DeepMarks: A Secure Fingerprinting Framework for Digital Rights Management of Deep Learning Models

Huili Chen
huc044@ucsd.edu
University of California San Diego

Bita Darvish Rouhani
bita@ucsd.edu
University of California, San Diego

Cheng Fu
cfu@ucsd.edu
University of California San Diego

Jishen Zhao
University of California, San Diego
jzhao@ucsd.edu

Farinaz Koushanfar
farinaz@ucsd.edu
University of California San Diego

ABSTRACT

Deep Neural Networks (DNNs) are revolutionizing various critical fields by providing an unprecedented leap in terms of accuracy and functionality. Due to the costly training procedure, high-performance DNNs are typically considered as the Intellectual Property (IP) of the model builder and need to be protected. While DNNs are increasingly commercialized, the pre-trained models might be illegally copied or redistributed after they are delivered to malicious users. In this paper, we introduce DeepMarks, the first end-to-end collusion-secure fingerprinting framework that enables the owner to retrieve model authorship information and identification of unique users in the context of deep learning (DL). DeepMarks consists of two main modules: (i) Designing unique fingerprints using anti-collusion codebooks for individual users; and (ii) Encoding each constructed fingerprint (FP) in the probability density function (pdf) of the weights by incorporating an FP-specific regularization loss during DNN re-training. We investigate the performance of DeepMarks on various datasets and DNN architectures. Experimental results show that the embedded FP preserves the accuracy of the host DNN and is robust against different model modifications that might be conducted by the malicious user. Furthermore, our framework is scalable and yields perfect detection rates and no false alarms when identifying the participants of FP collusion attacks under theoretical guarantee. The runtime overhead of retrieving the embedded FP from the marked DNN can be as low as 0.056%.

CCS CONCEPTS

• **Information systems → Information retrieval**; **Multimedia information systems**; • **Security and privacy → Digital rights management**.

KEYWORDS

Deep Neural Networks, Intellectual Property Protection, Digital Right Management, Digital Fingerprinting

ACM Reference Format:
Huili Chen, Bita Darvish Rouhani, Cheng Fu, Jishen Zhao, and Farinaz Koushanfar. 2019. DeepMarks: A Secure Fingerprinting Framework for Digital Rights Management of Deep Learning Models. In *International Conference on Multimedia Retrieval (ICMR '19), June 10–13, 2019, Ottawa, ON, Canada.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3323873.3325042

1 INTRODUCTION

The recent advances in Deep Learning (DL) have provided a paradigm shift in various domains including autonomous transportation, nuclear engineering and smart health [4, 11, 17]. Training a highly accurate DNN is a costly process since it requires: (i) Processing massive amounts of data acquired for the target application; (ii) Allocating substantial computing resources to fine-tune the topology and hyper-parameters of the deployed model. Given the costly process of designing/training, the pre-trained DNNs are considered as the Intellectual Property (IP) of the model owner and needs to be protected. As an increasing amount of pre-trained DNNs are open-sourced / distributed on the Internet [2], IP protection and Digital Right Management (DRM) of these public available models are particularly important to preserve the competitiveness of the model owner and facilitate reliable technology transfer.

Digital watermarks and fingerprints have been immensely leveraged to protect the authorship of multimedia content and functional artifacts [10, 14, 15]. However, the extension of watermarking and fingerprinting techniques to the DL domain for reliable model distribution is still in its infancy. Developing a practical DNN fingerprinting technique is challenging since: (C1) Fingerprint (FP) embedding shall not incur performance degradation of the original model; (C2) FP detection scheme shall yield a minimal false alarm rate to avoid incorrectly accusing innocent customers for misusing/stealing the model; (C3) The embedded FP shall be sustainable to withstand potential model modification and FP deconstruction attacks conducted by malicious users. In this paper, we investigate how to tackle these challenges and present DeepMarks as the first promising solution for large-scale model distribution systems.

A holistic IP protection technique is expected to provide the following two capabilities: **(i) DNN ownership proof**, the model owner shall be able to prove the authorship of her model after the DNN is distributed to the users; **(ii) Tracking/Identifying unique users**, the model owner can trace different customers who are using the same IP and determine which person has misused the model if IP infringement is discovered. These two properties are the requirements of IP authorship protection and DRM, respectively.

DNNs can be leveraged in either a white-box setting (model internals are publicly known) or a black-box setting (only model outputs are known). DeepMarks targets at secure and robust DNN fingerprinting in the white-box scenario, which is a common practice considering the prevalence of DL models on the Internet.

Prior works have proposed DNN watermarking methodologies for model authentication in the white-box [13, 16, 21] and the black-box setting [1, 12, 25]. However, all existing watermarking methods only address the first requirement of DNN IP protection (ownership proof) while ignoring the second one (tracking unique users). We demonstrate that the state-of-art DNN watermarking scheme [21] is deficient to provide a robust fingerprinting solution. More specifically, we show that multiple users can collaborate and construct an unmarked model that achieves a comparable accuracy as the baseline using their individually watermarked models (referred to as *'FP collusion attack'*), thus defeating the DNN watermarking approach in [21]. DeepMarks it motivated to overcome this limitation.

This paper proposes DeepMarks, the first *provably secure* DNN fingerprinting framework that empowers coherent integration of robust digital markers into DL models. DeepMarks takes the pre-trained DNN (owner's IP) together with a set of security parameters as its inputs. Multiple functionality-preserved variants of the original model are returned as the outputs, carrying unique fingerprints of individual users in the model distribution system. We address the challenges (C1-C3) by designing *collusion-aware* fingerprints and encoding the FP information in weights using a customized regularization loss during DNN re-training. The intuition behind is that there are abundant redundancies existing in the pre-trained DNN due to its high dimensionality. DeepMarks leverages such redundancies and tackles DNN fingerprinting as an auxiliary task of the original data application. As such, the designed fingerprint is integrated as an inseparable part of the weight parameters, ensuring that the adversary cannot remove the FP without compromising the performance of the marked model.

DeepMarks framework is innovative in a sense that it is the first collusion-secure fingerprinting framework with theoretical guaranteed on detection performance. Unlike prior works that only focus on addressing model ownership authentication for a single user, we consider a large-scale model distribution system where multiple users might perform collaborative FP deconstruction attacks. Such a scenario is more practical considering the real-world setting. Furthermore, DeepMarks provides model authorship verification and DRM simultaneously, thus is the first full-fledged IP protection solution in the domain of deep learning. Our approach is generic and compatible with various applications as well as DNN architectures.

