A Protection Method of Trained CNN Model Using Feature Maps Transformed With Secret Key From Unauthorized Access

MaungMaung AprilPyone and Hitoshi Kiya
Tokyo Metropolitan University, Tokyo, Japan

***Abstract*—In this paper, we propose a model protection method for convolutional neural networks (CNNs) with a secret key so that authorized users get a high classification accuracy, and unauthorized users get a low classification accuracy. The proposed method applies a block-wise transformation with a secret key to feature maps in the network. Conventional key-based model protection methods cannot maintain a high accuracy when a large key space is selected. In contrast, the proposed method not only maintains almost the same accuracy as non-protected accuracy, but also has a larger key space. Experiments were carried out on the CIFAR-10 dataset, and results show that the proposed model protection method outperformed the previous key-based model protection methods in terms of classification accuracy, key space, and robustness against key estimation attacks and fine-tuning attacks.**

I. Introduction

Convolutional neural networks (CNNs) are a type of deep neural network (DNN) inspired by the human visual system and are ubiquitous in the field of computer vision. Recent advances in deep learning show that CNNs have lead to major breakthroughs due to its effectiveness and efficiency [1]. Current commercial applications in image recognition, object detection and semantic segmentation are primarily powered by CNNs. Therefore, CNNs have become *de facto* standards for visual recognition systems in many different applications.

However, training successful CNNs requires three ingredients: huge amount of data, GPU-accelerated computing resources and efficient algorithms, and is not a trivial task. For example, the dataset for ImageNet Large Scale Visual Recognition Challenge (ILSVRC 2012) contains about 1.28 million images, and training on such a dataset takes days and weeks even on GPU-accelerated machines. In fact, collecting images and labeling them are also costly, and will also consume a massive amount of resources. Moreover, algorithms used in training a CNN model may be patented or have restricted licenses. Therefore, trained CNNs have great business value. Considering the expenses necessary for the expertise, money, and time taken to train a CNN model, a model should be regarded as a kind of intellectual property (IP).

There are two aspects of IP protection for DNN models: ownership verification and access control. The former focuses on identifying the ownership of the models and the latter addresses protecting the functionality of DNN models from unauthorized access. Ownership verification methods are inspired by digital watermarking, and embed watermarks into DNN models so that the embedded watermarks are used to verify the ownership of the models in question [2]–[9]. Although the above watermarking methods can facilitate the identification of the ownership of the models, in reality, a stolen model can be exploited in many different ways. For example, an attacker can use a model for their own benefit without arousing suspicion, or a stolen model can be used for model inversion attacks [10] and adversarial attacks [11]. Therefore, it is crucial to investigate mechanisms to protect DNN models from authorized access and misuse. In this paper, we focus on protecting a model from misuse when it has been stolen (i.e., access control).

A method to model protection against unauthorized access was inspired by adversarial examples and proposed to utilize secret perturbation to control the access of a model [12]. Another study introduced a secret key to protect a model [13], [14]. The secret key-based protection method [14] uses a key-based transformation which was originally used by an adversarial defense in [15], which was in turn inspired by perceptual image encryption methods [16]–[23]. This model protection method utilizes a secret key in such a way that a stolen model cannot be used to its full capacity without a correct secret key. However, in the previous model protection method [14], when a large block size is used, the accuracy drops, and when a small block size is used, key estimation attacks are possible due to the relatively small key space.

Therefore, for the first time, in this paper, we propose a model protection method by applying key-based transformation to feature maps. The proposed model protection method not only achieves a high classification accuracy (i.e., almost the same as non-protected accuracy), but also increases the key space substantially. We make the following contributions in this paper.

- We propose a model protection method with a secret key which improves the classification accuracy and increases the key space.
- We conduct relevant attacks to verify the effectiveness of the proposed model protection method.

In experiments, the proposed model protection method is confirmed to outperform the previous key-based model protection methods.

II. Related Work

A. Ownership Verification

Ownership verification is a concept to protect intellectual property of DNN models, in which digital watermarking techniques are adopted to embed watermarks into DNN models like copyright protection of media contents. The ownership is verified by using the extracted watermark to detect the intellectual property of copyright infringement.

