EURASIP Journal on Image and Video Processing

RESEARCH Open Access

Secure neural network watermarking protocol against forging attack

Renjie Zhu[1], Xinpeng Zhang[1,2*], Mengte Shi[1] and Zhenjun Tang[3]

* Correspondence: zhangxinpeng@fudan.edu.cn
[1]Fudan University School of Computer Science, No. 825, Zhangheng Road, Shanghai 201203, China
[2]Cyberspace Security Research Center, Peng Cheng Laboratory, No. 2, Xingke 1st Street, Shenzhen 518000, China
Full list of author information is available at the end of the article

Abstract

In order to protect the intellectual property of neural network, an owner may select a set of trigger samples and their corresponding labels to train a network, and prove the ownership by the trigger set without revealing the inner mechanism and parameters of the network. However, if an attacker is allowed to access the neural network, he can forge a matching relationship between fake trigger samples and fake labels to confuse the ownership. In this paper, we propose a novel neural network watermarking protocol against the forging attack. By introducing one-way hash function, the trigger samples used to prove ownership must form a one-way chain, and their labels are also assigned. By this way, an attacker without the right of network training is impossible to construct a chain of trigger samples or the matching relationship between the trigger samples and the assigned labels. Our experiments show that the proposed protocol can resist the watermark forgery without sacrificing the network performance.

Keywords: Watermarking protocol, Neural network, Trigger set

1 Introduction

The increasing application of neural network in daily life demonstrates that its intellectual property protection is an important issue, and that watermarking is an effective manner of ownership authentication. The techniques of watermarking neural networks can be generally classified into two categories: weight-parameter-based methods and trigger-set-based methods.

The first kind of neural network watermarking methods usually embed watermarks by slightly modifying weight parameters, and a matrix production between the watermarked parameters and a key-derived matrix is used to extract the embedded watermark. That implies the watermarked neural network must be a white box on watermark extraction phase. In fact, the owner may train the neural network by employing a loss function including an additional watermark extraction item for protection. Due to the large number of weight parameters of the neural network, the embedding positions and forms of specific watermarking scheme are also different. Uchida et al. [1] put the watermark bit into the parameters of convolutional layers through a binary cross entropy regularization loss function. When extracting, the average value of the corresponding parameters is multiplied by a pregenerated key matrix, and the product is the vector with the watermark information. Such kind of watermarking requires obtaining detailed information inside the neural network, including weight parameters, and output of layers, to directly investigate the network for verifying ownership. Watermark bits often contain identity information and may also be embedded in the variance of the weight parameters [2] or the output of activation layers [3].

The second kind of watermarking methods requires to construct a trigger set, including a few samples and their assigned labels, to train the neural network. After the training, the outputted result must be same as its corresponding assigned label when a trigger sample is input to the neural network, and the trigger set is only known by the owner. If the owner wants to prove his ownership, he may verify the matching relationship between the trigger samples and the labels by providing the trigger set to an authoritative third party. Here, any inner mechanism or weight parameter of the network is not revealed, i.e., the watermarked neural network could be a black box on ownership verification phase. In view of the good nature of trigger-set-based watermarking, the researchers have proposed many schemes to construct trigger sets. Adi et al. [4] took 100 abstract images and attached labels that did not correspond to the images' content to them. So this is their trigger set, and they utilized both the trigger set and the training set to train the classification neural network at the same time. Merrer et al. [5] used adversarial samples to construct the trigger set. These adversarial samples were obtained by adding disturbance to original images, and could not be correctly classified by ordinary models without trigger-set training, so they are also suitable for trigger samples. In addition, various schemes for constructing trigger sets are also proposed in [6, 7], including adding specific patterns or signatures, mixing Gaussian noise, and overlaying a special image related to watermark information on the original image.

Although the trigger-set-based methods are more convenient, for its fewer inner details of the neural network on the ownership verification phase, than the weight-parameter-based methods, an attacker capable of accessing the neural network may forge a fake trigger set to claim his ownership. Then, two watermarks exist in the neural network, and no one can distinguish the illegal watermark from the legal one. That implies the ownership of the neural network will be confused. The forging attack remains a problem of the trigger-set-based watermarking methods.