This paper makes the following contributions:

- **Enabling effective IP protection and DRM for DNNs in a model distribution system.** We propose DeepMarks, a novel fingerprinting methodology that encodes robust fingerprints in the probability density function (pdf) of weights for model ownership proof and unique users tracing. DeepMarks is provably more robust against FP collusion attacks compared to the state-of-the art DNN watermarking scheme.
- **Characterizing the requirements for an effective fingerprinting methodology in the deep learning domain.** We introduce a comprehensive set of metrics to assess the performance of a DNN fingerprinting methodology. Such metrics provide new perspectives for model designers and facilitate coherent comparison of current and pending DNN IP protection techniques.
- **Investigating the performance of DeepMarks on various DNN benchmarks.** We perform extensive proof-of-concept experiments to corroborate the efficacy and robustness of DeepMarks. Empirical results show that our framework yields perfect FP detection rates and no false alarms given the properly selected security parameters.

We emphasize that enabling the model owner to *retrieve information about model authorship and potential illegal usage* is important due to the prevalence of DNNs in critical fields. DeepMarks is motivated to address the pressing concerns about the IP and digital right management of valuable DL models. This paper opens a new axis for the growing research in secure deep learning and sheds light on the unexplored limitations of DNN watermarking techniques.

2 RELATED WORK

2.1 Conventional Watermarking and Fingerprinting

Digital watermarks (WMs) and fingerprints are identifiers invisibly embedded as an integral part of the host design for IP protection. The host of the identifier can be images, video contents, and functional artifacts such as digital integrated circuits ([5, 7, 14]). The main difference between watermarking and fingerprinting is that the WM remains the same for all copies of the IP while the FP is unique for each copy. As such, FPs address the ambiguity of WMs and enables tracking of IP misuse conducted by a specific user.

2.2 DNN Watermarking

IP protection of valuable DNN models is a subject of increasing interest to researchers and practitioners. Uchida *et al.* take the first step towards DNN watermarking and embeds the WM by adding constraints to the weight parameters. The WM is later extracted from the marked layer assuming a white-box scenario [21]. To alleviate the constraint that the parameters of the queried model are available during WM extraction, several papers propose zero-bit watermarking techniques that are applicable in the black-box scenario [1, 12, 25]. These works suggest different methods to generate watermark images and labels as the 'trigger set', which is then used to tweak the decision boundary of the pre-trained model to carry the WM. The existence of the WM is determined by querying the remote model with the WM images and thresholding the accuracy on the trigger set. All of the above papers consider a single-user setting, unaware of potential collusion attacks in multi-user scenarios. In this paper, we present a collusion-secure DNN fingerprinting framework to address the limitations of DNN watermarking, thus providing a holistic IP protection solution.

3 PROBLEM FORMULATION

While cloud-based DNN services are widely adopted in various applications, white-box DL model deployment provides a more powerful utilization alternative that encourages research communities and industrial developers to improve existing DL techniques. DeepMarks is motivated to protect the IP of white-box DNNs in a model sharing system. To the best of our knowledge, there is no prior work on DNN fingerprinting. In this paper, we define fingerprinting as the task of designing a v-bit binary code-vector

Table 1: Requirements for an effective fingerprinting methodology of deep neural networks.

Requirements	Description
Fidelity	The accuracy of the target neural network shall not be degraded as a result of fingerprint embedding.
Uniqueness	The fingerprint need to be unique for each user to achieve unambiguous identification.
Efficiency	The overhead of fingerprint embedding and extraction shall be negligible.
Security	Fingerprint embedding shall leave no tangible footprint in the host neural network; thus, an unauthorized individual cannot detect the presence of a fingerprint in the model.
Robustness	The fingerprint shall be robust against potential fingerprint destruction and model modification attacks.
Reliability	Fingerprint extraction shall yield minimal false negatives to ensure high detection rates.
Integrity	The fingerprinting methodology should yield minimal false alarm (a.k.a., false positive). This means that the probability of an innocent user being accused as a colluder should be very low.
Scalability	The fingerprinting methodology should be able to support numerous users in the distributed system.
Generality	The fingerprinting technique should be applicable to various datasets and network architectures.

$\mathbf{c_j} \in \{0,1\}^v$ for each user and embedding it in the parameters (e.g., weights) of one layer or multiple layers in the host neural network. Here, $j = 1, ..., n$ is the index for each distributed user and n is the total number of users. The objective of fingerprinting is *two-fold*: (i) Claiming the ownership of a specific DNN, and (ii) Tracing the unintended usage of the model conducted by distributed users.

3.1 Requirements

Table 1 summarizes the requirements for an effective fingerprint technique in the DL domain. In addition to fidelity, efficiency, security, reliability, integrity, and robustness requirements that are shared between fingerprinting and watermarking, a successful fingerprinting methodology should also satisfy uniqueness, scalability, and collusion resilience criteria. Uniqueness is the intrinsic property of fingerprints that enable unambiguous user identification. Scalability is a key factor to support model ownership authentication and DRM in large-scale systems. Collusion resistance is a desired property considering the practicality of collusion attacks.

3.2 Threat Model

Corresponding to the robustness requirement in Table 1, we discuss four types of DL domain-specific attacks that the fingerprinting methodology should be resistant to: model fine-tuning, parameter pruning, fingerprint collusion, and fingerprint overwriting attacks.
Parameter Pruning. Genuine users may leverage parameter pruning to reduce the memory and computation overhead of the DNN [6] while adversaries may apply pruning to remove the FP. As such, an effective fingerprinting technique shall be resistant to parameter pruning that incurs the change of parameters.
Model Fine-tuning. Fine-tuning might be performed by honest users for transfer learning, or by malicious attackers to remove the FP. Since parameters carrying the FP are altered during fine-tuning, the embedded FP should be robust against this modification.
Fingerprint Collusion Attack. A group of users who have the same host neural network with different embedded fingerprints may perform collusion attacks to construct a functional model where no fingerprints can be detected by the owner. In this paper, we focus on evaluating DeepMarks' robustness against the FP averaging attack.
Fingerprint Overwriting. Assuming an active adversary knows the deployed fingerprinting methodology, he may embed a new FP to destroy the original one inserted by the authentic model owner. While the location where the original FP is embedded shall be a secret to the malicious parties, it is conceivable that the attacker can embed the new FP into multiple layers of the target DNN to increase the success rate of destroying the original FP.

4 DEEPMARKS FRAMEWORK

Figure 1 demonstrates the global flow of DeepMarks framework. DeepMarks consists of three main modules: FP embedding, user identification, and colluder detection. The fingerprinted model is assumed to be deployed in a white-box setting where the model internals are transparent to the public. Such an assumption is practical considering the popularity of model sharing/distribution in the real-world setting. There are two types of FP modulation schemes in the multimedia domain: orthogonal modulation [9], and coded modulation [22]. Since coded fingerprinting possesses better collusion resilience and can be considered as a general case of orthogonal fingerprinting [18, 19], we focus on coded FPs using DeepMarks framework throughout the paper. Note that DeepMarks is orthogonal to the existing code modulation schemes and can be further augmented when advanced modulation methods are integrated. The workflows of FP embedding and detection are detailed in Section 4.1, 4.2 respectively. The computation and communication overhead of DeepMarks framework is analyzed in Section 4.3.

