DATA-FREE WATERMARK FOR DEEP NEURAL NETWORKS BY TRUNCATED ADVERSARIAL DISTILLATION

Chao-Bo Yan[†], *Fang-Qi Li*[†], *Shi-Lin Wang*[*], Senior Member, IEEE

Shanghai Jiao Tong University
School of Electronic Information and Electrical Engineering
{yanchaobo,solour_lfq,wsl}@sjtu.edu.cn

ABSTRACT

Model watermarking secures ownership verification and copyright protection of deep neural networks. In the black-box scenario, watermarking schemes commonly rely on injecting triggers and requiring the model's training data to maintain its performance. However, such knowledge might be unavailable in commercial settings as model transactions or copyright transfers. To tackle this challenge, we propose a novel data-free black-box watermarking scheme. Our approach modifies data-free adversarial distillation to efficiently obtain a generator that produces samples serving as a substitute for the training data so the watermark can achieve high fidelity without referring to the training data.

Index Terms— Machine learning security, Neural network watermarking, Data-free watermarking

1. INTRODUCTION

Deep neural network (DNN) models have found widespread application in various scenarios. Training a DNN is resource-intensive, so safeguarding the intellectual property of model owners becomes crucial. A prevalent method for protecting model copyrights is watermarking [1, 2, 3]. In the black-box setting where the judge can only access the suspicious model as an API, the watermarking schemes rely on a set of triggers [3, 4]. The verification program compares the suspicious model's predictions for these samples with predefined labels.

The learning of trigger samples has to be integrated with the training process. Otherwise, the performance of the watermarked model may decline catastrophically [5]. Existing schemes uniformly assume that the entity executing the watermarking algorithm has access to the model's training data. However, in the context of machine learning-as-a-service (MLaaS) [6], users who possess the model and require watermarking services are not necessarily the trainers, and the training data is not always publicly available for privacy concerns, as shown in Fig. 1.

[†]Equal contribution.[*]Corresponding author.

The work described in this paper was supported by the National Natural Science Foundation of China (62271307, 61771310).

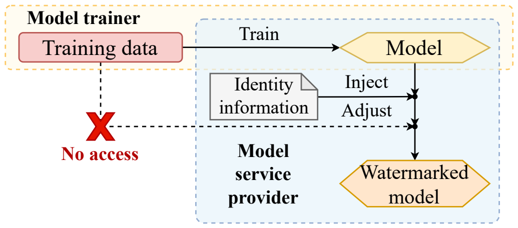

Fig. 1. A scenario where the DNN model service provider has no access to the training data.

To foster the applicability of model watermarking, we propose a data-free DNN watermarking scheme. A truncated data-free distillation process efficiently generates substitute samples for the original training data to maintain the model's performance under watermarking. The ownership of the model has sufficient robustness against pruning and fine-tuning attacks. The contributions of this paper are:

- We propose a truncated data-free adversarial distillation scheme to enhance the efficiency of generating substitute samples for the training data.
- By utilizing the substitute samples, we implement the first data-free black-box watermarking scheme.
- We conduct extensive experiments to evaluate and verify the advantage of the proposed method.

2. MOTIVATION

Denote the DNN model to be protected as M_{clean}. Black-box DNN watermarking is achieved by embedding a collection of triggers $\{\mathbf{t}_n, l_n\}_{n=1}^{N}$ into M_{clean}. The watermarked model's performance would significantly decline unless triggers are learned simultaneously with a batch of normal training data $\{\mathbf{x}_m, y_m\}_{m=1}^{M}$. The loss function for watermarking is:

$$\mathcal{L}(M_{\text{WM}}) = \underbrace{\sum_{n=1}^{N} \text{CE}(M_{\text{WM}}(\mathbf{t}_n), l_n)}_{\text{watermark injection loss}} + \lambda \underbrace{\sum_{m=1}^{M} \text{CE}(M_{\text{WM}}(\mathbf{x}_m), y_m)}_{\text{model adjustment loss}} \tag{1}$$

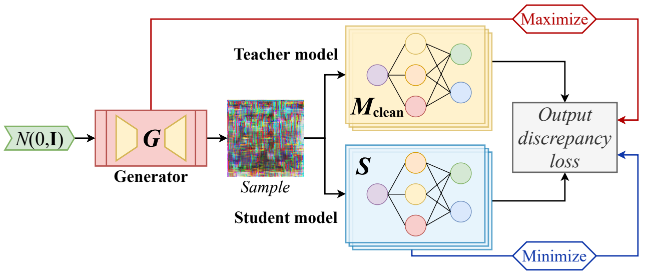

Fig. 2. Data-free adversarial distillation.