This paper proposes a novel trigger-set-based watermarking protocol against the forging attack. By introducing one-way hash function, the trigger samples to prove ownership must form a one-way chain. For an attacker who can assess the neural network but fail to train the neural network, it is impossible to construct a chain of trigger samples and the matching relationship between the trigger samples and the assigned labels. The protocol is suitable for neural networks with different function mechanisms and training methods. Our experiments also show that the proposed protocol can resist the watermark forgery without sacrificing the neural network performance.

2 Related work and forging attack

The forging attack is not aimed at removing the original legal watermark, but constructing another illegal watermark without any change on the watermarked cover. The forging attack of image watermarking is also called invertible attack or protocol attack, and a series of works are on this topic [8, 9]. If someone is given a watermarked image and a watermarking algorithm, he can construct a fake original image and a fake watermark and show the result of embedding the constructed watermark into the constructed original version is same as the given watermarked image, then the forging attack succeeds. By introducing a noninvertible mechanism into watermarking algorithm or protocol, the forging attack can be resisted. In [10], an enhanced forging attack is presented, in which the watermarking algorithm, as well as the original image and the watermark, is also constructed to perform a stronger attack. By establishing a relationship between the watermark extraction algorithm and the embedded watermark, the enhanced forging attack is therefore disabled in [10].

In the scenario of trigger-set watermarking for neural networks, after defining a trigger set and training a fairly well-performing watermarked network with a watermark, the owner puts the watermarked neural network on the cloud to serve others. Then, the users may submit inputs to the watermarked network to obtain the outputs. In particular, an attacker can collect a set of images and input them into the watermarked network, supposing it is a classification model, to obtain the corresponding outputted labels, and then, he could claim his trigger set including the collected images and the obtained labels. Since both the fake trigger set and the legal trigger set are effective for the watermarked network, the authoritative third party cannot determine who actually owns the trained network, even if the fake trigger set has not been used to train the network.

We call this kind of attack against the deep network watermark as the forging attack, which is a black-box attack [11]. In other words, it can only contact the network through a functional interface, but cannot modify the parameters and structure. The black-box attack is the most likely attack on trained networks. In many cases, an attacker cannot even get complete training data, let alone stealing a trained network. Therefore, the main purpose of our research is to resist this kind of black-box forging attack.

Although there have been some works aiming at resisting the forging attack, this problem has not been solved completely. For example, Li et al. [12] proposed a convolution operation to construct the trigger set so that the trigger samples are marked with identity information. Their trigger set is divided into two parts, one for watermark embedding and the other for preventing forging attacks. The former part of images is tagged as a common label, and the matching relation between these images and the common labels outputted by the watermarked network is used to verify ownership. Correspondingly, the latter images are attached with multiple labels, and this part of the training makes it difficult for forged trigger samples to be classified into the same category of the selected common label. In effect, although one cannot construct a fake trigger set compatible with the watermarking algorithm in [12], it is still feasible for an attacker to successfully construct fake trigger set compatible with other watermarking algorithms, thus the ownership can still be confused. To overcome this problem, we propose a secure watermarking protocol with wild compatibility in the next section.

In contrast to the black-box attack, the white-box attack [11] assumes that the attacker has obtained the complete network, including weight parameters and frame structure, but he still cannot use the stolen network publicly. What he needs is to remove the original watermark and even embed a new watermark in the network, while ensuring that the modification does not degrade its performance. Some researchers believe embedding new watermarks as an effective white-box attack. They argue that once multiple watermarks exist in the network, the copyright should be shared by all the owners of those watermarks. In fact, this is not reasonable in the case of an authoritative third-party certifier existing. Because one of the watermark embedders can provide the trusted third party with a network containing only one authenticated watermark to prove that he or she owns the copyright exclusively. Therefore, we believe that simply embedding a new watermark is not enough as a successful attack. For white-box attacks, we only consider the way to remove the original watermark by transfer learning. Once the original watermark is removed, the real owner cannot claim copyright. A secure neural network watermarking protocol should prevent the attack of black-box forging while resisting the attack of white-box removing. Later experiments will show that we did.