There are mainly two approaches in DNN model watermarking: white-box and black-box. In white-box methods, a watermark is embedded to model weights by an embedding regularizer during training. Therefore, the access to the model weights is required for extracting the watermark embedded in the model as in [2], [5], [6], [24]. In black-box model watermarking methods, an inspector observes the input and output of a model in doubt to verify the ownership as in [3], [4], [6]–[9]. Thus, the access to the model weights is not required to verify the ownership in the black-box approaches.

Model watermarking methods in general focus on identifying the ownership only when the model is in question. The functionality of the model is not protected regardless of the ownership. Therefore, methods for protecting DNN models from unauthorized use are put forward beyond the ownership verification.

B. Access Control

One straightforward way of protecting a model from illicit use is to encrypt the trained model weights by the traditional cryptographic methods such as advanced encryption standard (AES). In this case, to be able to use the protected model, rightful users have to decrypt the model. There are millions of parameters in modern DNN models, so encrypting/decrypting all the parameters is computationally expensive under the traditional cryptography in general. Besides, once the model is decrypted, it becomes vulnerable for IP thefts. Therefore, researchers have proposed to embed a key to the model's structure by other means. In the literature, there are two prior methods for protecting DNN models against unauthorized access.

The first method [12] utilizes an anti-piracy transform module which is a secret perturbation network in such a way that the secret perturbation is crucial to the model's decision. In other words, only the rightful users who have access to the secret perturbation can use the model properly. However, this method [12] requires to training a perturbation network along with the classification network so that the optimal perturbation can be learned. In addition, the classification accuracy of the method [12] slightly drops compared to non-protected models under the same training settings.

The second method [14] adopts a block-wise transformation with a secret key from an adversarial defense [15] to protect the model against unauthorized access. Images are transformed by a block-wise transformation with a secret key and transformed images are used for training and testing a model [14]. When using a large block size, the second method drops the classification accuracy, and key estimation attacks are possible when using a small block size. Therefore, in this paper, we aim to improve these issues on the classification accuracy and key space.

III. Proposed Model Protection Method

A. Overview

In the previous key-based model protection [14], input images are transformed by a block-wise transformation with secret key K prior to training and testing a model. Key K belongs to the model owner, and it needs to be stored at the model deployment (e.g., service provider). Figure 1a depicts the prediction pipeline of the model with the previous key-based model protection [14]. In contrast, instead of transforming an input image, one or more feature maps in the network are transformed by a block-wise transformation with secret key K in the proposed model protection (Fig. 1b). In the figure, the ResNet-18 is used as an example for illustrating the proposed method, and it can be replaced with other CNN architectures.

An overview of the proposed model protection is depicted in Fig. 2. A block-wise transformation with a secret key is applied to feature maps in the network (e.g., ResNet-18), where the modified network is trained by using secret key K as shown in Fig. 2a. The model predicts a test image correctly only for authorized users with secret key K, and the model provides incorrect predictions for unauthorized users with incorrect key K' (Fig. 2b). Accordingly, a stolen model cannot be used to its full capacity when secret key K is not available.

B. Block-Wise Transformation with Secret Key for Feature Maps

We utilize a block-wise transformation, Pixel Shuffling from [14]. In this paper, a feature map x in a dimension of $(c \times h \times w)$, where c is the number of filters, h is the height, and w is the width of the feature map, is transformed with key K. There are four steps in the process of transforming a feature map as shown in Fig. 3.