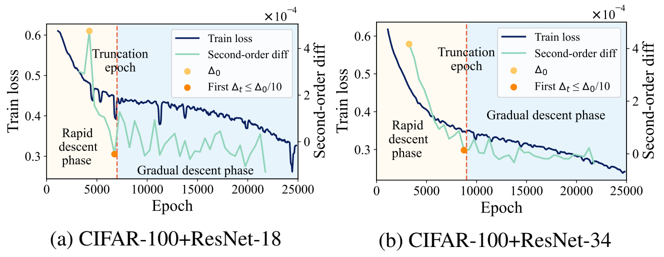

Fig. 3. The empirical discrepancy and its second-order difference during DFAD.

where M_{WM} is the watermarked model with parameters initialized as M_{clean}'s and CE is the cross-entropy loss. The objectives are (i) fixing M_{WM}'s predictions on triggers as the ownership evidence and (ii) maintaining M_{WM}'s functionality as M_{clean}'s. The second objective is referred to as *fidelity*. Our discussion concentrates on the classification task. However, it can be easily generalized to other tasks since respective watermarking schemes have been well-defined [7].

In real-world scenarios, it is common that the training data is unavailable during the watermarking phase. We refer to this as the *data-free* setting. For instance, Company A may outsource model development to Company B due to resource constraints. Company B delivers the model to Company A without providing the training data as it may constitute proprietary intellectual property exclusive to Company B (e.g., Google's machine translation dataset [8]) or contain sensitive data subject to Company B's privacy commitments (e.g., Facebook's DeepFace model dataset [9]).

Without access to the training data, the second term of the r.h.s of Eq. (1) becomes intractable. Finding a substitution to adjust the watermarked model is related to the target of data-free adversarial distillation (DFAD) [10, 11], whose aim is to build a student model with comparable performance from a teacher model without accessing the training data. A typical DFAD operates as shown in Fig. 2; it alternately trains the student to minimize the discrepancy between its decision boundary and that of the teacher-given samples produced by a generator, and tunes the generator to generate samples that maximize the discrepancy. Samples generated from the generator are considered as a substitute for the training data.

The longstanding objective of DFAD is to obtain high-performing student models, which is time-consuming and usually exceeds the training time of the teacher model. We are encouraged to improve the efficacy of finding a usable generator instead of completing the DFAD.

3. METHOD

3.1. Generator Training

We train a generator G to produce samples with which the watermarked model is adjusted. An auxiliary DNN model S with the same architecture as M_{clean} serves as the student. The workflow is shown in Fig. 2.

G transforms noises into samples, which are fed to the teacher and the student. The empirical discrepancy is:

$$\mathcal{D}(G,S) = \sum_{\mathbf{z}\sim\mathcal{N}(0,\mathbf{I})} \left(M_{\text{clean}}(G(\mathbf{z})) - S(G(\mathbf{z}))\right)^2 \qquad (2)$$

where the l_2 loss is computed from the logits of two models, and $\mathcal{N}$ is a multidimensional Gaussian distribution.

During training, G is tuned to maximize Eq. (2) while S is tuned to minimize it. It is expected that samples generated by G can serve as a substitution of training data in Eq. (1) to reduce the decline of fidelity.

To accelerate the training, we adopt a rule-based truncation strategy. As an illustration of our intuition, we visualize the variation of the discrepancy during a training process on CIFAR-100 for 25000 epochs in Fig. 3. It is observed that the loss underwent two distinctive phases: a rapid descent phase and a gradual descent phase. This change is characterized by the second-order difference, which is also illustrated in Fig. 3.