3 The proposed method

To resist the forging attack, the proposed watermarking protocol prescribes a manner of generating a trigger set. By introducing a one-way hash function, the trigger samples used to prove ownership must form a one-way chain, and their corresponding labels are also assigned. The protocol is compatible with various neural networks and training methods. As long as the owner obeys the protocol to embed a watermark, he can successfully provide an effective trigger set to prove his ownership. For an attacker who cannot train the network, although he accesses the watermarked network freely, it is impossible to construct a chain of trigger samples and the matching relationship between the trigger samples and the assigned labels. This way can disable the forging attack.

3.1 Secure watermarking protocol

Supposing we are protecting a N-classification neural network, the details of the protocol are as follows:

The authority T publishes two one-way hash functions H_X and H_y. Here, H_X takes an image X and a key k as input to output an image X_l of the same size

$$X_l = \mathrm{H}_X(X_{l-1}, k), l = 2, 3 \ldots, L \tag{1}$$

and H_y takes an image X as input to output an integer value y

$$y = \mathrm{H}_y(X) \tag{2}$$

When an owner of a neural network inserts his watermark into the network, he should select an initial image X_1 and a key k to calculate

$$X_l = \mathrm{H}_X(X_{l-1}, k), l = 2, 3, \ldots, L \tag{3}$$

and

$$y_l = \ mod\left(\mathrm{H}_y(X_l), N\right), l = 1, 2, \ldots, L \tag{4}$$

where N is the number of classification of the network and L samples, X_l ($l = 1, 2, \ldots, L$), form a chain. The value of L is assigned by the authority L and will be discussed later. Then, the owner collects the samples and the labels to generate the trigger set:

$$\{(X_l, y_l) \mid l = 1, 2, \ldots, L\} \tag{5}$$

and trains the network to ensure the classification result of inputting X_l must be y_l. Here, the training method may be determined by the owner.

For verifying the ownership, the owner should provide X_1 and a key to the authority and calculate X_l and y_l according to (3) and (4). The authority checks if the result of inputting X_l into the network is same as y_l. Denote

$$\mathrm{Check}(\mathrm{R}(\mathrm{X_l}), \mathrm{y_l}) = \begin{cases} 0, \text{if } \mathrm{R}(\mathrm{X_l}) \neq \mathrm{y_l} \\ 1, \text{if } \mathrm{R}(\mathrm{X_l}) = \mathrm{y_l} \end{cases} \tag{6}$$

and

$$t = \sum_{\mathrm{l}=1}^{\mathrm{L}} \mathrm{Check}(\mathrm{R}(\mathrm{X_l}), \mathrm{y_l}) \tag{7}$$

where $R(X_\mathrm{l})$ is the classification result of inputting X_l by the target network. In fact, t is the number of samples in the trigger set that can be correctly classified by the network. With a given threshold τ, if

$$t \geq \tau \tag{8}$$

the ownership is recognized. The value of the threshold τ will also be discussed later.

3.2 Protocol implementation

In the protocol, H_X and H_y can be any one-way hash functions that take an image and a key as input to output an image with the same size and take an image as input to

