training dataset. The model is trained to recognize the images in both training set and trigger set. In the second stage, loss $\mathcal{L}_{var}$ is used to fine-tune the model only on the trigger set, with the purpose of reducing the variance of predicted probability vector and classifying all the trigger images rightly. This watermark embedding process would not stop until two conditions are satisfied: 1). The classification accuracy on trigger samples is equal to 1, i.e., $Acc_{trig} = 1$; 2). The classification accuracy on normal samples in validation set is equal to or greater than the expected value, that is, $Acc_{val} \geq expected$. By using this alternate training method, the proposed fragile watermark are embedded easily into the DNN models. And the classification accuracy on normal images are usually not lower than the original un-watermarked models.

Here, we also defined the concept $\mathcal{C}_{\theta robust}$ as the classifier with a robust watermark. It is trained on both training set and trigger set with loss $\mathcal{L}_{cla}$, which means only the first stage of watermarking embedding is used. It can be expressed as: $\mathcal{C}_{\theta robust} = \mathcal{L}_{cla}(\mathcal{F}(Tra \ \cup \ Tri))$. This is also the way that many previous methods [14–18] embed robust watermarks. By doing so, the output of the watermarked model to the trigger set will not be easily changed by fine-tuning the model, in other words, the watermark is not fragile.

The performance of fragile watermarked model has a close relation to the size of Tra, which reflects the trade-off between watermarking embedding efficiency and model performance. When the size of Tra declines, it takes less time for watermark embedding, yet the classification accuracy on normal images will also decrease. Hence, we randomly take 10% of the images in raw training database as the training set Tra. The sensibility of watermark is enhanced gradually as α increases, but a too large value can also lead to performance degradation. Here we limit α ranging from 0 to several hundreds in our experiments.

The last step of our scheme is model authentication. After acquired, the DNN model's predicted labels of trigger images will be quantified into two authentication metrics to decide whether the model has been modified or not.

2.3 Authentication Metrics

Two novel authentication metrics are proposed to verify whether a watermarked model has been modified in our scheme. The first one is Acc_{trig}, which is the classification accuracy on the input trigger images. If $Acc_{trig} < 1$, it indicates that the model has been modified. The stronger the malicious attack is, the more Acc_{trig} will drop. If a model is modified, the trigger images may be classified into different classes as follows:

$$\mathcal{N} = \{n_0, n_1, \ldots, n_i, \ldots, n_{C-1}\}, \tag{4}$$

where n_i means the number of trigger samples that are classified as the i-th class image, and the summation of n_i is equal to the image number of trigger set. Similarly, $\mathcal{N}_0$ is the corresponding statistical result of unmodified model. Based on these two values, the second metric of our scheme is given:

$$Var_{dif} = \mathrm{Var}(\mathcal{N}) - \mathrm{Var}(\mathcal{N}_0), \tag{5}$$

where Var_{dif} is used to measure the difference of classification results before and after the watermarked model is tampered. The value of Var_{dif} reflects the degree to which the model has been modified. Var_{dif} is always greater than or equal to zero and a great value means the watermarked model is modified severely.

3 Experiments

In our paper, watermarking experiments are conducted with resnet18 [23] and resnet50 [23], which were trained on datasets cifar-10 & cifar-100, and Caltech101, respectively. The information of these datasets is showed in Table 1. We trained the classifiers from scratch with the following configurations: the optimizer of resnet18 is SGD with momentum with the learning rate of 0.1, which decreases ten times every twenty epochs; the optimizer of resnet50 is Adam, and the learning rate is 1e−4, which is reduced to 1e−5 in the last 20 epochs. Then fragile watermarking is embedded by fine-tuning the trained classifiers above. During the process of watermarking or attacking, the optimizer remains unchanged, and the learning rate is set to value in the final stage of training.

Table 1. Dataset information

Dataset	cifar-10	cifar-100	Caltech101	Our *Tri*
Classes (C)	10	100	101	-
Image size	32,32,3	32,32,3	300,200,3	32,32,3
Dataset size	60000	60000	9145	≥36
Training size	45000	45000	6583	-
Testing size	10000	10000	1830	-
Val size	5000	5000	732	-

3.1 Visualization of Watermarking

In this part, we compared the outputs of fragile watermarked models by some visual methods. Figure 3 records the classification accuracy of images during the watermark embedding fine-tuning process, in which Val Acc and Trigger Acc means classification accuracy on validation set and trigger set, respectively.