4.1 Fingerprint Embedding

DeepMarks' FP embedding via regularization is inspired by *constraint-based watermarking system* in the multi-media domain [8]. More specifically, the original problem (e.g., image classification) is used as the cover constraint and FP embedding is incorporated as the additional stego constraint. DeepMarks leverages the over-parameterization of high dimensional DNNs to enforce the stego constraint. As such, DeepMarks helps to alleviate model over-fitting and preserve the performance of the original model. Embedding FPs in the training-from-scratch fashion is impractical for large-scale distributed systems since the fingerprinting process is required for each copy of the target DNN. DeepMarks framework tackles this viability concern by treating FP embedding as a *post-processing* step implemented via fine-tuning the pre-trained model with the FP-specific regularization loss. Particularly, FP embedding is formulated as an *off-line, one-time* process performed locally by the owner before model distribution. We demonstrate the construction and embedding of coded FPs based on DeepMarks framework as follows.
(I) Fingerprint Construction. DeepMarks ensures provable collusion resilience by taking advantage of the anti-collusion code (ACC) theory when constructing the codebook. ACC is proposed in [22] for collusion-resistant coded fingerprinting and has the following property: the composition of any subset of K or fewer code-vectors is unique. This property allows the owner to identify a group of K or fewer colluders from the composition precisely. A K-resilient

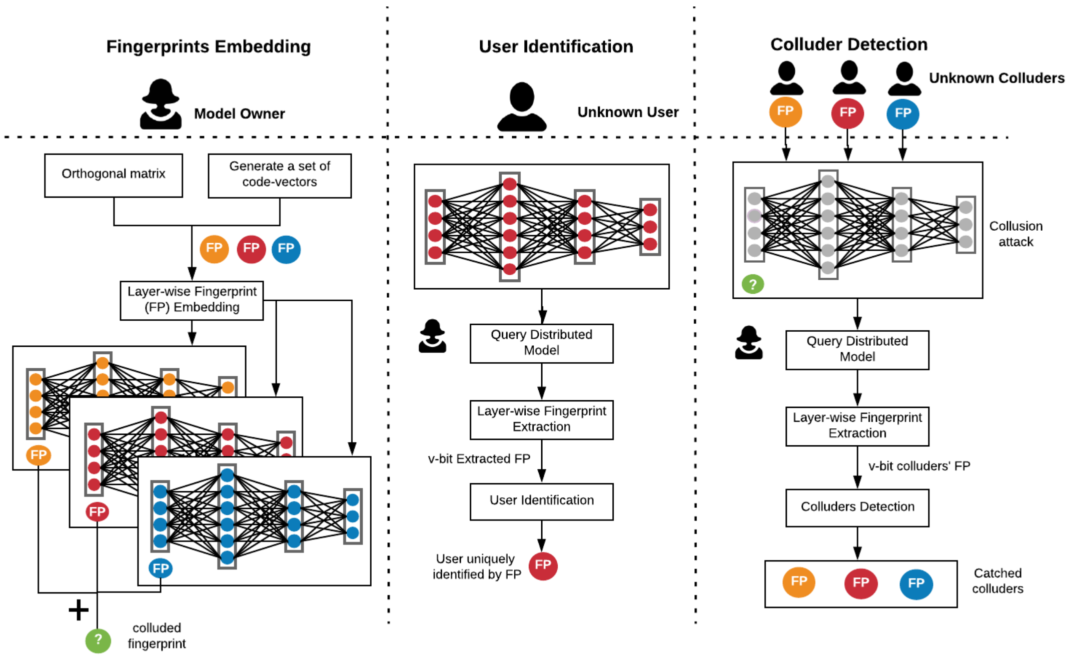

Figure 1: DeepMarks Global Flow: DeepMarks performs DNN fingerprinting by embedding the designated fingerprint information in the probability distribution of weights at selected layers. To enable model ownership authentication and and digital right management, DeepMarks enables the model owner to retrieve embedded fingerprints for user identification as well as colluder detection after distributing the models.

AND-ACC codebook is a matrix where the element-wise composition is logic-AND and allows for the accurate identification of K unique colluders from their composition.

To generate ACC of binary values, DeepMarks deploys Balanced Incomplete Block Design (BIBD) [23]. A (v, k, λ)-BIBD has $b = \lambda(v^2 - v)/(k^2 - k)$ blocks with block size k. The BIBD can be represented by its corresponding incidence matrix $\mathbf{C}_{v \times b}$ where each element:

$$c_{ij} = \begin{cases} 1, & \text{if } i^{th} \text{ value occurs in } j^{th} \text{ block} \\ 0, & \text{otherwise.} \end{cases}$$

By setting the number of concurrent occurrence to one ($\lambda = 1$) and assigning the bit complement of columns of the incidence matrix $\mathbf{C}_{v \times b}$ as the code-vectors, the resulting $(v, k, 1)$-BIBD code is $(k-1)$-resilient and supports up to $n = b$ users [19]. Note that DeepMarks is generic and compatible with other anti-collusion code design schemes. Here, we focus on illustrating the feasibility of DeepMarks and leave advanced codebook construction to future work.

DeepMarks generates coded FPs as follows. Given the designed incidence matrix $\mathbf{C}_{v \times b}$, the coefficient matrix $\mathbf{B}_{v \times b}$ for FPs is computed from the linear mapping $b_{ij} = 2c_{ij} - 1$. The FP of the j^{th} user is then generated from an orthogonal matrix $\mathbf{U}_{v \times v}$ and the coefficient matrix $\mathbf{B}_{v \times b}$ as:

$$\mathbf{f_j} = \sum_{i=1}^{v} b_{ij} \mathbf{u_i}, \tag{1}$$

where $\mathbf{b_j} \in \{\pm 1\}^v$ is the coefficient of user j. $\mathbf{U}$ is generated from element-wise Gaussian distribution for security consideration [22].

(II) Fingerprint Insertion. The FP obtained from Equation (1) is embedded in the selected layers of the pre-trained model by incorporating an FP-specific embedding loss term to the conventional loss function ($\mathcal{L}_0$):

$$\mathcal{L} = \mathcal{L}_0 + \gamma\, MSE(\mathbf{f_j} - \mathbf{X}\mathbf{w}). \tag{2}$$

Here, MSE is the mean square error function, γ is the embedding strength that controls the contribution of FP embedding loss, $\mathbf{X}$ is the owner's secret projection matrix generated from standard normal distribution $\mathcal{N}(0, 1)$. The vector $\mathbf{w}$ is the flattened averaged weights of the target layers that carry the FP information.