During DFAD, the generator first generates easy samples and later hard samples [10]. In model watermarking, the topology of the M_{WM}'s decision boundary does not significantly deviate from that of M_{clean} since the M_{WM}'s parameters are initialized as M_{clean}'s. Therefore, easy samples generated in the earlier stage are sufficient to preserve fidelity.

We truncate the training procedure based on the variation of the loss function. Firstly, both G and S are trained alternately for t_0 epochs, during which the maximal second-order difference is saved as Δ_0. Secondly, for the next t-th epoch, the second-order difference of the discrepancy is recorded as Δ_t. Once $\Delta_t \leq \frac{\Delta_0}{10}$, the training is truncated and the current generator G is saved as the final version.

3.2. Watermark Injection

The watermark injection process is shown in Fig. 4. The performance of the watermarked model is adjusted with respect to samples generated from G. Their soft labels are assigned by M_{clean}'s logits. Substituting the training data-dependent term in Eq. (1) by the synthetic samples yields the loss func-

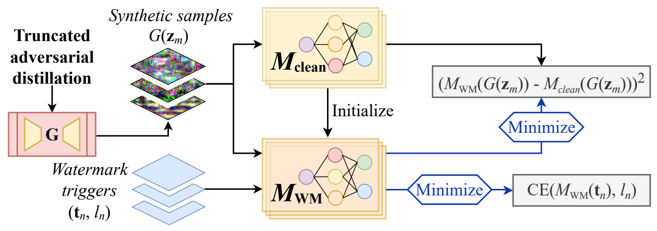
Truncated adversarial distillation
Synthetic samples $G(\mathbf{z}_m)$
G
M_{clean}
Initialize
M_{WM}
Watermark triggers $(\mathbf{t}_n, l_n)$
$(M_{\text{WM}}(G(\mathbf{z}_m)) - M_{clean}(G(\mathbf{z}_m)))^2$
Minimize
Minimize
$\text{CE}(M_{\text{WM}}(\mathbf{t}_n), l_n)$

Fig. 4. Watermark injection.

tion in data-free watermarking:

$$\mathcal{L}_{\text{DF}}(M_{\text{WM}}) = \sum_{n=1}^{N} \text{CE}(M_{\text{WM}}(\mathbf{t}_n), l_n) + \lambda \sum_{\mathbf{z}_m \sim \mathcal{N}(0,\mathbf{I})}^{M} (M_{\text{WM}}(G(\mathbf{z}_m)) - M_{\text{clean}}(G(\mathbf{z}_m)))^2 \tag{3}$$

4. EXPERIMENTS

4.1. Settings

We conducted experiments on three image classification datasets: CIFAR-10 [12], CIFAR-100 [12], Caltech-101 [13], and two DNN architectures: ResNet-18 and ResNet-34 [14].

Training M_{clean} took 200 epochs for each combination of dataset and model. In the generator training phase, we used a generator architecture comprising three convolutional layers with input dimension 256. We used SGD with momentum 0.9 and weight decay 5e-4 to update S and Adam to update G. The respective learning rates were 1e-2 and 1e-4. The ratio between training epochs for S and G in Eq. (2) was fixed at 5:1. The number of watermark triggers was set to $N = 100$. To evaluate the fidelity, we considered four settings: (S1) no adjustment, (S2) adjustment using pure noise images, (S3) adjustment using generator-generated images, and (S4) adjustment using the original training dataset. A formal comparison is given in Table 1. For S2, S3, and S4, the number of samples used for model adjustment was $M = 10N$. The hyperparameter λ in Eq. (1) and Eq. (3) was set to 5. Adam optimizer was utilized for watermarking the model.

Experiments were implemented in PyTorch, and a Nvidia GeForce RTX 3090Ti was used for GPU acceleration.

4.2. Effectiveness of Truncation

We first evaluated the effectiveness of the truncation strategy. Pure noise images were used as the watermark triggers. The baseline generator obtained from the complete DFAD schedule [10] and that trained with our strategy were used to adjust the watermarked model. We recorded the time required for training the generator in both configurations and the classification accuracy of the watermarked models on the testing dataset (the accuracy of watermark triggers was uniformly 100%). Results are shown in Table 2.