1) **Segment into blocks:** Segment x into blocks with a size of M such that $\{B_{(1,1)}, \ldots, B_{(\frac{h}{M}, \frac{w}{M})}\}$.
2) **Flatten blocks:** Flatten each block $B_{(i,j)}$ into a vector $b_{(i,j)} = [b_{(i,j)}(1), \ldots, b_{(i,j)}(c \times M \times M)]$.
3) **Transform blocks:** First, generate a random permutation vector $v = [v_1, \ldots, v_k, \ldots, v_{k'}, \ldots, v_{c \times M \times M}]$ with key K, where $v_k \neq v_{k'}$ if $k \neq k'$, and shuffle every vector $b_{(i,j)}$ with v as
$$b'_{(i,j)}(k) = b_{(i,j)}(v_k), \tag{1}$$
to obtain a shuffled vector,
$b'_{(i,j)} = [b'_{(i,j)}(1), \ldots, b'_{(i,j)}(c \times M \times M)]$.
4) **Concatenate blocks:** Concatenate the shuffled vectors to form a transformed feature map x' in the dimension of $(c \times h \times w)$.

If more than one feature map are required to be transformed, K is applied to each feature map, and the above same process is repetitively carried out for all desired feature maps.

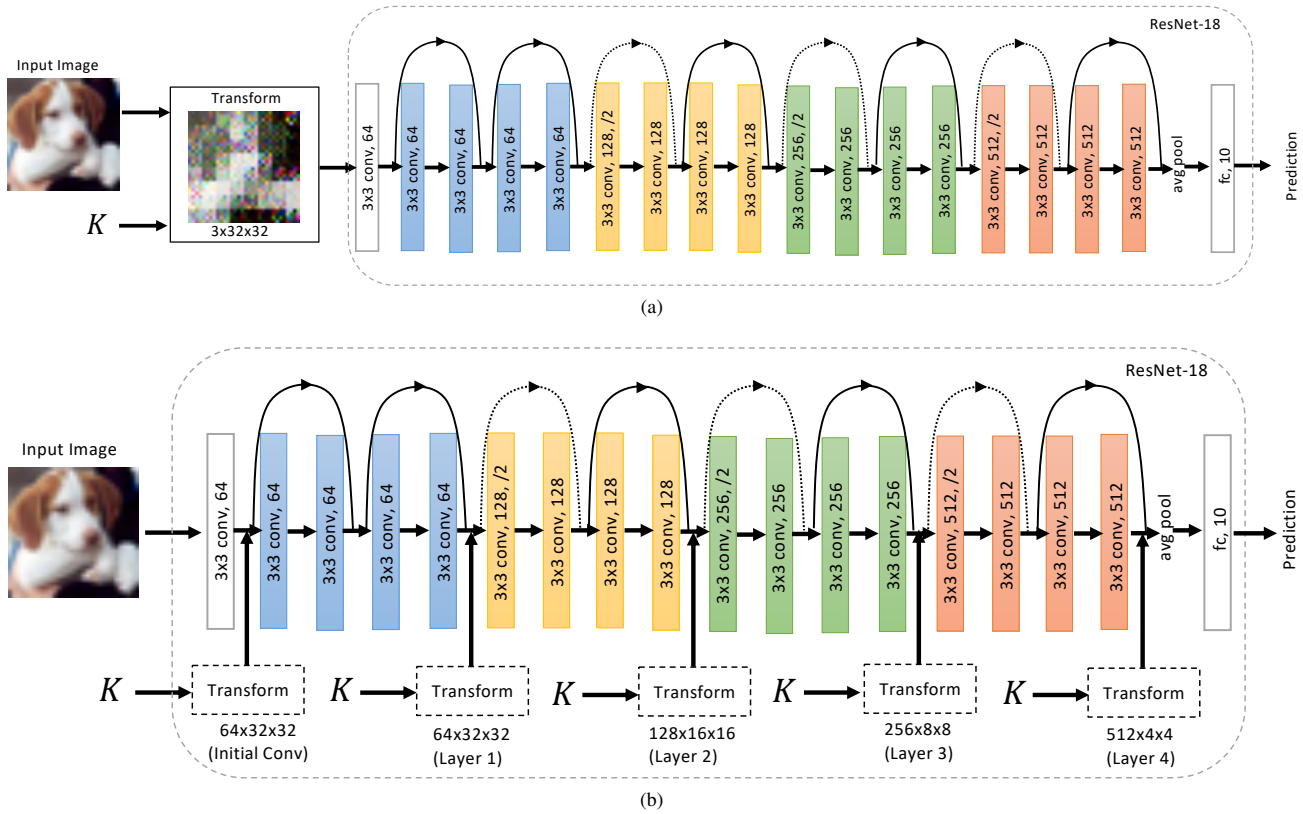

Fig. 1. Prediction pipeline of model protection methods for ResNet-18. (a) Model protection with transformed images [14]. (b) Proposed model protection with transformed feature maps.