As a proof-of-concept analysis, we embed the FP $\mathbf{f_j}$ in a convolutional layer of the host DNN. The weight is a 4D tensor $\mathbf{W} \in \mathbb{R}^{D \times D \times F \times H}$ where D is the kernel size, F and H is the number of input and output channels, respectively. We average the weight $\mathbf{W}$ over the output channel dimension and stretch the result to a vector $\mathbf{w}$. The FP is embedded in the vector $\mathbf{w} \in \mathbb{R}^N$ where $N = D \times D \times F$ is the embedding dimension. The additive FP embedding loss $\mathcal{L}_{FP} = MSE(\mathbf{f_j} - \mathbf{X}\mathbf{w})$ is minimized together with the conventional loss during DNN training to encode the FP $\mathbf{f_j}$ in the pdf of weights in the selected layer. We assume that the three matrices $\mathbf{U}$, $\mathbf{C}$, $\mathbf{X}$, and the layers selected for FP embedding are secret security parameters that are only known to the owner. The main difference between DeepMarks's FP embedding and *transfer learning* is that the latter one involves training with a new dataset.

4.2 Fingerprint Extraction

4.2.1 User Identification. DeepMarks uniquely identifies each individual user by recovering her associated code-vector assuming the availability of model parameters. To do so, DeepMarks undergoes four main steps: **(i)** Acquiring the weights in the marked layers to reconstruct the FP vector $\widetilde{\mathbf{f_j}} = \mathbf{X}\widetilde{\mathbf{w}}_\mathbf{j}$; **(ii)** Recovering the correlation score vector from the FP vector and the owner's secret basis matrix by computing $\widetilde{\mathbf{b_j}} = \widetilde{\mathbf{f_j}}^T \mathbf{U}$; **(iii)** Decoding the ACC code-vector $\widetilde{\mathbf{c_j}}$ from the element-wise hard-thresholding of $\widetilde{\mathbf{b_j}}$; **(iv)** Comparing the recovered code-vector $\widetilde{\mathbf{c_j}}$ with each column in the owner's codebook

C where the matching position uniquely identifies the user. We use Bit Error Rate (BER) computed between the true code-vector and the recovered one to assess DeepMarks's performance of FP extraction. The user identification process is considered successful if there exists one unique matching (BER=0 for a column in **C**).

To illustrate DeepMarks's workflow for user identification, let us consider a $(7, 3, 1)$-BIBD codebook shown in Equation (3). The FPs for 7 users shown in Equation (4) are constructed using the columns of the incidence matrix (codebook) **C** and the basis matrix **U** as described before.

$$\mathbf{C} = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}, \tag{3}$$

$$\begin{cases} \mathbf{f_1} = -\mathbf{u_1} - \mathbf{u_2} + \mathbf{u_3} - \mathbf{u_4} + \mathbf{u_5} + \mathbf{u_6} + \mathbf{u_7}, \\ \cdots \\ \mathbf{f_6} = +\mathbf{u_1} + \mathbf{u_2} - \mathbf{u_3} - \mathbf{u_4} + \mathbf{u_5} + \mathbf{u_6} - \mathbf{u_7}, \\ \mathbf{f_7} = +\mathbf{u_1} + \mathbf{u_2} + \mathbf{u_3} - \mathbf{u_4} - \mathbf{u_5} - \mathbf{u_6} + \mathbf{u_7}, \end{cases} \tag{4}$$

Note that for user 1, her coefficient vector can be recovered by computing the correlation scores:

$$\widetilde{\mathbf{b_1}} = \mathbf{f_1}^T[\mathbf{u_1}, ..., \mathbf{u_7}] = [-1, -1, +1, -1, +1, +1, +1].$$

The corresponding code-vector is then extracted by the inverse linear mapping $c_{ij} = \frac{1}{2}(b_{ij} + 1)$, resulting in $\widetilde{\mathbf{c_1}} = [0, 0, 1, 0, 1, 1, 1]$. The recovered $\widetilde{\mathbf{c_1}}$ is exactly the same as the first column of **C**, indicating the effectiveness of DeepMarks framework for user identification.

4.2.2 *Colluder Detection.*

In this section, we describe how DeepMarks deploys the intrinsic asset of AND-ACC for colluders detection. We focus on *FP averaging attack*, which is a typical and cost-effective FP collusion attack, in our evaluation. Furthermore, we consider the worst-case scenario where the colluders know the positions of the embedded layers. As a result, the colluders can perform element-wise average on their weights and produce $\widetilde{\mathbf{w}}_{\mathbf{avg}}$ to answer the owner's inquiry. The owner then computes the colluded correlation vector $\widetilde{\mathbf{b}}_{\mathbf{avg}}$ as follows:

$$\widetilde{\mathbf{f}}_{\mathbf{avg}} = \mathbf{X}\widetilde{\mathbf{w}}_{\mathbf{avg}}, \tag{5}$$

$$\widetilde{\mathbf{b}}_{\mathbf{avg}} = (\widetilde{\mathbf{f}}_{\mathbf{avg}})^{\mathbf{T}}\mathbf{U}. \tag{6}$$

DeepMarks leverages hard-thresholding detectors for colluder identification. The ACC code-vector is decoded from the correlation vector $\widetilde{\mathbf{b}}_{\mathbf{avg}} = [\widetilde{b}^1_{avg}, ..., \widetilde{b}^v_{avg}]$ by comparing each element with an owner-defined threshold τ:

$$\widetilde{c}^i_{avg} = \begin{cases} 1, \text{if } \widetilde{b}^i_{avg} > \tau, \\ 0, \text{otherwise.} \end{cases} \tag{7}$$

Given the AND-ACC code-vector of the colluders $\widetilde{\mathbf{c}}_{\mathbf{avg}}$, the remaining problem is to find the subsets of columns from the codebook **C** such that their logic-AND composition is equal to $\widetilde{\mathbf{c}}_{\mathbf{avg}}$. For a $(v, k, 1)$-BIBD-ACC, at most $(k - 1)$ colluders can be uniquely identified [19] with theoretical guarantee.