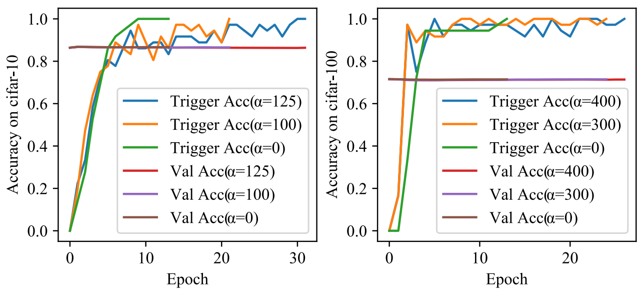

Fig. 3. The accuracy curves with different α values used during watermarking embedding. Val Acc and Trigger Acc means classification accuracy on validation set and trigger set, respectively.

When α is 0, a robust watermark instead of a fragile watermark is embedded, and with the increase of α, the sensibility of watermark increases and it will take more epochs for fragile watermark embedding. However, from Fig. 3, we can see that the increase of α has little effect on the performance of classifiers, for the watermarking embedding fine-tuning process would not stop until two conditions are satisfied, i.e., $Acc_{trig} = 1$ and $Acc_{val} \geq expected$. Thus, we can pick α ranging from zero to several hundreds casually for watermark embedding.

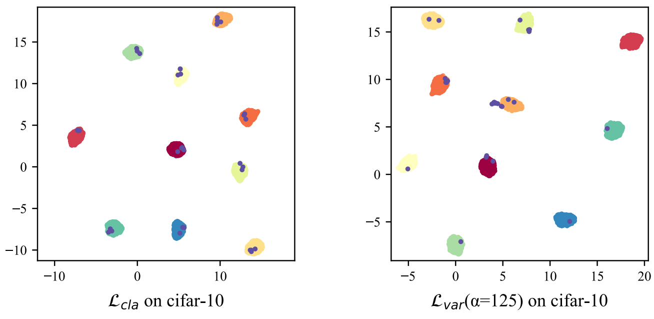

Fig. 4. Comparison of feature projection extracted by different models. The figures on the left and right are results of robust and fragile watermarked model trained on cifar-10, respectively.

The model's last convolution layer's output tensor were projected onto a two-dimension plane as demonstrated in Fig. 4 by a visualization tool UMAP. The left and right figures illustrate the projected features extracted by classifiers with robust and fragile watermark trained on cifar-10, respectively, where 10 larger colored circles represent the projected feature points of 5,000 normal samples, and the purple dots inside the circles are projected feature points of 36 trigger images, which all belong to 10 image classes. As shown in Fig. 4, once the model is embedded with fragile watermark, the projected feature points of trigger samples will deviate from the center to classification boundary.

3.2 Perceptual Transparency

The existence of fragile watermark has little influence on original classification ability of model. Table 2 lists the testing accuracy of un-watermarked models and fragile watermarked models when varying the value of α. Results show that the reduction of accuracy is less than 0.3% after watermark is embedded, and in some cases the accuracy of watermarked model even increase slightly compared to that of the clean model.

Table 2. The accuracy of un-watermarked classifier and fragile watermarked classifier for normal images in testing set when varying the value of α

Dataset	Un-water, Acc	α	Watermarked, Acc
cifar-10	86.40%	[0, 125]	[86.36%, 86.54%]
cifar-100	72.55%	[0, 400]	[72.31%, 72.57%]

3.3 Watermarking Property

In this part, we tested the property of proposed fragile watermarking under malicious attacking. To prove the effectiveness, 5 datasets listed as follows are used to attack the fragile watermarked models using data poisoning method.

1. The original training set (T).
2. An extra training set (ET), which has the same distribution with T. In our experiment, the validation set of original database is used as ET.
3. A poisoned dataset (DP), which has dozens of mislabeled samples with the aim to reduce the model performance.
4. A simple backdoor poisoned dataset (SB), which contains the images in training set (T) as well as some mislabeled samples with backdoor patterns.
5. A label-consistent poisoned dataset (LCB) constructed according to [20].