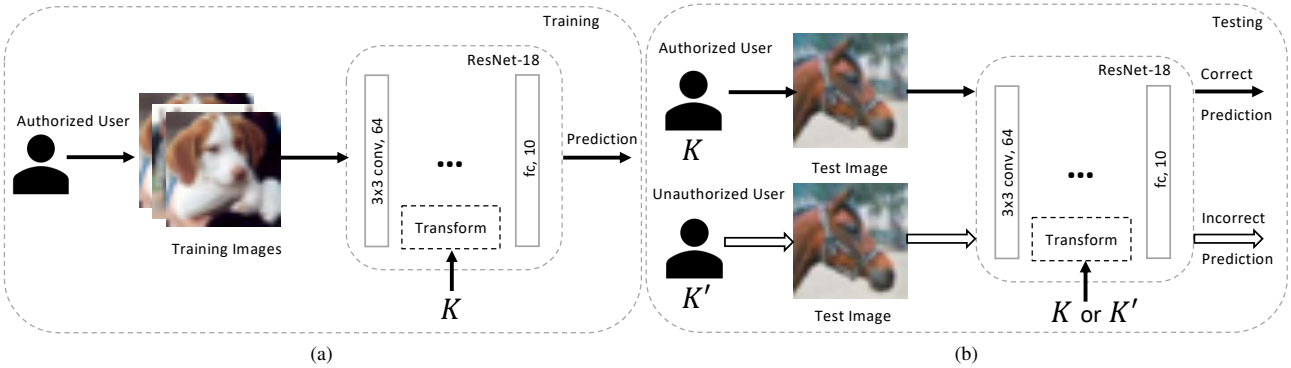

Fig. 2. Scenario of the proposed model protection. (a) Training process. (b) Testing process.

C. Key Space

In the previous method [14], key space $\mathcal{K}$ of key K depends on the number of pixels in a block, where c is fixed to 3 because there are 3 channels in an input RGB color image. In contrast, in a feature map, $c \in \{64, 128, 256, 512\}$ is the number of filters. Therefore, the key space is substantially increased and is defined as

$$\mathcal{K}(c \times M \times M) = (c \times M \times M)!, \tag{2}$$

which is larger than the key space in the previous method [14].

D. Threat Model

A threat model includes a set of assumptions such as an attacker's goals, knowledge, and capabilities. An attacker may steal a model to achieve different goals. In this paper, we consider the attacker's goal is to be able to make use of a stolen model. Therefore, we consider the attacker may estimate a key or fine-tune the model in order to remove the key. We assume the attacker obtains a clone of the model and a small subset of the training dataset. We carry out the following possible attacks with the intent of stealing a model to evaluate

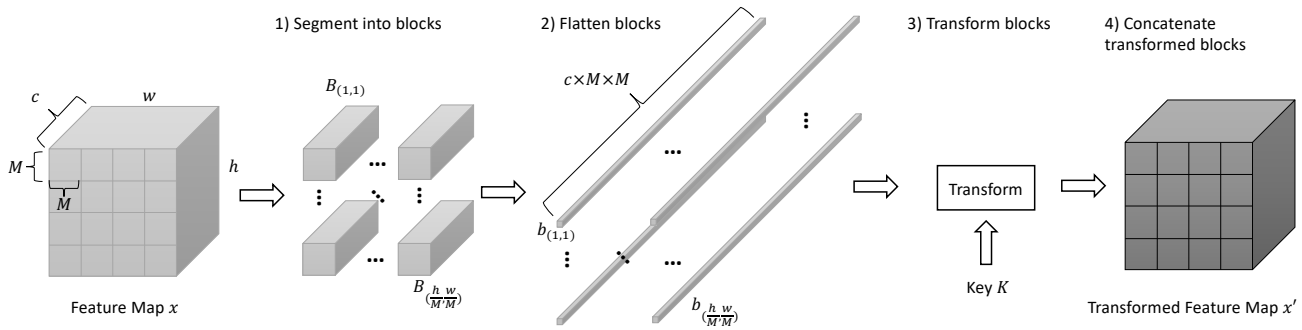

Fig. 3. Process of transforming a feature map by a block-wise transformation with secret key.