As an example, we demonstrate DeepMarks's colluder detection scheme using the codebook given in Equation (3). Assuming user 6 and user 7 collaboratively generate the averaged fingerprint:

$$\mathbf{f_{avg}} = \frac{1}{2}(\mathbf{f_6} + \mathbf{f_7}) = \frac{1}{2}(2\mathbf{u_1} + 2\mathbf{u_2} - 2\mathbf{u_4}).$$

where individual FPs are defined in Equation (4). The owner computes the colluders' correlation vector:

$$\mathbf{b_{avg}} = (\mathbf{f_{avg}})^{\mathbf{T}}\mathbf{U} = [1, 1, 0, -1, 0, 0, 0].$$

The corresponding code-vector is then extracted according to decision rule in Equation (7), resulting in $\mathbf{c_{avg}} = [1, 1, 0, 0, 0, 0, 0]$ One can observe that the logic-AND composition of column 6 and column 7 in the codebook **C** is exactly equal to $\mathbf{c_{avg}}$, while all the other compositions (of two or more columns) do not satisfy the constraint. This example shows that DeepMarks correctly identifies all participants of the collusion attack without any false alarms.

4.3 Computation Overhead Analysis

We discuss the overhead of a DNN fingerprinting technique from two perspectives: FP embedding, and FP extraction. Since FP embedding locally performed locally by the owner before model distribution, there is no communication overhead involved. The computation overhead is determined by the additional operations to compute the FP-specific loss $\mathcal{L}_{FP} = MSE(\mathbf{f}_j - \mathbf{Xw})$ during DNN training, which has complexity $O(vN + v)$. To extract the FP, the queried user sends the vector $\widetilde{\mathbf{w}}_{N\times1}$ of the marked layer to the owner, thus the communication overhead is $O(N)$. The computation overhead is incurred by two matrix multiplications: $\widetilde{\mathbf{f}} = \mathbf{X}_{v\times N} \cdot \widetilde{\mathbf{w}}_{N\times1}$, $\widetilde{\mathbf{b}} = \widetilde{\mathbf{f}}^T_{1\times v} \cdot \mathbf{U}_{v\times v}$ with complexity $O(vN)$ and $O(v^2)$, respectively. We provide the quantitative runtime overhead results in Section 5.1.

5 EXPERIMENTS

In this section, we present the evaluation of DeepMarks on image classification tasks using two popular types of DL models: Convolutional Neural Networks (CNNs) and Wide Residual Networks (WRNs). The benchmark datasets and topologies are summarized in Table 2. Note that DeepMarks framework is based on strategical regularization during DNN training, thus is generic and applicable to other network architectures such as multilayer perceptrons (MLP) and recurrent neural networks (RNN). We want to emphasize that DeepMarks does *not require prior knowledge about the number of colluders* (k) in the detection stage. All participants of the collusion attack are automatically identified by DeepMarks as discussed in Section 4.2.2, rendering the detection scheme useful in practice.

Experimental Setup. To evaluate the performance of DeepMarks coded fingerprinting scheme, we use a $(31, 6, 1)$-BIBD AND-ACC codebook that accommodates 31 users. We select the embedding strength γ in Equation (2) such that the embedding loss satisfies $\mathcal{L}_{FP} = 0.1 \cdot \mathcal{L}_0$ in the beginning of FP embedding. The total FP-regularized loss of the model is minimized during regular backpropagation. DeepMarks employs the BER computed between the code-vector recovered from the current weights and the ground-truth value as a 'monitor' to terminate FP embedding when BER=0. In our experiments, we use $\gamma = 0.1$ and retrain the target DNN for 5 epochs with the learning rate at the last stage of original training across all benchmarks. The threshold for code-vector extraction is set to $\tau = 0.85$ without explicit hyper-parameter tuning. We demonstrate a comprehensive examination of DeepMarks's performance (Section 5) and the comparison with the state-of-the-art DNN watermarking technique (Section 5.2) as follows.

Table 2: Benchmark neural network architectures. Here, $64C3(1)$ indicates a convolutional layer with 64 output channels and 3×3 filters applied with a stride of 2, $MP2(1)$ denotes a max-pooling layer over regions of size 2×2 and stride of 1, and $512FC$ is a fully-connected layer consisting of 512 output neurons. ReLU is used as the activation function in all the two benchmarks.

Dataset	Model Type	Architecture
MNIST	CNN	784-32C3(1)-32C3(1)-MP2(1)-64C3(1)-64C3(1)-512FC-10FC
CIFAR10	WRN	Please refer to [24]

Table 3: Fidelity requirement. The baseline accuracy is preserved after fingerprint embedding in the underlying benchmarks.

Benchmark	MNIST-CNN			CIFAR10-WRN		
Setting	Baseline	Fine-tune without fingerprint	Fine-tune with fingerprint	Baseline	Fine-tune without fingerprint	Fine-tune with fingerprint
Test Accuracy (%)	99.52	99.66	99.72	91.85	91.99	92.03

5.1 DeepMarks Properties Evaluation

We characterize the performance of DeepMarks based on the requirements discussed in Table 1 as follows.

5.1.1 Fidelity. **DeepMarks meets the fidelity criterion by preserving the model's functionality.** To study the effect of FP embedding on the functionality of the original task, we compare the test accuracy of the pre-trained baseline model, the fine-tuned model with and without the FP embedding loss. The results are summarized in Table 3. One can see from the comparison that embedding FPs in the DNN does not induce accuracy drop and can even slightly improve the accuracy of the target DNN. This is due to the fact that the additive embedding loss in Equation (2) introduces regularization and alleviates model over-fitting.

5.1.2 Security. **DeepMarks respects the security criterion by preserving the intrinsic distribution of weights.** To prevent the adversary from detecting the existence of a FP in the model, security requires the embedding of the fingerprint to leave no tangible changes in the distribution of the model parameters. Figure 2 shows the histograms of weights at the selected layer with and without the FP on the CIFAR10-WRN benchmark. The similarity between these two histograms corroborates DeepMarks security.

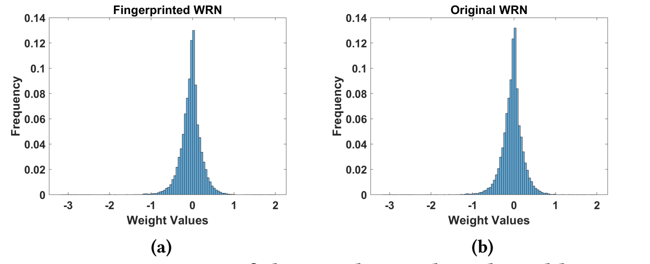

Figure 2: Histogram of the weights at the selected layer in the fingerprinted model (a) and the original model (b).

5.1.3 Robustness, Reliability, and Integrity. **DeepMarks yields high detection rates and low false alarm rates for user identification and colluder detection under various attacks.** We consider two FP deconstruction attacks: FP collusion and FP overwriting, and two model modification attacks: model fine-tuning and parameter pruning as discussed in Section 3.2. For a given number of colluders, we run 1,000 random simulations to generate different colluders sets from all users and report the average performance. When the colluder set is too large to be uniquely identified by the property of ACC, we consider all feasible colluder sets that match the extracted code-vector resulting from FP collusion. We detail the settings and results of each attack as follows.