Table 1. Comparison between four adjustment methods

Setting	Data-free	Model adjustment loss
S1	✓	–
S2	✓	$\sum_{\mathbf{z}_m \sim \mathcal{N}(0,\mathbf{I})}^{M}(M_{\text{WM}}(\mathbf{z}_m) - M_{\text{clean}}(\mathbf{z}_m))^2$
S3	✓	$\sum_{\mathbf{z}_m \sim \mathcal{N}(0,\mathbf{I})}^{M}(M_{\text{WM}}(G(\mathbf{z}_m)) - M_{\text{clean}}(G(\mathbf{z}_m)))^2$
S4	×	$\sum_{m=1}^{M} \text{CE}(M_{\text{WM}}(\mathbf{x}_m), y_m)$

Table 2. Efficacy of truncation in time and fidelity.

Configuration	Complete or truncated	Time consumption	Fidelity
CIFAR-10+ResNet-18	Truncated	00:51:26	95.23%
	Complete	03:38:57	95.26%
CIFAR-10+ResNet-34	Truncated	02:14:18	95.15%
	Complete	06:17:12	95.20%
CIFAR-100+ResNet-18	Truncated	00:59:57	77.28%
	Complete	03:34:57	77.32%
CIFAR-100+ResNet-34	Truncated	02:14:13	78.14%
	Complete	06:21:17	78.14%
Caltech-101+ResNet-18	Truncated	00:28:17	78.64%
	Complete	01:25:07	78.88%
Caltech-101+ResNet-34	Truncated	00:43:54	77.29%
	Complete	01:51:42	77.94%

It is observed that the truncation strategy significantly reduces the time consumption, and the generators are still capable of adjusting the watermarked model. Further tuning of the generator by adversarial distillation after truncation is unnecessary. Therefore, the generators used in the subsequent experiments (S3) were uniformly trained with truncation.

4.3. Fidelity

To comprehensively evaluate the fidelity, we considered three categories of watermark triggers: pure noise images [15], pure noise images with overflowed pixels [16], and generator-generated images (using the generator for model adjustment). We recorded the watermarked model's classification accuracy on the testing dataset (with all models' accuracy on watermark triggers 100%). The results are shown in Fig. 5.

In all combinations, S3 achieved comparable performance with S4 and outperformed S1 and S2. S2 was better than S1, which is likely attributed to the smaller size of the images, so pure noise images can still effectively cover the input space of the samples. On large-image datasets such as Caltech-101, the advantage of S3 was significant. The experimental results demonstrate that our approach accomplishes watermark injection with negligible impact on model classification accuracy, without reliance on the original training dataset and with minimal additional time consumption.

Since parties without knowledge of the training dataset can still watermark a DNN model, defenders ought to incorporate this universal overwriting into future threat models.

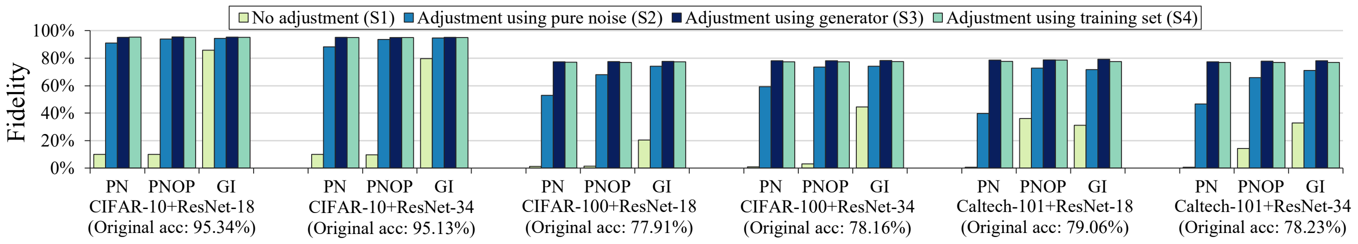

Fig. 5. Classification performance of watermarked models under different configurations. PN, PNOP, and GI respectively represent the backdoor trigger patterns as pure noises, pure noises with overflowed pixels, and generator-generated images.

