DeepIPR: Deep Neural Network Ownership Verification With Passports

Lixin Fan, *Senior Member, IEEE*, Kam Woh Ng, *Member, IEEE*, Chee Seng Chan, *Senior Member, IEEE*, and Qiang Yang, *Fellow, IEEE*

Abstract—With substantial amount of time, resources and human (team) efforts invested to explore and develop successful deep neural networks (DNN), there emerges an urgent need to protect these inventions from being illegally copied, redistributed, or abused without respecting the intellectual properties of legitimate owners. Following recent progresses along this line, we investigate a number of watermark-based DNN ownership verification methods in the face of ambiguity attacks, which aim to cast doubts on the ownership verification by forging counterfeit watermarks. It is shown that ambiguity attacks pose serious threats to existing DNN watermarking methods. As remedies to the above-mentioned loophole, this paper proposes novel *passport*-based DNN ownership verification schemes which are both *robust to network modifications* and *resilient to ambiguity attacks*. The gist of embedding digital passports is to design and train DNN models in a way such that, the DNN inference performance of an original task will be significantly *deteriorated due to forged passports*. In other words, genuine passports are not only verified by looking for the predefined signatures, but also reasserted by the *unyielding DNN model inference performances*. Extensive experimental results justify the effectiveness of the proposed passport-based DNN ownership verification schemes. Code is available at https://github.com/kamwoh/DeepIPR

Index Terms—Deep model protection, model ownership verification, intellectual property protection, model security, deep learning

1 Introduction

Protection of Intellectual Property Rights (IPR) has always been an issue, but has taken on new meaning and importance in the digital age. For instance, there is an urgent need to protect invented deep neural networks (DNN) models from being illegally copied, redistributed or abused (i.e., intellectual property infringement) as deep learning has revolutionized many tasks such as machine translation, speech recognition, face recognition, and photo-realistic image generation. As DNN models, e.g., pretrained language models are becoming extremely computationally complex and training data hungry, building a successful DNN is not a trivial task, which usually requires substantial investments on expertise, time and resources.

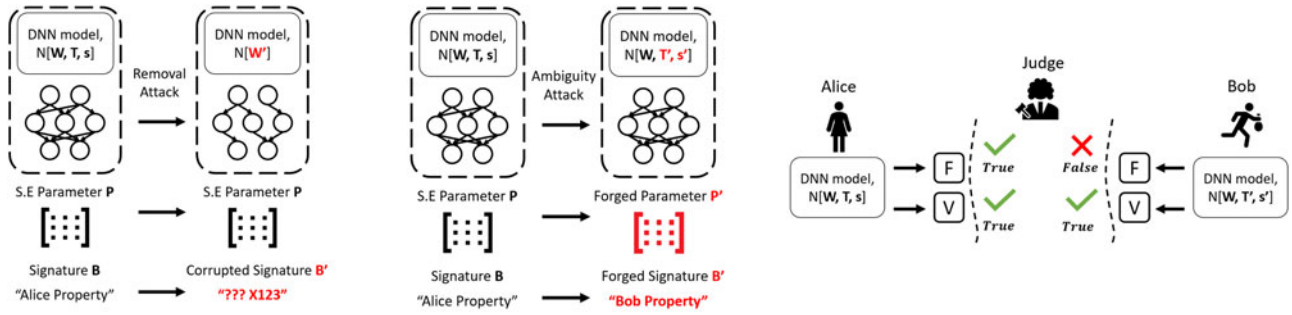
Recently, digital *watermarking* - a technology often used to protect copyright of multimedia data was adopted to provide such an IP protection, by embedding watermarks into DNN models during the training stage. Subsequently, ownerships of these inventions are verified by the detection of the embedded watermarks, which are supposed to be robust to multiple types of modifications such as model fine-tuning, model pruning and watermark overwriting [1], [2], [3], [4]. Generally, these approaches can be broadly categorized into two schools: a) the *feature-based* methods that embed designated watermarks into the DNN weights by imposing additional regularization terms [1], [3], [5]; and b) the *trigger-set* based methods that rely on adversarial training samples with specific labels (i.e., backdoor trigger sets) [2], [4]. Watermarks embedded with either of these methods have successfully demonstrated robustness against *removal attacks* which involve modifications of the DNN weights such as *fine-tuning* or *pruning*. However, our studies disclose the existence and effectiveness of *ambiguity attacks* which aim to cast doubt on the ownership verification by *forging additional watermarks*[1] for DNN models in question (see Fig. 1). We also show that *it is always possible to create a forged watermark at minor computational cost* where the original training dataset is also not needed (Section 3).