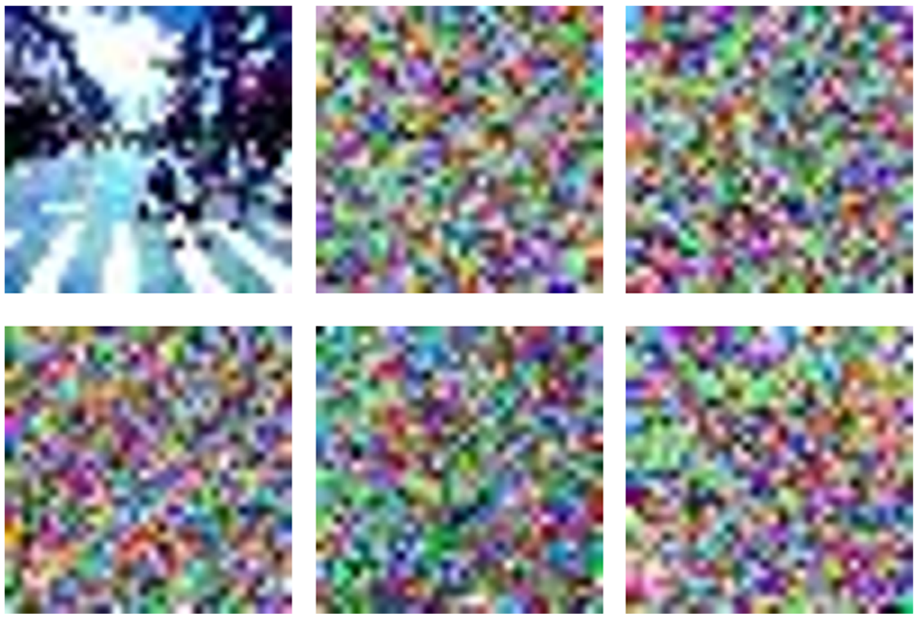

Fig. 1 An initial image and 5 trigger images in a hash chain

output an integer value, respectively. Actually, we define $H_x$and H_y based on some standard one-way hash function. To construct a trigger image, for example, an input image X_{l-1} is divided into non-overlapping blocks containing 32 pixels, and each block in image X_{l-1} is used to generate a block at the same position in the output image X_l,

$$b' = \mathrm{SHA256}(\mathrm{b} \parallel \mathrm{k}) \tag{9}$$

where b and b' are the blocks in X_{l-1} and X_l respectively. For the hash function, we select SHA256, whose output can be divided into exactly 32-pixel values. Then, X_l can be obtained from X_{l-1} in a block-by-block manner. As the hash is done each time, a fixed secret key is appended to the 32-pixel bit string b. The function H_y is to connect the bitstrings corresponding to all pixel values of image X_l in sequence, input it into SHA256, and output a hash value. Figure 1 gives an initial image and 5 obtained trigger images in the hash chain. In fact, there is not much required for the selection of the initial image.

We do not specify how similar or dissimilar it is to the training set images, as it will be used to construct numerous noise-like trigger set images. Although the trigger images except the first one similar to random noise, they can still be used for watermark verification. On the contrary, the huge visual difference between the trigger samples and the training images results in the trigger-set training without disrupting the learning of training images' visual content. In other words, the watermark embedding will not affect the establishment of the network function.

Here is to discuss the values of L and τ. In some application scenarios, the watermarked neural network may be transfer-learned, and we denote the probability that the result of inputting a single trigger sample X_l into the watermarked network is y_l as P. Since the classification results on trigger sets are produced by reliable training and difficult to erase, P is usually more than 99%. Then, the probability that the owner succeeds in verifying the ownership is

$$P_O = \sum_{i=\tau}^{L} C_L^i \times P^i \times (1-P)^{L-i} \tag{10}$$

On the other hand, attackers may forge trigger samples, and the labels of these forged samples are also randomly specified by equation (4). When a forged sample is input, the probability that the output of the watermarked network can be the same as the label calculated according to equation (4) is $\frac{1}{N}$, where N is the number of classifications of the classification network. Therefore, the success rate of forging attack is

$$P_A = \sum_{i=\tau}^{L} C_L^i \times N^{-i} \times \left(1-\frac{1}{N}\right)^{L-i} \tag{11}$$

Table 1 The recommended value of L and τ for the trigger sets of N-classification networks

N	2	5	10	20	50	100
The size of trigger set: L	135	55	38	29	23	20
The threshold: τ	130	52	36	27	21	18

To ensure the security of the protocol and limit the size of the trigger set, L and τ should be assigned the smallest values while satisfying $P_O \geq 99\,\%$ & $P_A \leq 10^{-32}$. Table 1 lists the values of L and τ with respect to various values of N.

