FOSTERING THE ROBUSTNESS OF WHITE-BOX DEEP NEURAL NETWORK WATERMARKS BY NEURON ALIGNMENT

Fang-Qi Li, Shi-Lin Wang, Senior Member, IEEE, Yun Zhu*

{solour_lfq,wsl,scott0518}@sjtu.edu.cn
School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University.

ABSTRACT

The wide application of deep learning techniques is boosting the regulation of deep learning models, especially deep neural networks (DNN), as commercial products. A necessary prerequisite for such regulations is identifying the owner of deep neural networks, which is usually done through the watermark. Current DNN watermarking schemes, particularly white-box ones, are uniformly fragile against a family of functionality equivalence attacks, especially the neuron permutation. This operation can effortlessly invalidate the ownership proof and escape copyright regulations. To enhance the robustness of white-box DNN watermarking schemes, this paper presents a procedure that aligns neurons into the same order as when the watermark is embedded, so the watermark can be correctly recognized. This neuron alignment process significantly facilitates the functionality of established deep neural network watermarking schemes.

Index Terms— Machine learning security, deep neural network watermark, neuron alignment.

1. INTRODUCTION

Deep learning models have made significant achievements in domains ranging from computer vision [1] to signal processing [2, 3]. Since deep neural networks (DNN) can provide high-quality service, they have been treating as commercial products and intellectual properties. One necessary condition for commercializing DNNs is identifying their owners. DNN watermark is an acknowledged technique for ownership verification (OV). By embedding owner-dependent information into the DNN and revealing it under an OV protocol [4], the owner of the DNN can be uniquely recognized.

If the pirated model can only be interacted as a black-box then backdoor-based DNN watermarking schemes are the only option. They encode the owner's identity into backdoor triggers by pseudorandom mapping [5], variational autoencoder [6], or deep image watermarking [7]. Adversarial samples [8] and penetrative triggers [9] have been designed to defend against adversarial tuning and filtering. However, in the realistic setting, an adversary can ensemble multiple DNN models or adding extra rules to invalidate backdoors.

White-box DNN watermarking schemes have better performance regarding unambiguity and forensics given unlimited access to the pirated model. They embed the owner's identity into the model's weight [10], its intermediate outputs for specific inputs [11], etc. The white-box assumption holds for many important scenarios such as model competitions, engineering testing, and lawsuits.

Despite their privileges, white-box DNN watermarking schemes are haunted by the functionality equivalence attack, in particular, the neuron permutation attack [12]. The watermark is uniformly tangled with the parameters of neurons, so the adversary can invalidate it and pirate the model by permutating neurons without affecting the model's performance.

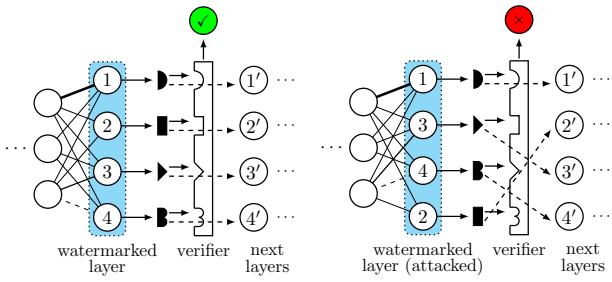
To cope with this threat and foster the robustness of white-box DNN watermarking schemes, we propose a neuron alignment framework. By encoding the neurons and generating proper triggers, the order of neurons can be recovered. Then the watermark can be correctly retrieved and the ownership is secured. The contribution of this paper is threefold:

- We propose a DNN protection framework against the neuron permutation attack. To the best of our knowledge, this is the first attemp in defending such threat.
- By aligning neurons, the proposed framework can recover the order of neurons and can be seamlessly combined with established watermarking schemes.
- Experiments have justified the efficacy of our proposal.