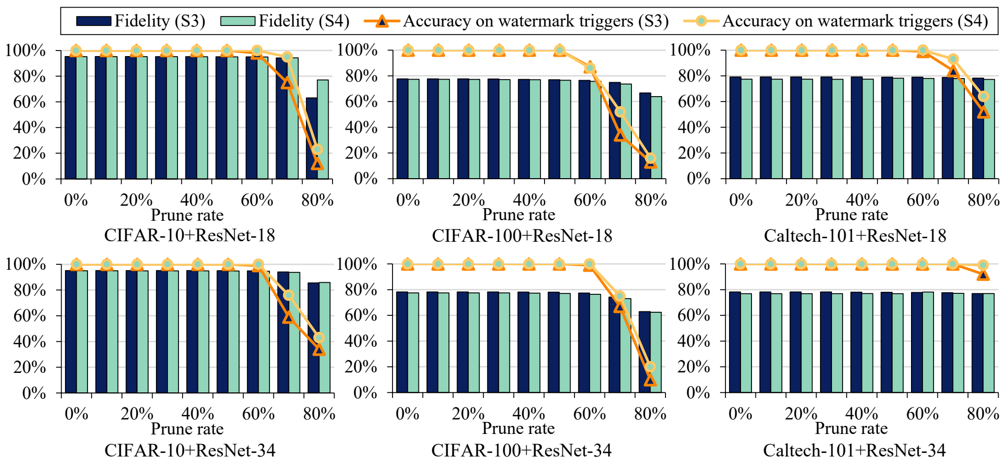

Fig. 6. Fidelity and accuracy on watermarked triggers of watermarked models under pruning.

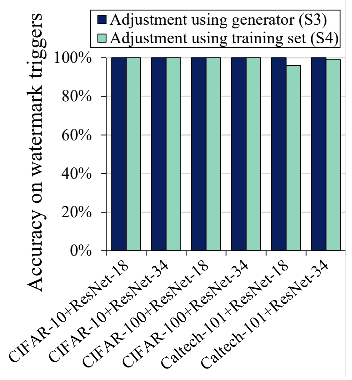

Fig. 7. Accuracy on watermark triggers of watermarked models after fine-tuning.

4.4. Robustness

Assuming that the attacker possesses partial training data and is capable of conducting removal attacks such as neuron pruning and fine-tuning. We evaluated the robustness of our watermarking method against these baseline malicious watermark removal attacks. The watermark triggers were fixed as generator-generated images that exhibited the best performance in fidelity evaluation.

Pruning. Model pruning sets some parameters inside a DNN model to zero to cut down the computation complexity [17]. It is reported that pruning can erase backdoors from DNN models, including watermark triggers [18]. Therefore, pruning is considered a baseline watermark removal attack. We conducted unstructured pruning with pruning rate varied from 0% to 80%, and recorded the pruned model's classification accuracy on the testing dataset and watermark triggers. The results are presented in Fig. 6. It is observed that our scheme S3 performed close to the data-dependent setting S4. As the accuracy on watermark triggers decreases, the model's classification accuracy is also sabotaged.

Fine-tuning. Model fine-tuning involves retraining the model using a local dataset to improve its performance on specific tasks [19]. This process can potentially reduce the accuracy on watermark triggers [20]. We assume that attackers can access a portion of the original training dataset to fine-tune the model in an attempt to erase the watermark. We randomly selected 20% of the training dataset and fine-tuned the watermarked model for 50 epochs. The results are presented in Fig. 7. All watermarked models adjusted using our method maintained a 100% accuracy for watermark triggers. This performance even slightly outperformed watermarked models adjusted with the training dataset, which experienced a decrease of 1% to 4% in watermark triggers' accuracy on Caltech-101.