4 Experimental results and discussion

In the experiment, we embedded watermarks for 10- and 100-classification resnet18 [13] trained with CIFAR-10 and CIFAR-100 data sets. According to the proposed protocol, we constructed trigger sets containing 38 and 20 samples for the two classification tasks respectively. And all the trigger samples are computed from the same image. For watermark embedding, two samples in trigger set are added into each training batch by turns, batch size of 100, for 60-epoch training, which ensures the watermark is embedded in the initial training phase of the network function establishment. To verify the embedded watermark, the number of samples in the two trigger sets that can be correctly classified according to the preset labels should be no less than τ (τ= 36 and 18, respectively, for 10-classification and 100-classification tasks).

4.1 Trigger-set training effect on classification

The watermark embedding does not seriously affect the original classification function of the network. The classification accuracy of the 10- and 100-classification networks we trained

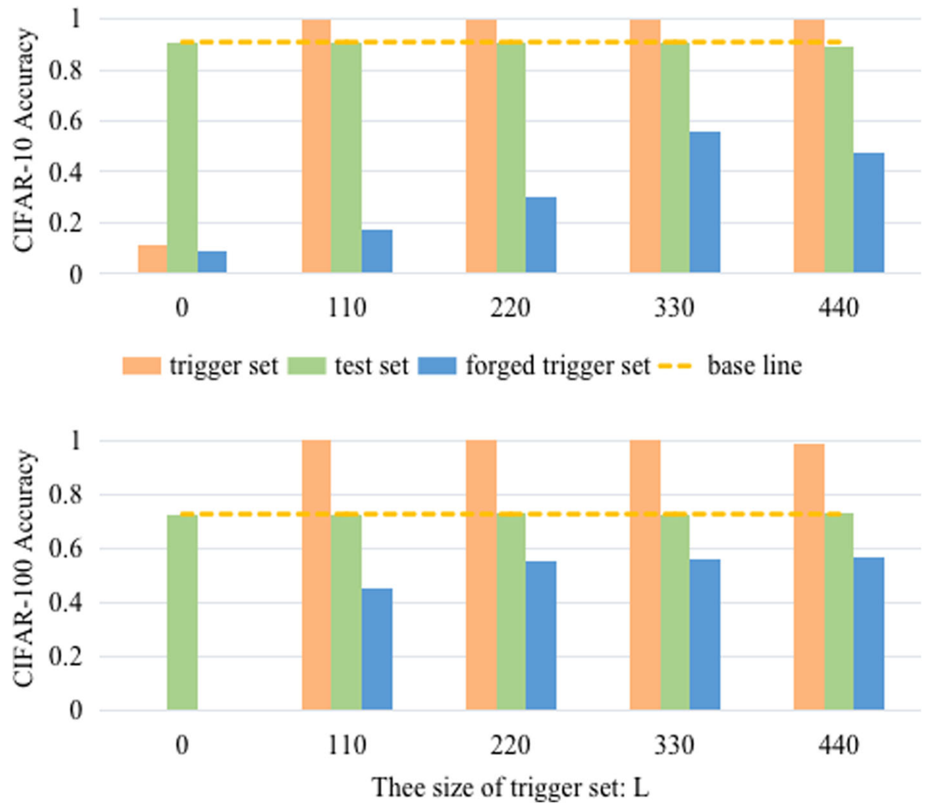

Fig. 2 The effect of increasing the size of the trigger set on the function of resnet18 [13]. The two bar charts above and below respectively represent the experimental results on CIFAR-10 and CIFAR-100. Orange, green, and blue bars are our trigger set, test set, and the attacker's forged trigger set classification accuracy

without watermarking was 0.9088 and 0.7273, respectively. When the trigger-set training was added, the two figures changed only slightly, to 0.9057 and 0.7334. We consider when we use images with visual perception significantly different from the training data as triggers, the trigger-set training makes little impact on the original classification function of the network. In addition, the strong fitting ability of the neural network can also support it to learn the classification of the trigger set. In fact, a negligible functional reduction is acceptable.