2. THE MOTIVATION

In OV, the verifier module takes the parameters/outputs of neurons as its input. An adversary can shuffle homogeneous neurons (whose forms and connections to previous layers are identical) using a permutation operator $\mathbf{P}$ such that the input from the verifier's perspective is no longer an identification proof. The impact to the subsequent processing can be canceled by applying $\mathbf{P}^{-1}$ before the next layer so the functionality of the DNN remains intact. This *neuron permutation attack* is examplified in Fig. 1. One solution to this threat is

This work was supported by the National Natural Science Foundation of China (61771310). Shi-Lin Wang is the corresponding author.



(a) The normal verification. (b) The verification under attack.

Fig. 1. A neuron permutation attack, $\mathbf{P} = (2, 3, 4)$.

designing verifier modules that are invariant to the permutation of its inputs, which is challenging due to the loss of information and detached from all established white-box DNN watermarking schemes. Instead, it is desirable that we can recognize $\mathbf{P}$ and cancel its influence by aligning the neurons into their original order. To perform aligning, we encode neurons by their scalar outputs, which are invariant under any permutation in precedent layers. The neurons' outputs on training data, which are supposed to be the most diversified, are clustered into several centroids as signals. To get rid of the deviation from a neuron's normal outputs to the centroids, some trigger samples are generated to correctly evoke these signal outputs as a neuron's identifier code. To guarantee robust reconstruction, such encoding also needs to have good error-correcting ability against model tuning.

3. THE PROPOSED METHOD

Assume that the watermarked layer contains N homogeneous neurons, the code for a neuron is its outputs on a specialized collection of inputs, known as *triggers*. Given the triggers, the owner can obtain the codes for neurons and align them properly. What remains to be specified is the encoding scheme and the generation of triggers.

3.1. Neuron encoding

Denote the length of the code by T and the size of the alphabet by K. Each trigger invokes one output from each neuron and is mapped into one position in each neuron's code, so T is also the number of triggers. Denote the output of the n-th neuron in the watermarked layer for an input $\mathbf{x}$ as $y_n(\mathbf{x})$, let the training dataset be $\{\mathbf{x}_d\}_{d=1}^{D}$. The normal output space of neurons in the watermarked layer is split into K folds. The centroid of the k-th fold, c_k, is computed as:

$$c_k = \frac{\sum_{n=1}^{N}\sum_{d=1}^{D} y_n(\mathbf{x}_d)\cdot\mathbb{I}[y_n(\mathbf{x}_d)\in\mathcal{C}_k]}{\lceil ND/K\rceil}, \qquad (1)$$

where $\mathcal{C}_k$ is the range of the k-th fold containing the $\lceil\frac{NDk}{K}\rceil$-th to the $\lceil\frac{ND(k+1)}{K}\rceil$-th smallest elements in $\{y_n(\mathbf{x}_d)\}_{d=1,n=1}^{D,N}$.

This process is demonstrated in Fig. 2.

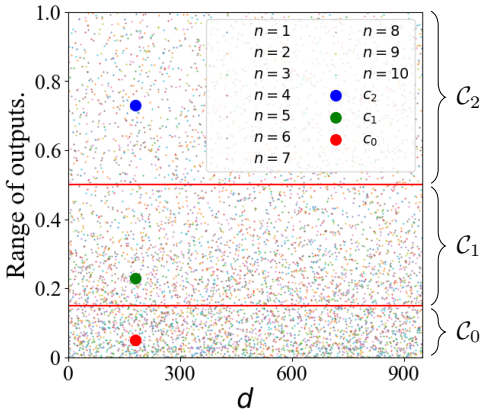


Fig. 2. Splitting the output space, D=1024, N=10, and K=3.