Five datasets above are created on the basis of a image database, such as cifar-10 and cifar-100. As [20] demonstrates, the minimum number of poisoned samples for backdoor attacking should not be lower than 0.15% of the training set size. Thus, for dataset cifar-10, we only added 75 (0.15% of the size) poisoned

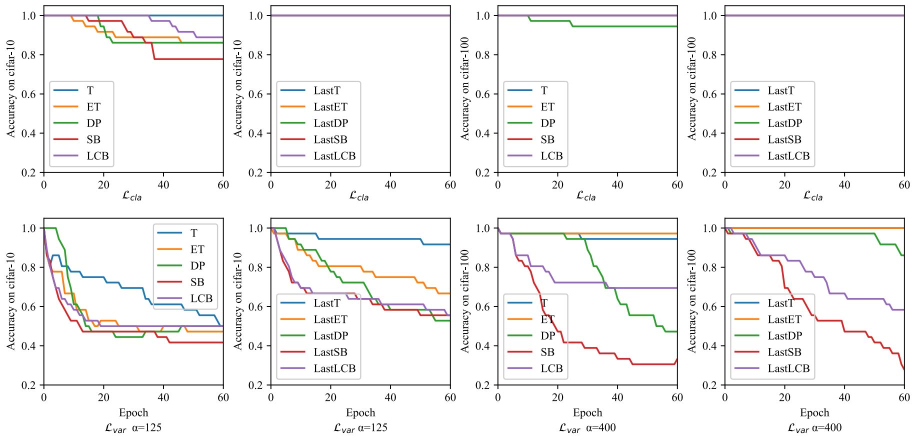

Fig. 5. The classification accuracy of trigger set (Acc_{trig}) when the robust watermarked models (the first row) and fragile watermarked models (the second row) are fine-tuned with 5 datasets (T, ET, DP, SB, and LCB) on cifar-10 or cifar-100, respectively. If the curve's label is prefixed with Last, it means only the last layer of model is fine-tuned, otherwise all layers are fine-tuned.

samples to dataset DP, SB, and LCB. In our experiment, the backdoor pattern is a $3 \times 3 \times 3$ white-and-black square block, which is superimposed on the lower right corner of normal samples.

Figure 5 illustrates the mean accuracy of trigger set when watermarked models are fine-tuned with 5 datasets above. The malicious attacking experiments are conducted either on all the layers, or only on the last layer of the watermarked models. In Fig. 5, the first and second row are respectively the attacking results of models with robust watermarks and fragile watermarks. It can be easily seen that the fragile watermarked models are vulnerable to all kinds of modifications compared to the robust watermarked models. The accuracy on trigger set Acc_{trig} declines rapidly when the fragile watermarked models are fine-tuned with poisoned datasets. And the more different the poisoned data is, the faster Acc_{trig} goes down. Thus, the accuracy curves of DP, SB, and LCB decline more quickly than the curves of T and ET.

Table 3. Value of Var_{dif} when the fragile watermarked model has been fine-tuned with 5 different datasets for ten epochs

Dataset	T	ET	DP	SB	LCB
Var_{dif}(cifar-10)	0.40	1.80	4.60	24.40	13.80
Var_{dif}(cifar-100)	0.00	0.00	0.02	0.30	0.30

Table 3 lists the Var_{dif} values, when the fragile watermarked model is fine-tuned with 5 datasets mentioned above. As can be seen, the Var_{dif} value of T and ET are much smaller than the values of other three poisoned datasets, with the reason that the images in T and ET are the same or similar to the images in raw training set. When fragile watermarked models are maliciously fine-tuned with poisoned dataset DP, SB, and LCB, Var_{dif} will be much greater than zero. The Var_{dif} values of cifar-10 are much larger than the values of cifar-100, mainly because that cifar-10 owns less images classes and is sensitive to modification.

In order to view the variation of Var_{dif} more intuitively, we depicted the heat maps of classified trigger image numbers according to $\mathcal{N}$ in equation (4). As shown in Fig. 6, the well trained model is fine-tuned with four datasets T, DP, SB, and LCB. The horizontal and vertical axis are the class number and the epoch number of fine tuning, respectively. For each block, the intensity of the color is proportional to the predicted number of trigger samples falling into the corresponding image categories. In the heat maps of poisoned datasets DP, SB, and LCB, the distribution of colored blocks are more uneven compared to blocks of T. This is because the embedded fragile watermark is damaged greatly when model is fine-tuned by these three poisoned datasets.