Watermarks in multimedia files are required not to destroy the perceived quality. For deep networks, we also require that watermarking cannot damage the networks' function. Figure 2 shows the effect of continuously increasing the size of the trigger set to 400 on the function of the networks. The yellow line indicates the classification accuracy of the network's function itself when trigger-set training is not added. Gradual increasing the quantity of trigger samples does not impair network's function (green column), or decrease the accuracy of the watermark (orange column). At the same time, the number of trigger samples which are forged in a black-box manner, being correctly classified, was always far below the threshold τ agreed to in the protocol. This indicates that embedding the watermark with the proposed trigger set size has little effect on the network function.

4.2 Watermark verification

In the watermark verification, the authoritative third party needs to input the submitted trigger set $\{(X_l, y_l) \mid l = 1, 2, \ldots, L\}$ into the watermarked network and count the number of trigger sample Xl, whose outputted label is same as y_l. Consider that an attacker uses his initial image X_1 to forge a fake trigger set according to (3) and (4) in the protocol, and submit the trigger set to an authoritative third party for watermark verification. Table 2 shows the verification results of the network owner's legal trigger set and the attacker's forged trigger set, where t_{owner} and t_{forged} are respectively calculated by (7) using the two sets. We performed the experiments for five times with different trigger sets, t_{owner} and t_{forged} taking the average value of the results. It can be seen that all L results of inputting owner's trigger sample X_l into the network are the same as y_l, which implies a powerful argument for ownership. On the other hand, since the forged trigger set has not been learned by the network, t_{forged} is much less than the threshold τ, leading to a failed watermark verification.

4.3 Comparison

After the forging attack is disabled, comparisons to the backdoor watermarking scheme [4, 12], and [4–7] are made in this part to show that the proposed method is as accurate and robust as previous studies and it also has little impact on the function of the network itself. In Table 3, we use the same network and datasets (resnet18 [13] on CIFAR-10 and CIFAR-100). The third and seventh rows are the

Table 2 Watermark verification results of owner's legal trigger set and attacker's forged trigger set

N	L	τ	t_{owner}	t_{forged}
10	38	36	38	3.4
100	20	18	20	0.4

Table 3 Comparison between our watermark and others'

Metrics	Fromscratch [4]	*WMcontent* [6]	*WMunrelated* [6]	*WMnois e*[6]	[7]	*Ax*⊕*W* [12]	Ours
			CIFAR-10				
Trigger-set accuracy	1	0.9993	1	0.9986	0.9998	1	1
Performance reduction	+0.0039	−0.0019	−0.0048	−0.0011	−0.0028	−0.0061	−0.0051
Trigger set size	100	100				≤ 100	38
			CIFAR-100				
Trigger-set accuracy	1					1	1
Performance reduction	−0.0034					+0.0011	+0.0066
Trigger set size	100					≤ 100	20