Having determind the centroids, the n-th neuron is assigned a code $\mathbf{r}_n \in \{0, 1, \cdots, K\}^T$, the dictionary $\{\mathbf{r}_n\}_{n=1}^{N}$ is spanned using a error correction coding scheme [13]. It is expected that the output of the n-th neuron on the t-th trigger, $y_n(\mathbf{x}_t)$, is close to $c_{\mathbf{r}_{n,t}}$. To enable error correcting within fewer than $T_{\text{corrupted}}$ positions and each position shifts within at most $K_{\text{corrupted}}$ folds, it is necessary that T and K satistisfy:

$$N\cdot\left(\sum_{t=1}^{T_{\text{corrupted}}}\binom{T}{t}\cdot K_{\text{corrupted}}^{t}\right)\le K^T. \qquad (2)$$

3.2. Trigger generation

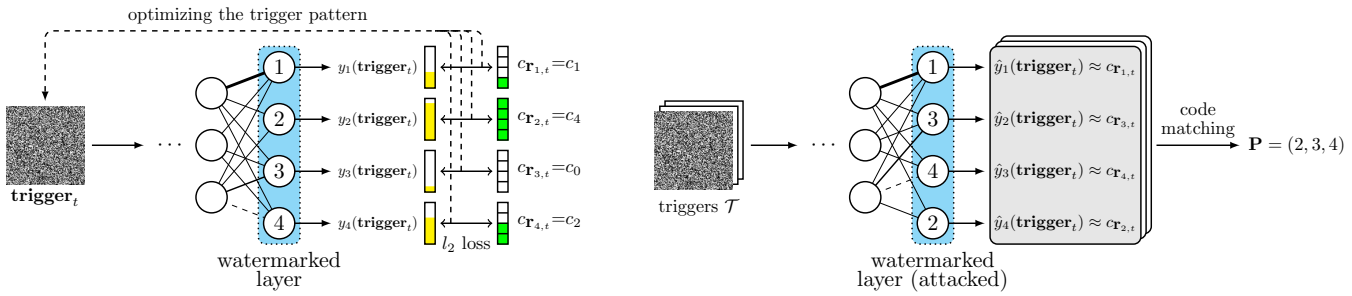
To generate triggers that correctly evoke the neurons' outputs as codes, we adopt the method in forging adversarial samples [14]. Concretely, the t-th trigger $\mathbf{trigger}_t$ is obtained by minimizing the following loss:

$$\mathcal{L}_1(\mathbf{trigger}_t) = \sum_{n=1}^{N}\|y_n(\mathbf{trigger}_t) - c_{\mathbf{r}_{n,t}}\|_2^2, \qquad (3)$$

in which the parameters of the entire DNN are frozen. To increase the robustness of this encoding against the adversary's tuning, we suggest that $\mathbf{t}_m$ be optimized w.r.t. the adversarially tuned versions of the watermarked DNN as well. Let $\{y_n^j\}_{j=0}^{J}$ denotes the mapping introduced by the n-th neuron under all J kinds of tuning (j=0 represents the original model), the loss function becomes:

$$\mathcal{L}_2(\mathbf{trigger}_t) = \sum_{j=0}^{J}\sum_{n=1}^{N}\|y_n^j(\mathbf{trigger}_t) - c_{\mathbf{r}_{n,t}}\|_2^2, \qquad (4)$$

The collection of all triggers $\mathcal{T} = \{\mathbf{trigger}_t\}_{t=1}^{T}$ forms the owner's evidence for neuron alignment. This process is demonstrated in Fig. 3 (a).



(a) Generating the t-th trigger, the code for the t-th position is `(1402)`. (b) Aligning neurons from the intermediate output code `(1024)` on the t-th trigger.

Fig. 3. The trigger generation process and the neuron alignment process.

3.3. Neuron alignment

Given the white-box access to the suspicious DNN, the owner can recover the order of neurons in the watermarked layer by the following steps: (i) Inputting all triggers $\mathcal{T}$ sequentially into the DNN. (ii) Recording the outputs of the n-th neurons in the watermarked layer as $\{\hat{y}_n(\mathbf{trigger}_t)\}_{t=1}^{T}$. (iii) Transcripting $\hat{y}_n(\mathbf{trigger}_t)$ into a code $\hat{\mathbf{r}}_n \in \{0, 1, \cdots, K\}^T$:

$$\hat{\mathbf{r}}_{n,t} = \arg\min_{k} \{\|\hat{y}_n(\mathbf{trigger}_t) - c_k\|_2\}.$$