After a model is embedded with proposed fragile watermark, it may be upload to cloud or sent to the users directly. To verify the integrity of acquired model, we can apply the two metrics Acc_{trig} and Var_{dif} with the trigger image set offered

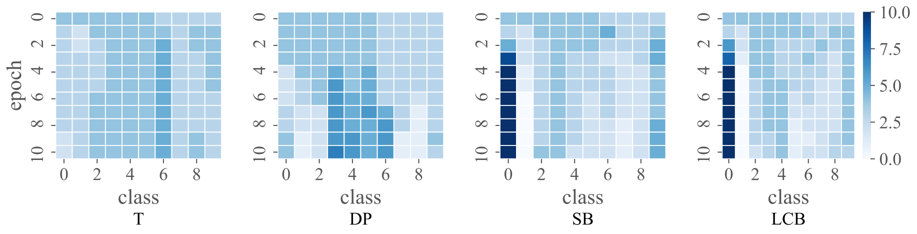

Fig. 6. The numbers of trigger samples being classified into each class during the malicious fine-tuning process carried on 4 datasets. The horizontal axis here is the image classes and vertical axis is the epochs. The color depth of blocks means the numbers of trigger images that are predicted as the corresponding image class.

Table 4. The classification accuracy of fragile watermarked models on normal images in testing set with different sizes of trigger set used in fragile watermarking

Trigger set size	Accuracy (cifar-10)	Accuracy (cifar-100)
0	86.40%	73.22%
36	86.54%	73.11%
100	86.38%	73.03%
200	86.21%	72.91%

by the model owner. As long as $Acc_{trig} < 1$ or $Var_{dif} > 0$, we can believe that the received models have been tampered.

3.4 Extensibility

Here, we explored the influence of trigger image set size on model performance. First, we constructed several trigger sets of size ranging from 0 to 200, and then each trigger set is used to embed fragile watermark individually. At last, the classification accuracy on the testing set are tested in Table 4. It can be seen that the size of trigger image set has no influence on the prediction consequences of fragile watermarked models.

The fragile watermarking method is also experimented on dataset Caltech101. In our experiments, classifier resnet50 is first embedded with fragile watermark, and then be fine-tuned with 5 datasets (T, ET, DP, SB, and LCB). Caltech101 is a 101-class dataset with the image size of about $300 \times 200 \times 3$ pixels, and the backdoor pattern for Caltech101 is a $9 \times 9 \times 3$ block. As a result, the testing set accuracy of the un-watermarked classifier resnet50 is 94.46%, and the accuracy of fragile watermarked classifiers is between 94.23% and 94.49%. The classification ability of DNN model is proved to be almost unaffected by watermarking.

The performance of watermarked resnet50 is also tested by fine tuning the model with 5 datasets on the basis of Caltech101, and the results are presented in Fig. 7. The left two and right two figures are separately the fine-tuned results

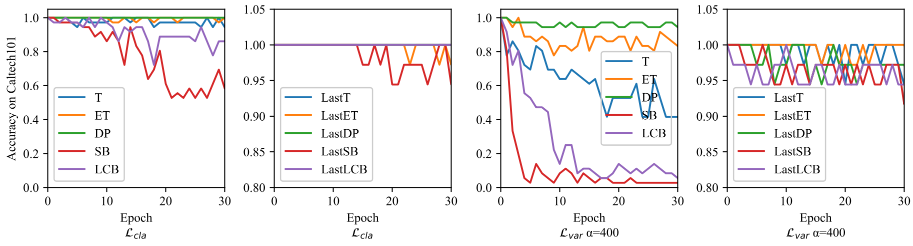

Fig. 7. The classification accuracy of trigger set (Acc_{trig}) when 5 datasets (T, ET, DP, SB, LCB) are used to fine-tune robust watermarked resnet50 and fragile watermarked resnet50 respectively on dataset Caltech101.

of robust and fragile watermarked resnet50. As illustrated in Fig. 7, fragile watermarked resnet50 is sensitive to all kinds of fine tuning, especially when fine-tuned with poisoned datasets DP, SB, and LCB. When all the layers of watermarked resnet50 are fine-tuned, the curves of poisoned datasets DP, SB, and LCB goes down more quickly than other curves. When only the last layer of model is fine-tuned, accuracy curves fluctuate wildly. But in general, our proposed fragile watermarking method is sensitive to malicious fine-tuning and can be used to detect model tampering carried on various datasets.

4 Conclusion

In this paper, we proposed a black-box based fragile neural network watermarking method with trigger image set for authenticating the integrity of DNN models. In our approach, models are trained to fit both the training set and the trigger set simultaneously in a two-stage alternate training process, which aims to embed fragile watermark while keeping the original classification performance. The embedded fragile watermark is sensitive to model tampering, and thus can be used to verify the integrity of models. Two meaningful metrics are provided to determine whether the fragile watermarked model has been modified as well as assess the distribution difference between the training set and the data used for malicious attacking. The experiments on three benchmark datasets have shown that our proposed fragile watermarking method is widely applicable to various classifiers and datasets. We leave the research on more sensitive semi-fragile neural network watermarking to future work.