the robustness of the proposed model protection method. In experiments, the proposed method will be demonstrated to be robust against attacks.

1) Key Estimation Attack: In reality, the key estimation attack is hard to carry out for the proposed model protection because the location of the transformed feature map cannot be known from the model itself. There are many layers in a conventional CNN architecture and brute-forcing all layers to estimate the key will not be feasible. To be practical, the cost of an attack should always be lower than that of training a new model. We consider a worst-case scenario in which an attacker obtains additional information about the transformed feature map and estimate the key heuristically by observing the accuracy over their test images.

Algorithm 1 describes the process of estimating a key. First, we randomly initialize a key K'. Next, we also initialize a set of index pairs $\mathcal{P}$ as $\mathcal{P} = \{(1,2),(1,3),\ldots,(c\times M\times M-1, c\times M\times M)\}$ for a random permutation vector, v' generated by K'. The number of all possible combinations of pairs for v' can be computed as a binomial coefficient given by

$$|\mathcal{P}| = {}_nC_r = \frac{n!}{r!(n-r)!}, \tag{3}$$

where $n = c \times M \times M$, and $r = 2$. For each index pair, we swap the pair in v' if the swap improves the accuracy as shown in Algorithm 1.

Key estimation attacks do not guarantee that the attacker will find the correct key because the attacker does not know the actual performance of the correct key. However, the attacker may perform fine-tuning attacks to exploit a stolen model as below.

2) Fine-tuning Attack: In practice, CNNs are not trained from the beginning with random weights because creating a large dataset like ImageNet is difficult and expensive. Therefore, CNNs are usually pre-trained with a larger dataset (e.g., ImageNet), known as transfer learning [25], which is to train a model on top of pre-trained weights. An attacker may use fine-tuning to remove a key from a protected model. We can consider such an attack scenario where the adversary has a subset of dataset D' and fine-tunes the model.

Algorithm 1 Key Estimation

Input: Input images with labels
Output: v'

```
Initialize K' randomly.
Initialize P = {(1,2),(1,3),...,(i,j),...,(c×M×M−1, c×M×M)}
Generate v' by K'.
accuracy ← Calculate accuracy of input images
for Each index pair (i,j) in P do
    (v'_i, v'_j) ← (v'_j, v'_i)
    current_accuracy ← Calculate accuracy of input images
    if current_accuracy < accuracy then
        // Accuracy does not improve, return the swap.
        (v'_i, v'_j) ← (v'_j, v'_i)
    else
        accuracy ← current_accuracy
    end if
end for
```

IV. Experiment Results

A. Setup

We conducted image classification experiments on the CIFAR-10 dataset [26] with a batch size of 128 and live augmentation (random cropping with padding of 4 and random horizontal flip) on a training set. CIFAR-10 consists of 60,000 color images (dimension of $32\times32\times3$) with 10 classes (6000 images for each class) where 50,000 images are for training and 10,000 for testing. We used deep residual networks [27] with 18 layers (ResNet-18) and trained for 200 epochs with cyclic learning rates [28] and mixed precision training [29]. The parameters of the stochastic gradient descent (SGD) optimizer were: momentum of 0.9, weight decay of 0.0005 and maximum learning rate of 0.2.