(iv) Transcripting $\hat{\mathbf{r}}_n$ into an index i_n:

$$i_n = \arg\min_{n'} \left\{ \sum_{t=1}^{T} |\mathbf{r}_{n',t} - \hat{\mathbf{r}}_{n,t}| \right\},$$

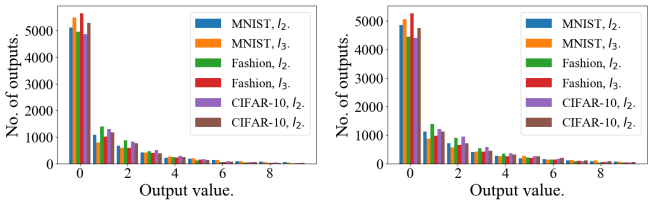
Finally, the owner aligns all neurons according to their indices and conducts OV using its white-box watermark verifier. This process is demonstrated in Fig. 3 (b).

Remark: An adaptive adversary might breach this alignment by rescaling the weights across layers. This can be neutralized by normalizing parameters before alignment or adopting a smaller K to ensure distinguishability.

4. EXPERIMENTS AND DISCUSSIONS

4.1. Settings

To examine the validity of the proposed framework, we selected two DNN structures, ResNet-18 and ResNet-50 [15]. In each DNN, we selected the second (l_2) and the third (l_3) layers to be watermarked. l_2 contains 64 homogeneous neurons and l_3 contains 128 ones. For these convolutional layers, the output where neurons are recognized and decoded is the value of one specific pixel. Both networks were trained for three computer vision tasks: MNIST [16], FashionMNIST [17], and CIFAR-10 [18]. The training of all DNN backbones and triggers was implemented by Adam [19] with `PyTorch`.



(a) ResNet-18. (b) ResNet-50.

Fig. 4. Distributions of watermarked layers' outputs.

N \ T	20	40	60	80	100	120	140	160
64	4	12	21	29	38	47	56	65
128	4	11	20	28	37	46	55	64

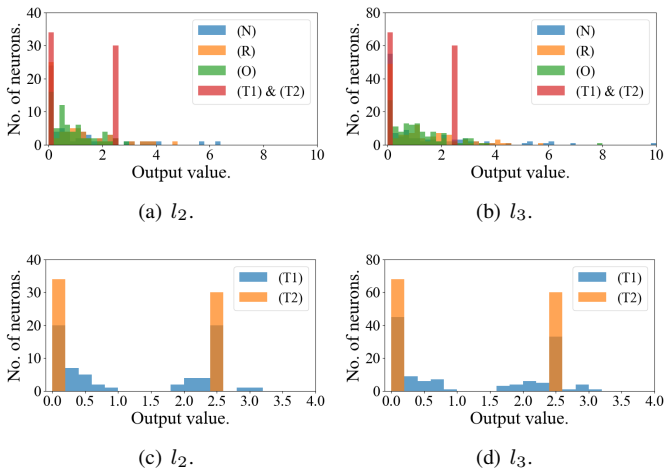
Table 1. The maximal number of flipped positions that can be corrected, $T_{\text{corrupted}}$, w.r.t. N and T, $K = 2$.

4.2. The configuration of parameters

To compute the centroids, we measured the distributions of outputs for watermarked layers on normal samples, results are demonstrated in Fig. 4. These distributions remained almost invariant to the selection of dataset, network, and layer. To ensure maximal distinguishability, we adopted K=2 and computed the centroids c_0=0, c_1=2.5 by (1). With $K_{\text{corrupted}}$=1, the error correcting ability computed from (2) is shown in Table. 1. We adopted T=160 in the following experiments, where flipped bits up to 40% would not compromise the unique decoding.

4.3. Comparative studies

For comparison, we compared five candidate schemes for trigger selection. **(N)**: **N**ormal samples from the training dataset. **(R)**: **R**andom noises. **(O)**: **O**ut-of-dataset samples. **(T1)**: **T**riggers generated be minimizing $\mathcal{L}_1$ in (3). **(T2)**:



(a) l_2. (b) l_3. (c) l_2. (d) l_3.