As remedies to the above-mentioned loophole, this paper proposes a novel *passport*-based strategy, which is fundamentally different from the watermark-based approaches and provide protection against the weakness of watermarks. Specifically, we propose to *modulate the inference performances of the DNN model depending on the presented passports*, i.e., the inference performance of a pre-trained DNN model will either remain intact given the presence of valid passports, or be significantly deteriorated due to either the modified or forged passports. By taking advantage of the modulated DNN model

- *Lixin Fan is with WeBank AI Lab, Shenzhen 518052, China. E-mail: lixinfan@webank.com.*
- *Kam Woh Ng is with the University of Surrey, GU2 7XH Guildford, U.K. E-mail: kamwoh@gmail.com.*
- *Chee Seng Chan is with the Center of Image and Signal Processing (CISiP), Faculty of Computer Science and Information Technology, Universiti Malaya, Kuala Lumpur 50603, Malaysia. E-mail: cs.chan@um.edu.my.*
- *Qiang Yang is with Webank, Shenzhen 518052, China, and also with the Hong Kong University of Science and Technology, Hong Kong. E-mail: qyang@cse.ust.hk.*

Manuscript received 3 Sept. 2020; revised 20 Mar. 2021; accepted 6 June 2021. Date of publication 14 June 2021; date of current version 9 Sept. 2022. (Corresponding author: Chee Seng Chan.) Recommended for acceptance by D. Crandall. Digital Object Identifier no. 10.1109/TPAMI.2021.3088846

1. Noted that *watermark*, *signature* and *trigger set* are used interchangeably. We would sometime describe *watermark* to include both *signature* (could be a string) and *trigger set* (data that is to "trigger" the model to output designated labels).



(a) Removal Attack. (b) Ambiguity Attack. (c) Proposed Solution.

Fig. 1. Two threat models considered in this work i.e., removal attack and ambiguity attack, and the proposed solution to defeat both attacks. *(a)* Removal attack aims to remove or overwrite original watermark $\mathbf{B}$, by modifying DNN model weights W to W' (marked red); *(b)* Ambiguity attack aims to forge counterfeit watermarks $(\mathbf{T}', \mathbf{s}')$ *without* modifying DNN model weights W (see Proposition 1 and Section 3.2); *(c)* A *non-invertible* verification scheme is proposed to defeat ambiguity attacks, whereas the attacker Bob is unable to forge a new watermark that can pass both verification process V and fidelity process F (see Definition 1 and Section 4.4).

performance, which is an unique feature of the proposed passport-based approach, one can develop ownership verification schemes that are both robust to removal attacks and resilient to ambiguity attacks at once (Section 4). Moreover, we introduce a novel *sign loss* that is to embed binary signature into the scale factors of a passport layer. The binary signature embedded guarantee strong resilient to ambiguity attacks (Section 4.3).

The *contributions* of our work are twofold:

- We put forth a general formulation of DNN ownership verification schemes (Definition 1) to defeat both removal attacks and ambiguity attacks. It is shown by Proposition 1 that existing watermark-based schemes are invertible processes and thus are volunerable to ambiguity attacks. The feasibility of employing an non-invertible process is then given by Proposition 2.
- We propose novel passport-based verification schemes and demonstrate with extensive experiment results that these schemes successfully defeat both removal attacks and ambiguity attacks. Passport-embedded DNN networks are designed in a way such that, the DNN inference performance of an original task will be significantly deteriorated due to forged passports. In other words, genuine passports are not only verified by looking for the predefined signatures, but also reasserted by the unyielding DNN model inference performances.