B. Classification Performance

We trained five protected models by applying a block-wise transformation with a secret key to different feature maps in ResNet-18. All models in the experiments used a block size M of 2. We tested the protected models under three conditions

TABLE I
ACCURACY (%) AND KEY SPACE OF PROPOSED PROTECTED MODELS COMPARING WITH PREVIOUS PROTECTED ONES AND BASELINE MODEL

	Model	Key Space	Accuracy (K)	Accuracy (K')	Accuracy (without transformation)
Proposed	Initial Conv ($M = 2$)	256!	94.83	10.74	9.94
	Layer 1 ($M = 2$)	256!	95.38	9.64	10.08
	Layer 2 ($M = 2$)	512!	95.16	10.64	6.55
	Layer 3 ($M = 2$)	1024!	95.39	10.16	10.22
	Layer 4 ($M = 2$)	2048!	95.21	11.36	1.30
Previous [14]	SHF ($M = 2$)	12!	94.76	36.55	31.43
	NP ($M = 2$)	2^{12}	95.32	19.40	13.91
	FFX ($M = 2$)	2^{12}	93.80	14.67	38.84
	SHF ($M = 4$)	48!	92.58	20.15	27.77
	NP ($M = 4$)	2^{48}	93.41	12.50	12.17
	FFX ($M = 4$)	2^{48}	92.29	18.45	37.06
	Baseline		95.45 (Not protected)		

for transformation: with correct key K, with incorrect key K', and without applying the transformation.

Table I summarizes the results of proposed protected models comparing with the previous protected models [14], where the classification accuracy for incorrect key K' was averaged over 100 random keys. The key space for all models is also presented in Table I. The model trained by transforming the feature map of the initial convolution is indicated as "Initial Conv", that of the first group of residual blocks as "Layer 1", the second as "Layer 2", and so on (see also Fig. 1b). The previous protected models are named after the shorthand of the type of transformation; the model trained by using images transformed by Pixel Shuffling is denoted as SHF, that by negative/positive transformation as NP, and that by Feistel-based Format Preserving Encryption as FFX in Table I. Experiment results show that the accuracy of the proposed models are almost the same as that of non-protected accuracy (i.e., Baseline). Moreover, the proposed models significantly increased the key space, and maintained a higher classification accuracy for correct key K, a lower classification accuracy for incorrect key K' and without transformation. Therefore, the proposed models outperformed the previous protected models in any case.

C. Robustness Against Key Estimation Attack

The proposed method was evaluated against key estimation attack in accordance with Algorithm 1. As described in Section III-D1, elements in v' which are generated by K' were rearranged in accordance with the improvement in accuracy, and the resulting estimated v' was used to evaluate the performance of the protected models.

Table II captures the classification performance of the proposed protected models comparing with the previous protected ones under the key estimation attack. Note that we compared the previous models with $M = 4$ because the key space of the previous models for $M = 2$ is relatively small. We observed that the accuracy of estimated key K' for Layer 3 and 4 are $50.13\,\%$ and $89.30\,\%$ respectively. Interestingly, although the key space of Layer 3 and 4 was larger, key estimation attacks found a good key to provide a reasonable accuracy.

TABLE II
ACCURACY (%) OF PROPOSED MODELS UNDER KEY ESTIMATION ATTACK COMPARING WITH PREVIOUS PROTECTED MODELS

Model	Correct (K)	Estimated (K')
Initial Conv ($M = 2$)	94.83	20.17
Layer 1 ($M = 2$)	95.38	16.35
Layer 2 ($M = 2$)	95.16	23.58
Layer 3 ($M = 2$)	95.39	50.13
Layer 4 ($M = 2$)	95.21	89.30
SHF ($M = 4$) [14]	92.58	25.66
NP ($M = 4$) [14]	93.41	37.44
FFX ($M = 4$) [14]	92.29	80.97

However, the estimated keys were not good enough to provide a reasonable accuracy for the other models. Comparing with the previous protected models, the proposed models provided better resistance against key estimation attacks except the models: Layer 3 and 4.

D. Robustness Against Fine-tuning Attack

We ran an experiment with different sizes of the attacker's dataset (i.e., $|D'| \in \{100, 500, 1000\}$). We fine-tuned the models with D' for 30 epochs with the same training settings in Section IV-A. Table III shows the results of fine-tuning attacks for the proposed protected models comparing with the previous protected models. Although the accuracy improved with respect to the size of the adversary's dataset, it was still lower than the performance of the correct key K except for Layer 3 and 4. Comparing to the previous protected models, the model "Initial Conv" provided better robustness against fine-tuning attacks than any other models.