Fig. 5. The distribution of watermarked layers' outputs for different triggers. In (c)(d), the DNN has been fine-tuned.

Metric	(N)	(R)	(O)	(T1)	(T2)
Inter-cluster (↑).	2.4	1.9	1.5	**2.5**	**2.5**
Intra-cluster (↓).	1.3	0.8	0.8	**0.1**	0.2
Accuracy (%).	1.0	2.3	1.3	**98.4**	97.2

Table 2. The statistics of neurons' outputs.

Triggers generated by minimizing $\mathcal{L}_2$ in (4) with J=6 involving three rounds of fine-tuning and three of neuron-pruning. For **(N)(R)(O)**, the centroids were also selected by (1) and the code of each neuron at the t-th position was assigned as the index of the closest centroid to its output on the t-th input.

The outputs of neurons in ResNet-50 trained on CIFAR-10 for one input are shown in Fig. 5(a)(b). From which we noticed that the outputs w.r.t. **(T1)(T2)** concentrated to K=2 centers. (The percentage of neurons outputting approximately 0 or 2.5 was not strictly 50%, since the outputs of around 2% of neurons were uniformly zero.) Therefore, the code of neurons under **(T1)(T2)** can be unambiguously retrieved. Numerically, we computed the averaged inter/intra-cluster distance for all trigger patterns with two clusters obtained by K-means [20] and the accuracy of aligning against random shuffling on l_3, the results are listed in Table. 2. From which we justified that the codes derived by **(T1)(T2)** are more informative. After fine-tuning, the distributions of outputs under **(T1)(T2)** were differentiated as shown in Fig. 5(c)(d), so **(T2)** is more robust against model tuning.

4.4. The performance of watermarking backends

To study the performance of white-box DNN watermarking schemes after the neuron permutation attack and alignment, we considered four state-of-the-art watermarking schemes:

Attack	Uchida	Fan	Residual	MTLSign
(NP)	0.0, (95.7,95.5)	0.0, (95.1,95.1)	0.0, (98.3,98.0)	0.0, (99.1,98.6)
(FTP)	0.0, (69.4,90.3)	0.0, (74.3,75.2)	0.0, (82.7,87.4)	0.0, (79.9,87.9)
(NPP)	0.0, (54.4,74.3)	0.0, (67.9,76.5)	0.0, (77.7,82.6)	0.0, (75.4,83.9)
Attack	Uchida	Fan	Residual	MTLSign
(NP)	0.0, (96.3,95.4)	0.0, (96.0,95.6)	0.0, (99.1,99.4)	0.0, (98.7,98.7)
(FTP)	0.0, (70.1,89.1)	0.0, (74.9,75.9)	0.0, (84.6,88.0)	0.0, (79.6,90.5)
(NPP)	0.0, (58.9,72.5)	0.0, (63.7,73.4)	0.0, (79.8,81.3)	0.0, (75.9,84.9)

Table 3. The performance of watermarking backends after neuron alignments for ResNet-18 and ResNet-50. The results in each entry are: the accuracy of OV after the attack and its increase (in %) with alignment by ((**T1**), (**T2**)), averaged across three datasets.

Uchida [10], Fan [21], Residual [22], and MTLSign [11]. All watermarks were embedded into both l_2 and l_3.

We conducted three attacks to the watermarked layer: **(NP)**: **N**euron **P**ermutation; **(FTP)**: **F**ine-**T**uning and neuron **P**ermutation; **(NPP)** **N**euron-**P**runing and **P**ermutation. Then we applied neuron alignment and recorded the percentage of correct verifications from the watermarking backends in 1,000 instances, results are summarized in Table. 3. Without neuron alignment, any permutation-based attack can reduce the OV accuracy to 0.0%. After alignment, the accuracy in all cases increased significantly. Compared with **(T1)**, **(T2)** is more robust against tuning and pruning in better reconstructing the order of neurons. Therefore, by adopting the neuron alignment framework, the security levels of these watermarking schemes are substantially increased. Meanwhile, the trigger generation process does not modify the original DNN, so it can be parallelized and would not bring extra damage to the protected DNN.

