Effective Ambiguity Attack Against Passport-based DNN Intellectual Property Protection Schemes through Fully Connected Layer Substitution Supplementary Material

Yiming Chen[1], Jinyu Tian[2], Xiangyu Chen[1,3], and Jiantao Zhou[1,†]
[1]State Key Laboratory of Internet of Things for Smart City
Department of Computer and Information Science, University of Macau
[2]Faculty of Innovation Engineering, Macau University of Science and Technology
[3]Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences
{yc17486, jtzhou}@umac.mo, jytian@must.edu.mo, chxy95@gmail.com

In this Appendix, we give more details on generalizing our attack to other DNN watermark methods.

A. Details of attack with CERB on other watermark methods

In Section 4.3 of the main text, we demonstrate that our method can be well generalized to attack other DNN watermark methods. Here we provide more implementation details.

A.1. Uchida

Uchida *et al*. [5] embedded a watermark into the model weights. The extraction process from the weights is as follows:

$$\mathbf{s} = \text{step}(\mathbf{A} * Avg(\mathbf{W})). \tag{1}$$

Specifically, after performing the average pooling $Avg(\cdot)$, the flattened weight is then transformed by matrix $\mathbf{A}$ to a C-bit signature, where the step function $\text{step}(\cdot)$ returns 1 when the condition is non-negative and 0 otherwise. Through a regular binary cross entropy loss, the extracted watermark can be regularized to match a signature.

Attack Settings: We adopt the embedding process in Uchida where the watermark is embedded into a Wide Residual Network [6] (WRN-10-4) with a depth of 10 and a widen factor of 4 trained on CIFAR10 [3] for 200 training epochs. The batch size for training is set to 64 and the initial learning rate is 0.1. The learning rate decays at epoch 60, 120 and 160 with a rate of 0.2. We use stochastic gradient descent (SGD) with a momentum of 0.9 for optimization. The watermark is injected into the third convolutional layer weights of the network. The signature is randomly initialized from $\{0, 1\}$ with a length of 256.

Attack Implementation: In our launched attack, we insert a CERB after the convolutional layer weight and use the output weight from the CERB for subsequent convolutional operation. We freeze the parameters in other layers and only optimize the watermark embedded convolutional weight and the introduced CERB parameters. We use the same embedding process as the [5] so that the weights after the attack can match with the new signature chosen by the attacker. The learning rate is initialized as 0.01 and reduced by a factor of 0.2 at epoch 60, 120 and 160, respectively.

A.2. DeepSigns

Instead of using model weights for embedding, DeepSigns [1] used the output activations from specific layers. Particularly, DeepSigns regularized the activations of input samples $\mathcal{D}_c = \{(x_i, y_i)|y_i = c\}_{i=1}^{n_c}$ from the specific class c. The regularizer can be formulated as:

$$\mathbf{s} = \text{step}(\mathbf{A} * Avg(f^l(x_i))), \tag{2}$$

where $f^l(\cdot)$ represents the feature of the l-th layer. The other steps are the same as those of Uchida's method.

Attack Settings: The watermark was hidden in the flattened features before the last linear layer in a WRN-10-4 network trained on CIFAR10. The batch size for training is 64 and the embedding process takes 200 epochs of training. The dimension of the feature in this experiment is 512. The signature length is 256 and initialized from $\{0, 1\}$.

Attack Implementation: To ensure that the feature can be verified by another signature specified by an attack, We choose to modify the normalization layer ahead of this feature with our CERB structure. Specifically, the CERB is inserted after the affine factors in this normalization layer.

†Corresponding author.

We only update the affine factors along with the CERB parameters using the same embedding process as [1], so that the features after the attack can be successfully verified with the new signature. The SGD optimizer is chosen for optimization with a learning rate of 0.01. The whole process takes 200 epochs for training.

A.3. Greedy-Residual

In Greedy-Residual [4], the signature was embedded in the model weights. We here briefly introduce the extraction process. After the one-dimensional average pooling (Avg_pool_1D), the flattened model weight $\hat{\mathbf{W}}$ is transformed to a two-dimension matrix $\mathbf{W}' \in \mathbb{R}^{C \times C'}$:

$$\mathbf{W}' = Avg\_pool\_1D(\hat{\mathbf{W}}). \tag{3}$$

Then we only keep the parameters in $\mathbf{W}'$ with larger absolute values in the 2-nd dimension by a ratio of η (we use the default ratio of 0.5 as set in [4]). Subsequently, the matrix is averaged on the 2-nd dimension and transformed to a C-dimensional vector whose signs are regularized to match with a C-bit signature.

Attack Settings: We follow the default setting and focus on the ResNet18 trained on Caltech-256 [2] dataset for 200 epochs. The optimizer is chosen to be SGD. The initial learning rate is 0.01 and decays at epoch 100 and epoch 150 with a rate of 0.1. The watermark is embedded in the first convolutional layer weights. The signature is matched with the sign of the parameter that is randomly initialized with $\{-1, 1\}$ with a length of 256.

Attack Implementation: We modify the normalization layer behind the target convolutional layer by inserting a CERB block after the affine factors. Similar to the aforementioned attacks, we freeze parameters in other layers and only update affine factors in this normalization layer and the CERB parameters. The learning rate in our attack starts at 0.001 and reduced by a factor of 0.1 after epoch 100 and epoch 150. We conduct the attack for 200 epochs.

References

[1] Bita Darvish Rouhani, Huili Chen, and Farinaz Koushanfar. Deepsigns: An end-to-end watermarking framework for ownership protection of deep neural networks. In *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 485–497, 2019. 1, 2

[2] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007. 2

[3] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 1

[4] Hanwen Liu, Zhenyu Weng, and Yuesheng Zhu. Watermarking deep neural networks with greedy residuals. In *Proceedings of the IEEE International Conference on Machine Learning*, pages 6978–6988, 2021. 2

[5] Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh. Embedding watermarks into deep neural networks. In *Proceedings of the ACM on International Conference on Multimedia Retrieval*, pages 269–277, 2017. 1

[6] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 1