5. CONCLUSION

Watermark can protect DNNs as proprietary assets in industrial settings. As the trend of privatizing training data gains momentum, watermarking techniques that can operate independently of the original training data have become preferable. Inspired by data-free adversarial distillation, we propose a black-box DNN watermarking scheme that does not rely on the original training data. We introduce a truncation strategy to significantly reduce time overhead. The proposed scheme demonstrates a comparable equivalent ability to maintain the fidelity as using the original training data. It is also robust against common watermark removal attacks, establishing itself as an effective and reliable watermarking approach.

6. REFERENCES

[1] Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh, "Embedding watermarks into deep neural networks," in *Proceedings of the 2017 ACM on international conference on multimedia retrieval*, 2017, pp. 269–277.

[2] Jie Zhang, Dongdong Chen, Jing Liao, Weiming Zhang, Huamin Feng, Gang Hua, and Nenghai Yu, "Deep model intellectual property protection via deep watermarking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 8, pp. 4005–4020, 2021.

[3] Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet, "Turning your weakness into a strength: Watermarking deep neural networks by backdooring," in *27th USENIX Security Symposium (USENIX Security 18)*, 2018, pp. 1615–1631.

[4] Jialong Zhang, Zhongshu Gu, Jiyong Jang, Hui Wu, Marc Ph Stoecklin, Heqing Huang, and Ian Molloy, "Protecting intellectual property of deep neural networks with watermarking," in *Proceedings of the 2018 on Asia conference on computer and communications security*, 2018, pp. 159–172.

[5] Ronald Kemker, Marc McClure, Angelina Abitino, Tyler Hayes, and Christopher Kanan, "Measuring catastrophic forgetting in neural networks," in *Proceedings of the AAAI conference on artificial intelligence*, 2018, vol. 32.

[6] Mauro Ribeiro, Katarina Grolinger, and Miriam AM Capretz, "Mlaas: Machine learning as a service," in *2015 IEEE 14th international conference on machine learning and applications (ICMLA)*. IEEE, 2015, pp. 896–902.

[7] Yue Li, Hongxia Wang, and Mauro Barni, "A survey of deep neural network watermarking techniques," *Neurocomputing*, vol. 461, pp. 171–193, 2021.

[8] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al., "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[9] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1701–1708.

[10] Gongfan Fang, Jie Song, Chengchao Shen, Xinchao Wang, Da Chen, and Mingli Song, "Data-free adversarial distillation," *arXiv preprint arXiv:1912.11006*, 2019.

[11] Paul Micaelli and Amos J Storkey, "Zero-shot knowledge transfer via adversarial belief matching," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[12] Alex Krizhevsky, Geoffrey Hinton, et al., "Learning multiple layers of features from tiny images," 2009.

[13] Li Fei-Fei, Robert Fergus, and Pietro Perona, "One-shot learning of object categories," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 4, pp. 594–611, 2006.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[15] Renjie Zhu, Xinpeng Zhang, Mengte Shi, and Zhenjun Tang, "Secure neural network watermarking protocol against forging attack," *EURASIP Journal on Image and Video Processing*, vol. 2020, pp. 1–12, 2020.

[16] Huiying Li, Emily Willson, Haitao Zheng, and Ben Y Zhao, "Persistent and unforgeable watermarks for deep neural networks," *arXiv preprint arXiv:1910.01226*, vol. 2, 2019.

[17] Song Han, Jeff Pool, John Tran, and William Dally, "Learning both weights and connections for efficient neural network," *Advances in neural information processing systems*, vol. 28, 2015.

[18] Nils Lukas, Edward Jiang, Xinda Li, and Florian Kerschbaum, "Sok: How robust is image classification deep neural network watermarking?," in *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2022, pp. 787–804.

[19] Nima Tajbakhsh, Jae Y Shin, Suryakanth R Gurudu, R Todd Hurst, Christopher B Kendall, Michael B Gotway, and Jianming Liang, "Convolutional neural networks for medical image analysis: Full training or fine tuning?," *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1299–1312, 2016.

[20] Xinyun Chen, Wenxiao Wang, Chris Bender, Yiming Ding, Ruoxi Jia, Bo Li, and Dawn Song, "Refit: a unified watermark removal framework for deep learning systems with limited data," in *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, 2021, pp. 321–335.