E. Comparison with State-of-the-art DNN Access Control Method

Since underlying mechanisms of the DNN access control method by [12] which uses a perturbation network and the proposed model protection method are different, it is difficult to directly compare them. To make a high-level comparison, we implemented the anti-piracy method [12] in the same training settings as in Section IV-A for the CIFAR-10 dataset. Then, we compared the anti-piracy method [12] with the

TABLE III
ACCURACY (%) OF PROPOSED MODELS UNDER FINE-TUNING ATTACKS COMPARING WITH PREVIOUS PROTECTED MODELS. ALL MODELS ARE WITH $M = 2$.

Model	Original	$\|D'\| = 100$	$\|D'\| = 500$	$\|D'\| = 1000$
Initial†	94.83	18.47	55.38	69.42
Layer 1	95.38	22.37	66.90	78.54
Layer 2	95.16	21.75	66.38	74.94
Layer 3	95.39	20.84	74.59	80.99
Layer 4	95.21	87.43	73.28	94.63
SHF [14]	94.76	33.89	70.69	78.01
NP [14]	95.32	16.84	58.17	75.00
FFX [14]	93.80	44.32	75.45	80.86

†Initial stands for "Initial Conv".

TABLE IV
COMPARISON OF PROPOSED PROTECTED MODEL AND STATE-OF-THE-ART ANTI-PIRACY MODEL [12]

Model	Authorized Accuracy	Unauthorized Accuracy	Method
Initial Conv (Proposed)	94.83	9.94	Block-Wise Transformation
Anti-piracy [12]	92.89	14.21	Perturbation Network
Baseline	95.45	95.45	Non-protected

proposed model protection method ("Initial Conv" model) in terms of authorized accuracy (i.e., with correct transformation/perturbation), unauthorized accuracy (i.e., without transformation/perturbation), and the core method used in the two mechanisms (Table IV). The main difference is that the proposed model protection method uses a block-wise transformation with a secret key and the anti-piracy method [12] utilizes a perturbation network. In terms of classification performance, the proposed method achieves a higher authorized accuracy, which is close to baseline accuracy, and a lower unauthorized accuracy than that of the anti-piracy method [12].

V. CONCLUSION

We proposed a model protection method by directly applying a block-wise transformation with a secret key to feature maps in the network. As a result, the proposed model protection method not only improves the classification accuracy, but also increases the key space substantially. The performance accuracy of the proposed protected model was closer to that of a non-protected model when the key was correct, and it dropped drastically when an incorrect key was given, suggesting that the proposed model is not usable to its full capacity for unauthorized users. Experiments results show that the proposed models outperformed the previous protected models in terms of classification accuracy and robustness against key estimation attacks and fine-tuning attacks.

ACKNOWLEDGMENT

This study was partially supported by JSPS KAKENHI (Grant Number JP21H01327) and JST CREST (Grant Number JPMJCR20D3).

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[2] Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh, "Embedding watermarks into deep neural networks," in *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, 2017, pp. 269–277.

[3] Y. Adi, C. Baum, M. Cissé, B. Pinkas, and J. Keshet, "Turning your weakness into a strength: Watermarking deep neural networks by backdooring," in *27th USENIX Security Symposium*, 2018, pp. 1615–1631.

[4] J. Zhang, Z. Gu, J. Jang, H. Wu, M. P. Stoecklin, H. Huang, and I. 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] B. D. Rouhani, H. Chen, and F. Koushanfar, "Deepsigns: A generic watermarking framework for IP protection of deep learning models," *arXiv:1804.00750*, 2018. [Online]. Available: http://arxiv.org/abs/1804.00750

[6] L. Fan, K. Ng, and C. S. Chan, "Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks," in *Advances in Neural Information Processing Systems*, 2019, pp. 4716–4725.