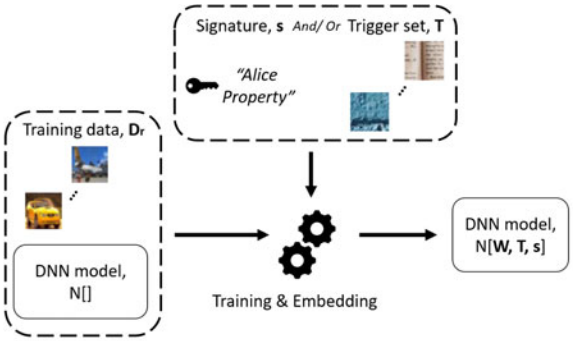
A preliminary version of this work was presented earlier [6]. The present work extends the initial version in three aspects. First, we complete a thorough investigation and disclose the existence of ambiguity attacks that will cast doubt on the existing watermark-based solutions. Second, considerable new analyses and empirical studies are added to the initial results. For instance, new experiments using ImageNet are added in the study. Third, to further demonstrate the ability of our proposed model, we proposed an innovative way, i.e., embed sign of scale factors as signature to guarantee strong resilient to ambiguity attacks.

2 Related Work

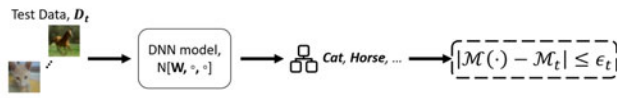
Uchida *et al.* [1] was probably the first work that proposed to embed watermarks into DNN models by imposing an additional *regularization term* on the weights parameters. [2], [7] proposed to embed watermarks in the classification labels of adversarial examples in a *trigger set*, so that the watermarks can be extracted remotely through a service API without the need to access the network weights (i.e., black-box setting). Also in both black-box and white box settings, [3], [5], [8], [9], [10], [11], [12] demonstrated how to embed watermarks (or fingerprints) that are robust to various types of attacks. In particular, it was shown that embedded watermarks are in general robust to *removal attacks* that modify network weights via fine-tuning or pruning. Watermark overwriting, on the other hand, is more problematic since it aims to simultaneously embed a new watermark and destroy the existing one. Although [5] demonstrated robustness against overwriting attack, it did not resolve the ambiguity resulted from the counterfeit watermark. Adi *et al.* [2] also discussed how to deal with an adversary who fine-tuned an already watermarked networks with new trigger set images. Nevertheless, [2] required the new set of images to be distinguishable from the true trigger set images. This requirement is however often unfulfilled in practice, and our experiment results show that none of existing watermarking methods are able to deal with ambiguity attacks explored in this paper (see Section 3). For a more comprehensive survey, please refer to [13].

In the context of digital image watermarking, [14], [15] have studied *ambiguity attacks* that aim to create an ambiguous situation in which a watermark is reverse-engineered from an already watermarked image, by taking advantage of the invertibility of forged watermarks [16]. It was argued that *robust watermarks do not necessarily imply the ability to establish ownership*, unless *non-invertible watermarking* schemes are employed (see Proposition 2 for our proposed solution).

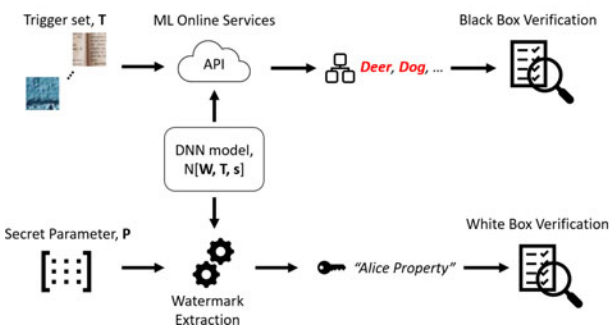
In our work, we aim to protect a DNN model by embedding a watermark that is *unique and non-removable*. *Ambiguity attacks* aim to forge a counterfeit watermark by inserting the new counterfeit watermark into the model while maintaining the trained weights and the performance of the model. Although the motivation of inserting new watermark into the model for ambiguity attack is very similar for watermark overwriting i.e., try to cheat the verification process (see III in Definition 1 and Fig. 2c), we would like to highlight that the approaches taken by both attacks are fundamentally different:



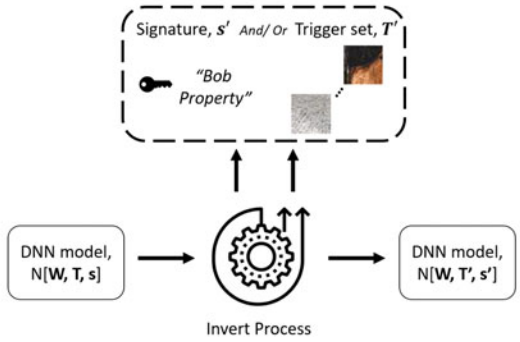
(a) An embedding process E takes as inputs 1) normal training dataset D_r; 2) watermarks to be embedded $\mathbf{T}$ and/or $\mathbf{s}$; 3) network architecture $\mathbb{N}[\,]$. The final outcome is a model $\mathbb{N}[\mathbf{W}, \mathbf{T}, \mathbf{s}]$ with embedded $\mathbf{T}$ and/or $\mathbf{s}$ and learned weight $\mathbf{W}$.



(b) A fidelity evaluation process F, takes as inputs 1) a trained model $\mathbb{N}[\mathbf{W}, \cdot, \cdot]$; 2) a set of test data D_t; 3) a target model performance $\mathcal{M}_t$. The final decision is whether the evaluated model performance is close to $\mathcal{M}_t$ within a predefined threshold ϵ_t.



(c) A verification process V takes as inputs, either a trigger set $\mathbf{T}$ or secret parameters $\mathbf{P}$, and outputs decision whether the DNN model in question is embedded with designated signature.



(d) An invert process I takes a trained DNN model $\mathbb{N}[\mathbf{W}, \mathbf{T}, \mathbf{s}]$ as input. The outcome is the model with forged watermark $\mathbb{N}[\mathbf{W}, \mathbf{T}', \mathbf{s}']$.

Fig. 2. Visual explanation for processes E, F, V, I defined in Definition 1.

a) Overwriting attacks in [1], [5] modify network weights W to embed new watermark; b) Ambiguity attacks do not need to modify any trained network weights W, instead, new watermarks are forged by an invert process to fool the verification process. (see IV in Defintion 1 and Fig. 2d); c) Existing watermarking methods are proved to be robust to overwriting attacks [5] but are vulnerable to ambiguity attacks (see Proposition 1 and Section 3.2).

3 Rethinking Deep Neural Network Ownership Verification

This section analyses and generalizes existing DNN watermarking methods in the face of ambiguity attacks. We must emphasize that the analysis mainly focuses on three aspects i.e., *fidelity*, *robustness* and *invertibility* of the ownership verification schemes, and we refer readers to representative previous work [1], [2], [3], [4] for formulations and other desired features of the entire watermark-based intellectual property (IP) protection schemes.

3.1 Reformulation of DNN Ownership Verification Schemes

Fig. 1 summarizes the application scenario and threat model of DNN model ownership verification. Two types of threat models are considered in our work, i.e., removal attack (Fig. 1a) and ambiguity attack (Fig. 1b). A clear distinction between these two attacks lies in the fact that attackers have to modify DNN model weights to remove or overwrite embedded watermarks, while forged watermarks can be created with DNN model weights being kept intact for ambiguity attack. Fig. 1c illustrates an ambiguous situation in which rightful ownerships cannot be uniquely resolved by existing watermarking methods. This loophole is largely due to an intrinsic weakness of the watermark-based methods i.e., *invertibility*, which can be resolved by employing the passport-based approach. Formally, the definition of DNN model ownership verification schemes is generalized as follows.