(I) Fingerprints Collusion. The FP averaging attack is described in Section 4.2.2. Figure 3 shows the detection (true positive) rates and false alarm (false positive) rates of DeepMarks when different numbers of users participate in the collusion attack. DeepMarks features ideal detection rates and false alarm rates when the number of colluders is smaller than or equal to the theoretical threshold $(k-1)$ guaranteed by the ACC codebook. As such, DeepMarks satisfies the *reliability* and *integrity* requirements in Table 1. The consistency with the theorem corroborates that DeepMarks provides provable collusion resistance.

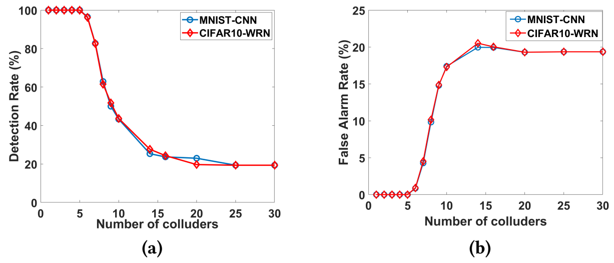

Figure 3: Detection rate (a) and false alarm rate (b) of DeepMarks against fingerprint collusion attacks.

(II) Fingerprint Overwriting. Besides FP collusion, an active adversary that is aware of the fingerprinting method may try to destroy the original FP by embedding a new one in the marked DNN. To perform the attack, the adversary generates a new set of secret matrices (**C**, **U**, **X**), construct his own FP, and randomly selects a layer in the distributed model to embed the FP as outlined in Section 4.1. In our experiment, we assume both the original FP and the new FP deploy single-layer embedding for simplicity.

Table 4 summarizes the results of DeepMarks's user identification performance when the FP overwriting attack is performed on a different layer or the same layer as the original FP. In our experiments, we assume all 31 users individually implement FP overwriting attacks on their fingerprinted models and report the average metrics. DeepMarks retains perfect user identification when the overwriting attack occurs at a different layer while incurs false negatives (non-zero BER) when the same layer is attacked.

Table 4: DeepMarks's User identification in case of FP overwriting attack at a different or the same layer.

Overwrite Condition	Overwrite Different Layer		Overwrite Same Layer	
Metrics	Accuracy (%)	BER	Accuracy (%)	BER
MNIST-CNN	99.68	0	99.69	0.06
CIFAR10-WRN	91.90	0	91.96	0.01

We further assess DeepMarks' collusion resilience against FP overwriting attacks and show the results in Figure 4. In this case, the colluders first agree on which layers to embed their FPs and obtain the overwritten models. Then the weights at the attacked layers are averaged across colluders and used as the response to the owner's query. Comparing Figure 4 with Figure 3a, it can be seen that DeepMarks' colluders detection performance is not degraded by FP overwriting if the colluders embed their new FPs in different layers as the original one. When the originally marked layer is attacked, the detection rate has a significant drop in case of a small number of colluders. Although the colluders may embed new FPs in multiple layers to increase the chance of finding the secret embedding position of the original FP, such an approach introduces excessive regularization and might incur performance degradation.

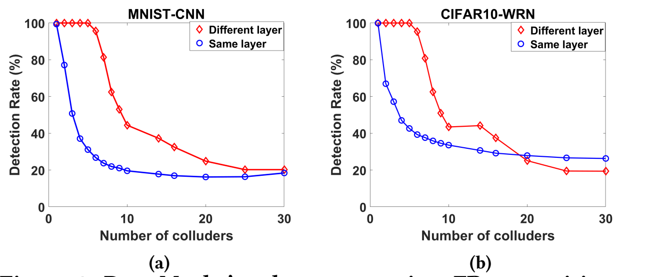

Figure 4: DeepMarks's robustness against FP overwriting at a different (red color) or the same layer (blue color).

(III) Parameter Pruning. To prune the target layer, we use the pruning method proposed in [6] and set $\alpha\%$ of the weights that possess the smallest weight values to zero. The obtained mask is then used to sparsely fine-tune the fingerprinted model on the training data with the conventional cross-entropy loss to compensate for the accuracy drop induced by pruning. We first assess the code-vector extraction (decoding) accuracy for individual users under different pruning rates and show results in Figure 5. One can see that increasing the pruning rate leads to a drop in the test accuracy, while the code-vector can always be decoded with 100% accuracy. The perfect FP decoding suggests that: (i) DeepMarks is robust against pruning attacks and reliably identifies the queried user; (ii) DeepMarks has no false alarms and satisfies the integrity criteria.

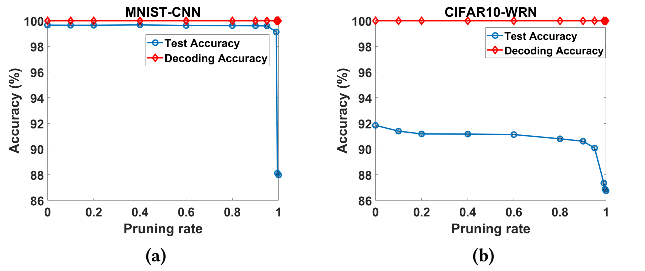

Figure 5: Code-vector extraction accuracy (red color) and test accuracy (blue color) for MNIST-CNN (a) and CIFAR10-WRN (b) benchmark under different pruning rates.

We further assess DeepMarks' robustness of colluder detection against parameter pruning. Figures 8 shows the detection rates and false alarm rates of DeepMarks framework under three different pruning rates (10%, 50%, 99%). Comparing Figure 8 with Figure 3, one can observe that DeepMarks is robust and tolerates up to 99% parameter pruning for both MNIST and CIFAR10 benchmarks.

(IV) Model Fine-tuning. Recall that the fine-tuning attack is implemented by re-training the fingerprinted model on the user's new dataset using only the conventional cross-entropy loss. We simulate this process by adding random Gaussian noise with zero mean and different standard deviations (std) to the weights of the marked DNN and extract the code-vector from the noisy weights. Figure 6 shows the test error and BER of FP detection after injecting noise on the fingerprinted model. Our key observation is that test error is more sensitive to noise compared to BER, thus the malicious user cannot remove the FP while preserving the model's performance.

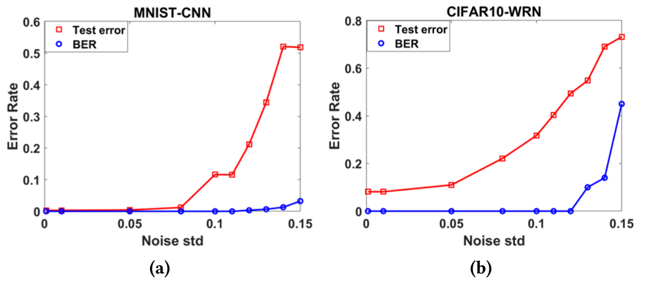

Figure 6: DeepMarks' robustness against model fine-tuning. Adding excessive noise incurs large increase of test error while the embedded FP might be removed.