[7] S. Sakazawa, E. Myodo, K. Tasaka, and H. Yanagihara, "Visual decoding of hidden watermark in trained deep neural network," in *2nd IEEE Conference on Multimedia Information Processing and Retrieval*, 2019, pp. 371–374.

[8] E. L. Merrer, P. Pérez, and G. Trédan, "Adversarial frontier stitching for remote neural network watermarking," *Neural Computing and Applications*, vol. 32, no. 13, pp. 9233–9244, 2020.

[9] M. AprilPyone and H. Kiya, "Piracy-resistant DNN watermarking by block-wise image transformation with secret key," in *Proceedings of the 2021 ACM Workshop on Information Hiding and Multimedia Security*, 2021, pp. 159–164.

[10] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015, pp. 1322–1333.

[11] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *International Conference on Learning Representations*, 2014.

[12] M. Chen and M. Wu, "Protect your deep neural networks from piracy," in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2018, pp. 1–7.

[13] M. AprilPyone and H. Kiya, "Training dnn model with secret key for model protection," in *2020 IEEE 9th Global Conference on Consumer Electronics (GCCE)*, 2020, pp. 818–821.

[14] ——, "A protection method of trained CNN model with a secret key from unauthorized access," *APSIPA Transactions on Signal and Information Processing*, vol. 10, p. e10, 2021.

[15] ——, "Block-wise image transformation with secret key for adversarially robust defense," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 2709–2723, 2021.

[16] A. Kawamura, Y. Kinoshita, T. Nakachi, S. Shiota, and H. Kiya, "A privacy-preserving machine learning scheme using etc images," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 103, no. 12, pp. 1571–1578, 2020.

[17] M. Tanaka, "Learnable image encryption," in *2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, 2018, pp. 1–2.

[18] W. Sirichotedumrong, Y. Kinoshita, and H. Kiya, "Pixel-based image encryption without key management for privacy-preserving deep neural networks," *IEEE Access*, vol. 7, pp. 177 844–177 855, 2019.

[19] T. Chuman, W. Sirichotedumrong, and H. Kiya, "Encryption-then-compression systems using grayscale-based image encryption for jpeg images," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 6, pp. 1515–1525, June 2019.

[20] W. Sirichotedumrong, T. Maekawa, Y. Kinoshita, and H. Kiya, "Privacy-preserving deep neural networks with pixel-based image encryption considering data augmentation in the encrypted domain," in *2019 IEEE International Conference on Image Processing (ICIP)*, 2019, pp. 674–678.

[21] W. Sirichotedumrong and H. Kiya, "Grayscale-based block scrambling image encryption using ycbcr color space for encryption-then-compression systems," *APSIPA Transactions on Signal and Information Processing*, vol. 8, 2019.
[22] K. Kurihara, S. Imaizumi, S. Shiota, and H. Kiya, "An encryption-then-compression system for lossless image compression standards," *IEICE transactions on information and systems*, vol. 100, no. 1, pp. 52–56, 2017.
[23] T. Chuman, K. Kurihara, and H. Kiya, "On the security of block scrambling-based EtC systems against extended jigsaw puzzle solver attacks," *IEICE Transactions on Information and Systems*, vol. 101, no. 1, pp. 37–44, 2018.
[24] H. Chen, B. D. Rouhani, and F. Koushanfar, "Deepmarks: A digital fingerprinting framework for deep neural networks," *arXiv:1804.03648*, 2018. [Online]. Available: http://arxiv.org/abs/1804.03648
[25] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.
[26] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," University of Toronto, Tech. Rep., 2009.
[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[28] L. N. Smith and N. Topin, "Super-convergence: Very fast training of residual networks using large learning rates," *arXiv:1708.07120*, 2017. [Online]. Available: http://arxiv.org/abs/1708.07120
[29] P. Micikevicius, S. Narang, J. Alben, G. F. Diamos, E. Elsen, D. García, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh, and H. Wu, "Mixed precision training," *arXiv:1710.03740*, 2017. [Online]. Available: http://arxiv.org/abs/1710.03740