Acknowledgment. This work was supported in part by the National Natural Science Foundation of China under Grants U1936214, U20A20178, 62072114, U20B2051, 61872003.

References

1. Deng, J., Berg, A., Satheesh, S., et al.: Imagenet large scale visual recognition competition 2012. See net. org/challenges/LSVRC, p. 41 (2012)

2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: 3rd International Conference on Learning Representations (2014)
3. Gai, K., Qiu, M.: Reinforcement learning-based content-centric services in mobile sensing. IEEE Netw. **32**(4), 34–39 (2018)
4. Gai, K., Qiu, M.: Optimal resource allocation using reinforcement learning for IoT content-centric services. Appl. Soft Comput. **70**, 12–21 (2018)
5. Yang, Q., Liu, Y., Chen, T., et al.: Federated machine learning: concept and applications. ACM Trans. Intell. Syst. Technol. (TIST) **10**(2), 1–19 (2019)
6. Dai, W., Qiu, M., Qiu, L., et al.: Who moved my data? privacy protection in smartphones. IEEE Commun. Mag. **55**(1), 20–25 (2017)
7. Szegedy, C., Zaremba, W., Sutskever, I., et al.: Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 (2013)
8. Gu, T., Dolan-Gavitt, B., Garg, S.: Badnets: Identifying vulnerabilities in the machine learning model supply chain. arXiv preprint arXiv:1708.06733 (2017)
9. Uchida, Y., Nagai, Y., Sakazawa, S., et al.: Embedding watermarks into deep neural networks. In: Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval, pp. 269–277 (2017)
10. Wang, T., Kerschbaum, F.: Attacks on digital watermarks for deep neural networks. In: ICASSP 2019–2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2622–2626 (2019)
11. Fan, L., Ng, K., Chan, C.: Rethinking deep neural network ownership verification: embedding passports to defeat ambiguity attacks. In: Advances in Neural Information Processing Systems, pp. 4714–4723 (2019)
12. Feng, L., Zhang, X.: Watermarking neural network with compensation mechanism. In: Li, G., Shen, H.T., Yuan, Y., Wang, X., Liu, H., Zhao, X. (eds.) KSEM 2020, Part II. LNCS (LNAI), vol. 12275, pp. 363–375. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-55393-7_33
13. Guan, X., Feng, H., Zhang, W., et al.: Reversible watermarking in deep convolutional neural networks for integrity authentication. In: Proceedings of the 28th ACM International Conference on Multimedia, pp. 2273–2280 (2020)
14. Adi, Y., Baum, C., Cisse, M., et al.: Turning your weakness into a strength: watermarking deep neural networks by backdooring. In: 27th {USENIX} Security Symposium ({USENIX} Security 18), pp. 1615–1631 (2018)
15. Zhang, J., Gu, Z., Jang, J., et al.: Protecting intellectual property of deep neural networks with watermarking. In: Proceedings of the 2018 on Asia Conference on Computer and Communications Security, pp. 159–172 (2018)
16. Guo, J., Potkonjak, M.: Watermarking deep neural networks for embedded systems. In: 2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 1–8 (2018)
17. Le Merrer, E., Pérez, P., Trédan, G.: Adversarial frontier stitching for remote neural network watermarking. Neural Comput. Appl. **32**(13), 9233–9244 (2019). https://doi.org/10.1007/s00521-019-04434-z
18. Zhu, R., Zhang, X., Shi, M., Tang, Z.: Secure neural network watermarking protocol against forging attack. EURASIP J. Image Video Process. **2020**(1), 1–12 (2020). https://doi.org/10.1186/s13640-020-00527-1
19. Zhang, X., Wang, S.: Fragile watermarking with error-free restoration capability. IEEE Trans. Multimed. **10**(8), 1490–1499 (2008)
20. Turner, A.l., Tsipras, D., Madry, A.: Label-consistent backdoor attacks. arXiv preprint arXiv:1912.02771 (2019)

21. Wang, B., Yao, Y., Shan, S., et al.: Neural cleanse: identifying and mitigating backdoor attacks in neural networks. In: 2019 IEEE Symposium on Security and Privacy (SP), pp. 707–723 (2019)
22. Chen, H., Fu, C., Zhao, J., et al.: Deepinspect: a black-box trojan detection and mitigation framework for deep neural networks. In: International Joint Conferences on Artificial Intelligence Organization, pp. 4658–4664 (2019)
23. He, K., Zhang, X., Ren, S., et al.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)