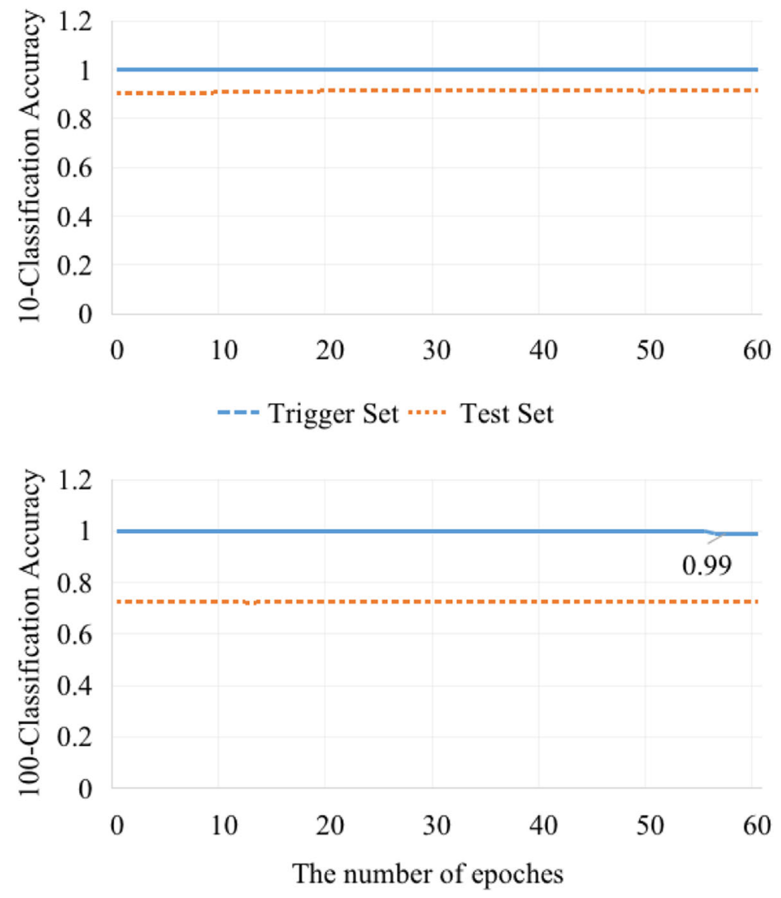

Fig. 3 Transfer learned watermarked network classification accuracy for the owner's trigger set and test set. The data in the figure is the average of five experiment results

accuracy comparison, the correct classification rate of the trigger set. Under the same training condition, it represents that the watermark we embed can be extracted completely correctly. The fourth and eighth lines show the performance reduction of the classification function on the test set caused by trigger-set training. Since only a small trigger set is used, the results show some randomness, and even sometimes watermarking causes a slight improvement. What is more, with a smaller trigger set, there is no doubt that our watermarks will be easier to embed and have less impact on network functionality.

However, our secure watermarking protocol able to eliminate forging attacks completely is the more important point that must be mentioned in the comparison. It is a feature that all other watermarking methods do not have and the novelty of the research lies in. Many existing watermarking methods [4–7] cannot resist forging attacks. Although the research in [12] can prevent attackers from constructing fake trigger sets compatible with their own watermarking algorithm, it is still feasible for an attacker to successfully construct fake trigger set compatible with other watermarking algorithms, so that the ownership can still be confused. In our scheme, the authority of the protocol combined with the one-way hash function completely blocks the forging attack.

4.4 Transfer learning

Another widely discussed attack in the white-box manner is transfer learning to re-embed a new watermark in the presence of the original watermark. However, it cannot succeed due to the existence of third-party authenticators. Furthermore, a network embedded with more than one watermark has no authentication significance, because the true owner can constantly submit a network containing only one watermark to prove his exclusive ownership. This is a fact that has been overlooked by some current researches. Therefore, for the white-box attack, it is only necessary to consider the attack of removing the original watermark.

The watermark embedded under the protocol is also secure enough to resist the watermark removing in a white-box manner of transfer learning. One type of removing attack is to fine-tune the watermarked network using a training set with the purpose to make the fine-tuned network no longer output y_l to the input sample X_l in the owner's trigger set. In the experiment, we fine-tuned the watermarked networks' all layers for 60 epochs with their datasets (CIFAR-10 and CIFAR-100). This kind of experiment was carried out five times, and as a result, 100% of the owner's trigger samples can be correctly classified into the categories calculated according to equation (4), meaning the watermark could not be erased. Another possible way to remove watermarks is to construct a new trigger set and then use the full training set with the new trigger set to train the watermarked network in the same way that the original watermark is embedded. Figure 3 shows the classification accuracy of the owner's trigger set and test set. The data in the figure is also the average value of the five experiment results. Only one sample in the 100-classification trigger set was misclassified in one experiment, and the remaining 19 samples were correctly classified, in which case, the owner's watermark verification was also successful according to the threshold we set in Table 1. In all experiments, no less than 99% of the owner's trigger samples were correctly classified by transfer learned watermarked networks, which illustrated the robustness of the watermark. Although this transfer learning attack does not degrade the classification accuracy of the test set, the copyright of the watermarked network is protected as long as the owner's watermark cannot be removed.