In summary, DeepMarks possesses a high detection rate and a low false alarm rate when confronted with various FP deconstruction and model modification attacks. Thus, DeepMarks satisfies the **robustness, reliability** and **integrity** criteria for an effective fingerprinting technique as discussed in Table 1. The performance consistency across various benchmarks suggests its generality.

5.1.4 Scalability. **DeepMarks is applicable to large-scale distribution systems.** We define scalability of a fingerprinting technique as the number of supported users per code bit: $\beta = \frac{n}{v}$. For a $(v, k, 1)$ codebook, the maximum number of users is determined by the code-vector length v and the block size k by $n = \frac{v(v-1)}{k(k-1)}$. Thus, the scalability metric can be computed as follows:

$$\beta = \frac{v-1}{k(k-1)}. \tag{8}$$

It is straightforward to see that longer FPs provide better scalability for a fixed block size. Equation (8) also shows the trade-off between the length of the code-vector v and the collusion resilience level $(k-1)$. When the scalability is fixed, a higher resistance level requires longer fingerprinting codes. Systematic approaches to construct various BIBDs have been developed [3], providing a vast supply of ACCs for DeepMarks.

We assess DeepMarks's performance with three different codebooks $(13, 4, 1)$, $(31, 6, 1)$, $(133, 11, 1)$ BIBD, and illustrate the comparison results in Figure 7. **DeepMarks provides various levels of user capacity and collusion resistance by allowing the owner to specify codebook parameters.** Particularly, a larger codebook (e.g., $(133, 11, 1)$-BIBD ACC) accommodates more customers in the distribution system and yields better detection metrics. As such,

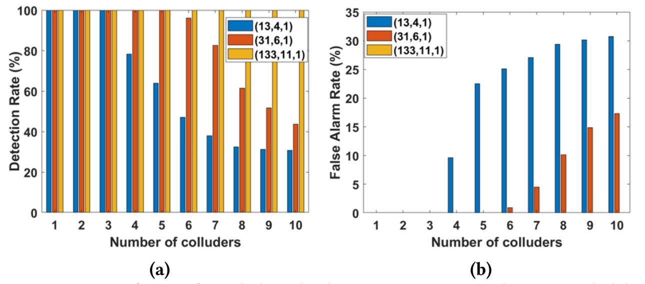

Figure 7: Effect of codebook design. DeepMarks is scalable and provides various levels of detection performance.

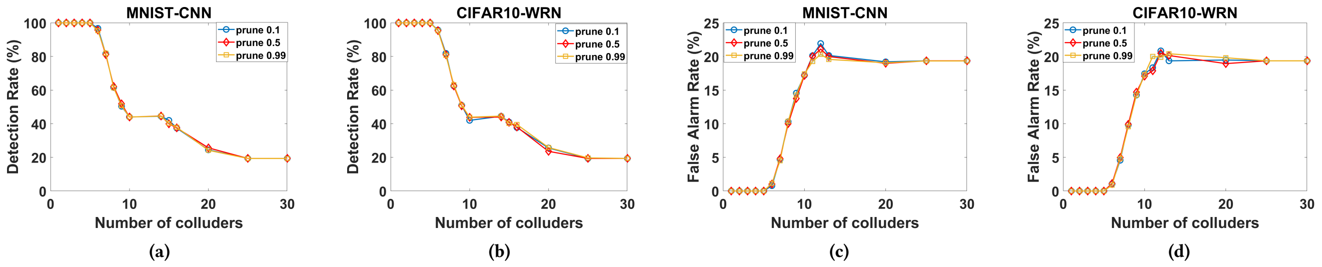

Figure 8: DeepMarks’ robustness of colluders identification against parameter pruning attack. Detection rate (a, b) and false alarm rate (c, d) of DeepMarks framework are not affected by a wide range of pruning rates.

DeepMarks framework can be customized to provide guaranteed performance based on the requirements of the IP owner.

5.1.5 *Efficiency.* **DeepMarks fingerprinting framework incurs negligible overhead and is highly efficient.** We define the efficiency of embedding and extracting an FP as the normalized runtime overhead of retraining the target DNN and recovering the code-vector from weights, respectively. To quantitatively evaluate DeepMarks’s overhead as discussed in Section 4.3, we measure the ratio of FP embedding time to original DNN training time, and the ratio of code-vector extraction time to prediction time. The results are summarized in Table 5, suggesting DeepMarks’s efficiency.

Table 5: Efficiency evaluation of DeepMarks’s FP embedding and extraction in terms of normalized runtime overhead.

Normalized Runtime Overhead (%)	FP Embedding	FP Extraction
MNIST-CNN	5.214	0.006
CIFAR10-WRN	2.562	0..056

5.2 Comparison with the State-of-the-art

In this section, we compare DeepMarks with the state-of-the-art DNN watermarking method in literature. The work of [13, 21] proposed a white-box digital watermarking technique for DL models using constraint-based watermarking. The authors evaluate their approach on CIFAR10-WRN benchmark and show that the embedded watermark tolerates up to 65% parameter pruning. For fair comparison, we also assess the performance of DeepMarks on CIFAR10-WRN benchmark and encode the FP information in the same layer as reported in the paper [21]. Compared to the prior watermarking method, **DeepMarks is more robust against parameter pruning attack** since the embedded fingerprint remains even after 99% parameters are removed (shown in Figure 5b).

We further demonstrate the superior collusion resilience of DeepMarks compared to the fingerprinting scheme that employs multiple distinct watermarks constructed by [21]. In our experiments, we assume there are 31 customers in the model distribution system and use the open-source code [20] to implement WM signature generation, WM embedding, and WM extraction. The $(31, 6, 1)$-BIBD ACC codebook is selected for DeepMarks. Table 6 summarizes the comparison results. The test accuracy of the original unmarked CIFAR10-WRN and the marked one are shown in Column 2 and Column 3, respectively. The BER of WM/FP extraction is shown in Column 4, indicating that both methods can retrieve the digital marker correctly for authenticating model authorship.