5. CONCLUSIONS

We propose a neuron alignment framework to enhance established white-box DNN watermarking schemes. Clustering and error-correcting encoding are adopted to ensure the availability and distinguishability of neuron encoding. Then we use a generative method to forge triggers that can correctly and robustly reveal the neurons' order. Experiments demonstrate the effectiveness of our framework against the neuron permutation attack, a realistic threat to OV for DNN.

6. REFERENCES

[1] Nam Le, Honglei Zhang, Francesco Cricri, Ramin Ghaznavi-Youvalari, and Esa Rahtu, "Image coding for machines: an end-to-end learned approach," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 1590–1594.

[2] Cem Subakan, Mirco Ravanelli, Samuele Cornell, Mirko Bronzi, and Jianyuan Zhong, "Attention is all you need in speech separation," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 21–25.

[3] Zhepei Wang, Ritwik Giri, Umut Isik, Jean-Marc Valin, and Arvindh Krishnaswamy, "Semi-supervised singing voice separation with noisy self-training," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 31–35.

[4] Fangqi Li, Shilin Wang, and Alan Wee-Chung Liew, "Regulating ownership verification for deep neural networks: Scenarios, protocols, and prospects," *IJCAI Workshop*, 2021.

[5] 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, no. 1, pp. 1–12, 2020.

[6] Zheng Li, Chengyu Hu, Yang Zhang, and Shanqing Guo, "How to prove your model belongs to you: a blind-watermark based framework to protect intellectual property of dnn," in *Proceedings of ACSAC*, 2019, pp. 126–137.

[7] 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*, 2021.

[8] Erwan Le Merrer, Patrick Perez, and Gilles Trédan, "Adversarial frontier stitching for remote neural network watermarking," *Neural Computing and Applications*, vol. 32, no. 13, pp. 9233–9244, 2020.

[9] Fang-Qi Li and Shi-Lin Wang, "Persistent watermark for image classification neural networks by penetrating the autoencoder," in *2021 IEEE International Conference on Image Processing (ICIP)*, 2021, pp. 3063–3067.

[10] 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.

[11] Fangqi Li, Lei Yang, Shilin Wang, and Liew Alan Wee-Chung, "Leveraging multi-task learning for unambiguous and flexible deep neural network watermarking," *AAAI SafeAI Workshop*, 2021.

[12] Nils Lukas, Edward Jiang, Xinda Li, and Florian Kerschbaum, "Sok: How robust is image classification deep neural network watermarking? (extended version)," 2021.

[13] Nandivada Sridevi, K. Jamal, and Kiran Mannem, "Implementation of error correction techniques in memory applications," in *2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*, 2021, pp. 586–595.

[14] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[15] 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.

[16] Li Deng, "The mnist database of handwritten digit images for machine learning research [best of the web]," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[17] Han Xiao, Kashif Rasul, and Roland Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

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

[19] Zijun Zhang, "Improved adam optimizer for deep neural networks," in *2018 IEEE/ACM 26th International Symposium on Quality of Service*. IEEE, 2018, pp. 1–2.

[20] Fang-Qi Li, Shi-Lin Wang, and Gong-Shen Liu, "A bayesian possibilistic c-means clustering approach for cervical cancer screening," *Information Sciences*, vol. 501, pp. 495–510, 2019.

[21] Lixin Fan, Kam Woh Ng, Chee Seng Chan, and Qiang Yang, "Deepip: Deep neural network intellectual property protection with passports," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[22] Hanwen Liu, Zhenyu Weng, and Yuesheng Zhu, "Watermarking deep neural networks with greedy residuals," in *International Conference on Machine Learning*. PMLR, 2021, pp. 6978–6988.