5 Conclusion

For the intellectual property protection of trained neural networks, we propose a novel trigger-set-based watermarking protocol against the forging attack. The introduction of one-way hash function makes trigger samples generated as a chain, and the precursor in the chain is the source of the successor. In addition, we randomly specify the labels of these samples by hashing. With this kind of trigger-set training, the authorized third party can determine whether the user's ownership is recognized according to the number of trigger samples being matched with the preset labels by the watermarked network. And it is impossible for an attacker to obtain a set of images that can be verified as a legal trigger set through random testing. In other words, watermarks can only be embedded by training.

It should be emphasized that our study is not to propose a method that only solves the problem of forging attack based on a single watermarking scheme, but to publish a secure protocol that is compatible with various watermarking algorithms. It ensures that watermarks embedded according to the protocol can avoid the trouble of forgery.

The resolution of the forging attack promotes more complete and practical neural network watermarking.

At the same time, there is a big visual difference between our trigger images and the training images, where the embedding of watermark will not affect the establishment of the network function. The removing attack experiments also demonstrate that our watermark is robust enough to ensure that the ownership of transfer learned watermarked networks can also be verified successfully.

Acknowledgements
Not applicable.

Authors' contributions
XZ conceived the idea. RZ designed and performed the experiments. MS did literature research. XZ and RZ wrote the paper. All authors read and approved the final paper. ZT provided meaningful discussion and experimental assistance.

Funding
This work was supported by the Natural Science Foundation of China (U1936214, U1636206, and 61525203).

Availability of data and materials
The datasets used and/or analyzed during the current study are available from the corresponding author on reasonable request.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]Fudan University School of Computer Science, No. 825, Zhangheng Road, Shanghai 201203, China. [2]Cyberspace Security Research Center, Peng Cheng Laboratory, No. 2, Xingke 1st Street, Shenzhen 518000, China. [3]Guangxi Normal University School of Computer Science and Information Technology, No. 15 YuCai Road, Guilin 541004, China.

Received: 22 April 2020 Accepted: 23 August 2020
Published online: 04 September 2020

References

1. Uchida, Y., Nagai, Y., Sakazawa, S., Satoh, S.: Embedding watermarks into deep neural networks. In: Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval, pp. 269–277 (2017). ACM
2. 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). IEEE
3. Darvish Rouhani, B., Chen, H., Koushanfar, F.: Deepsigns: An end-to-end watermarking framework for ownership protection of deep neural networks. In: Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, pp. 485–497 (2019). ACM
4. Adi, Y., Baum, C., Cisse, M., Pinkas, B., Keshet, J.: Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In: 27th {USENIX} Security Symposium ({USENIX} Security 18), pp. 1615–1631 (2018)
5. Le Merrer, E., Perez, P., Tredan, G.: Adversarial frontier stitching for remote neural network watermarking. Neural Computing and Applications, 1–12 (2019)
6. Zhang, J., Gu, Z., Jang, J., Wu, H., Stoecklin, M.P., Huang, H., Molloy, I.: 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). ACM
7. 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). IEEE
8. Craver, S., Memon, N., Yeo, B.-L., Yeung, M.M.: On the invertibility of invisible watermarking techniques. In: Proceedings of International Conference on Image Processing, vol. 1, pp. 540–543 (1997). IEEE
9. Fridrich, J.: Security of fragile authentication watermarks with localization. In: Security and Watermarking of Multimedia Contents IV, vol. 4675, pp. 691–700 (2002). International Society for Optics and Photonics
10. X. Zhang, S. Wang, Invertibility attack against watermarking based on forged algorithm and a countermeasure. Pattern Recogn. Lett. **25**(8), 967–973 (2004)
11. Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z.B., Swami, A.: Practical black-box attacks against machine learning. In: Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, pp. 506–519 (2017). ACM
12. H. Li, E. Willson, H. Zheng, B.Y. Zhao, Persistent and unforgeable watermarks for deep neural networks. arXiv preprint arXiv **1910**, 01226 (2019)
13. 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)