To assess the robustness of these two marking methods against FP averaging attack, we assume that the first three users collude and average the weights in the embedded layer. The result $\widetilde{\mathbf{w}}_{\mathbf{avg}} = \frac{1}{3}(\widetilde{\mathbf{w}}_1 + \widetilde{\mathbf{w}}_2 + \widetilde{\mathbf{w}}_3)$ is used as the response to the owner’s query. In our experiment, we assume user 1 produces the colluded DNN by replacing the weights in the marked layer $\widetilde{\mathbf{w}}_1$ with $\widetilde{\mathbf{w}}_{\mathbf{avg}}$ while keeping the weights in the other layers unchanged. The test accuracy of the resulting colluded model is shown in Column 5 of Table 6, which is comparable to the baseline and makes the collusion attack effective. Note that the DNN watermarking method in [21] is unaware of potential collusion attacks and does not propose any collusion detection approach. We implement colluder identification in the watermarking setting by randomly selection from all feasible colluder sets that satisfy the constraint obtained from the colluded watermark. We compare the detection rate of the collusion attack in the last column of Table 6. **DeepMarks outperforms the fingerprinting extension built on the state-of-the-art DNN watermarking [21] by achieving** 100% **detection accuracy, which is significantly higher than** 3.84%**.**

Table 6: Performance comparison between DeepMarks and the state-of-the-art DNN watermarking technique [21].

Method	Baseline Accuracy	Accuracy with WM/FP	Average BER	Colluded Accuracy	Colluders Detection Rate
Uchida et.al [19]	91.85%	92.15%	0	92.18%	**3.84%**
Ours	91.85%	92.03%	0	92.14%	**100%**

6 CONCLUSION

Deep neural networks are facilitating breakthroughs in various fields and increasingly commercialized. Systematic IP protection and digital right management for pre-trained, ready-to-deploy models has been a standing challenge. We take the first step to tackle this problem by providing an efficient, end-to-end framework that is functionality-preserving and enables coherent fingerprint insertion in the distribution of weights within the target DNN. We introduce a comprehensive set of requirements for DNN fingerprinting and empirically corroborate that DeepMarks respect all criteria. Evaluation results show that DeepMarks yields high detection rates and low false alarm rates for model ownership proof and user tracking. Furthermore, DeepMarks is the first framework that is provably collusion-secure in a large-scale model distribution system and is robust against various attacks. Our framework can be seamlessly integrated within existing DL framework (e.g., TensorFlow, PyTorch, Theano), thus paves the way for model designers to achieve reliable technology transfer. Future research directions include developing advanced codebook construction scheme for further improving collusion resistance and investigating collaborative fingerprint embedding for multiple users to improve efficiency.

REFERENCES

[1] Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. 2018. Turning Your Weakness Into a Strength: Watermarking Deep Neural Networks by Backdooring. *arXiv preprint arXiv:1802.04633* (2018).

[2] Caffe. 2017. Model Zoo. https://github.com/BVLC/caffe/wiki/Model-Zoo.

[3] Charles J Colbourn and Jeffrey H Dinitz. 2006. *Handbook of combinatorial designs*. CRC press.

[4] C Fu, A Di Fulvio, SD Clarke, D Wentzloff, SA Pozzi, and HS Kim. 2018. Artificial neural network algorithms for pulse shape discrimination and recovery of piled-up pulses in organic scintillators. *Annals of Nuclear Energy* 120 (2018), 410–421.

[5] Borko Furht and Darko Kirovski. 2004. *Multimedia security handbook*. CRC press.

[6] Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*. 1135–1143.

[7] Frank Hartung and Martin Kutter. 1999. Multimedia watermarking techniques. *Proc. IEEE* 87, 7 (1999), 1079–1107.

[8] Andrew B Kahng, John Lach, William H Mangione-Smith, Stefanus Mantik, Igor L Markov, Miodrag Potkonjak, Paul Tucker, Huijuan Wang, and Gregory Wolfe. 2001. Constraint-based watermarking techniques for design IP protection. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 20, 10 (2001), 1236–1252.

[9] Negar Kiyavash and Pierre Moulin. 2009. Performance of orthogonal fingerprinting codes under worst-case noise. *IEEE Transactions on Information Forensics and Security* 4, 3 (2009), 293–301.

[10] Deepa Kundur and Kannan Karthik. 2004. Video fingerprinting and encryption principles for digital rights management. *Proc. IEEE* 92, 6 (2004), 918–932.

[11] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436.

[12] Erwan Le Merrer, Patrick Perez, and Gilles Trédan. 2017. Adversarial Frontier Stitching for Remote Neural Network Watermarking. *arXiv preprint arXiv:1711.01894* (2017).

[13] Yuki Nagai, Yusuke Uchida, Shigeyuki Sakazawa, and ShinâĂŹichi Satoh. 2018. Digital watermarking for deep neural networks. *International Journal of Multimedia Information Retrieval* 7, 1 (2018), 3–16.

[14] Gang Qu and Miodrag Potkonjak. 2007. *Intellectual property protection in VLSI designs: theory and practice*. Springer Science & Business Media.

[15] David Ross, Brian Elmenhurst, Mark Tocci, John Forbes, and Heather Wheelock Ross. 2017. Digital fingerprinting track and trace system. US Patent 9,582,714.

[16] Bita Darvish Rouhani, Huili Chen, and Farinaz Koushanfar. 2019. DeepSigns: An End-to-End Watermarking Framework for Protecting the Ownership of Deep Neural Networks. In *The 24th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*. ACM.

[17] Jürgen Schmidhuber. 2015. Deep learning in neural networks: An overview. *Neural networks* 61 (2015), 85–117.

[18] Wade Trappe, Min Wu, and KJ Ray Liu. 2002. Collusion-resistant fingerprinting for multimedia. In *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, Vol. 4. IEEE, IV–3309.

[19] Wade Trappe, Min Wu, Z Jane Wang, and KJ Ray Liu. 2003. Anti-collusion fingerprinting for multimedia. *IEEE Transactions on Signal Processing* 51, 4 (2003), 1069–1087.

[20] Yusuke Uchida. 2017. Embedding Watermarks into Deep Neural Networks. https://github.com/yu4u/dnn-watermark.

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

[22] Min Wu, Wade Trappe, Z Jane Wang, and KJ Ray Liu. 2004. Collusion-resistant multimedia fingerprinting: a unified framework. In *Security, Steganography, and Watermarking of Multimedia Contents VI*, Vol. 5306. International Society for Optics and Photonics, 748–760.

[23] Yongsheng Yu, Hongwei Lu, Xiaosu Chen, and Zhiguang Zhang. 2010. Group-oriented anti-collusion fingerprint based on bibd code. In *e-Business and Information System Security (EBISS), 2010 2nd International Conference on*. IEEE, 1–5.

[24] Sergey Zagoruyko and Nikos Komodakis. 2016. Wide residual networks. *arXiv preprint arXiv:1605.07146* (2016).

[25] Jialong Zhang, Zhongshu Gu, Jiyong Jang, Hui Wu, Marc Ph Stoecklin, Heqing Huang, and Ian Molloy. 2018. Protecting Intellectual Property of Deep Neural Networks with Watermarking. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*. ACM, 159–172.