Definition 1. *A DNN model ownership verification scheme is a tuple $\mathcal{V} = (E, F, V, I)$ of processes:*

I) An embedding *process $E(\mathbf{D}_r, \mathbf{T}, \mathbf{s}, \mathbb{N}[\cdot], L) = \mathbb{N}[\mathbf{W}, T, s]$, is a DNN learning process that takes* training data *$\mathbf{D}_r = \{\mathbf{X}_r, \mathbf{y}_r\}$ as inputs, and additionally together with, either trigger set data $\mathbf{T} = \{\mathbf{X}_T, \mathbf{y}_T\}$ or signature $\mathbf{s}$, and outputs the model $\mathbb{N}[\mathbf{W}, \mathbf{T}, \mathbf{s}]$ by minimizing a given loss L. (see Fig. 2a)*

Remark*: the DNN architectures are pre-determined by $\mathbb{N}[\cdot]$ and, after the DNN weights $\mathbf{W}$ are learned, either the trigger set $\mathbf{T}$ or signatures $\mathbf{s}$ will be embedded and can be verified by the verification process defined next.*[2]

2. Learning hyper-parameters such as learning rate and the type of optimization methods are considered irrelevant to ownership verifications, and thus they are not included in the formulation.

II) *A* fidelity evaluation *process* $F\big(\mathbb{N}[\mathbf{W},\cdot,\cdot],\mathbf{D}_t,\mathcal{M}_t,\epsilon_f\big)=\{True,\ False\}$ *is to evaluate whether or not the discrepancy of model performances* κ *is less than a predefined threshold i.e.,* $|\mathcal{M}(\mathbb{N}[\mathbf{W},\cdot,\cdot],\mathbf{D}_t)-\mathcal{M}_t|\leq\epsilon_f$*, in which* $\mathcal{M}(\mathbb{N}[\mathbf{W},\cdot,\cdot],\mathbf{D}_t)$ *is the DNN inference performance tested against a set of* test data $\mathbf{D}_t$ *where* $\mathcal{M}_t$ *is the target inference performance. (see Fig. 2b)*

Remark*: it is often expected that a well-behaved embedding process will not introduce a significant inference performance change that is greater than a predefined threshold* ϵ_f*. Nevertheless, this fidelity condition remains to be verified for DNN models under either removal attacks or ambiguity attacks.*

III) *A* verification *process* $V(\mathbb{N}[\mathbf{W},\cdot,\cdot],\mathbf{T},\mathbf{s},\epsilon_s)=\{True,\ False\}$ *checks whether or not the expected signature* $\mathbf{s}$ *or trigger set* $\mathbf{T}$ *is successfully verified for a given DNN model* $\mathbb{N}[\mathbf{W},\cdot,\cdot]$*. (see Fig. 2c)*

Remark*: for feature-based schemes,* V *involves the detection of embedded signatures* $\mathbf{s}=\{\mathbf{P},\mathbf{B}\}$ *with a false detection rate that is lesser than a predefined threshold* ϵ_s*. Specifically, the detection boils down to check whether the Hamming distance* $H(f_e,\mathbf{B})$ *is below a Hamming radius* ϵ_s*, in which* $f_e(\mathbf{W},\mathbf{P})=\mathbf{PW}$*.* V = True *if* $H\leq\epsilon_s$ *and* False *otherwise.*

Remark*: for trigger-set based schemes,* V *first invokes a DNN inference process that takes trigger set samples* $\mathbf{X}_T$ *as inputs, and then it checks whether the prediction* $f(\mathbf{W},\mathbf{X}_T)$ *produces the designated labels* $\mathbf{y}_T$ *with a false detection rate lesser than a threshold* ϵ_t*.* V = True *if* $\sum_i\mathbf{1}(f(W,\mathbf{X}_T^{(i)})\neq\mathbf{y}_T^{(i)})\leq\epsilon_s$ *and* False *otherwise in which* $\mathbf{1}()$ *is an indicator function.*

IV) *An* invert *process* $I(\mathbb{N}[\mathbf{W},\mathbf{T},\mathbf{s}])=\mathbb{N}[\mathbf{W},\mathbf{T}',\mathbf{s}']$ *exists and constitutes a successful* ambiguity attack *(see Fig. 2d), if a set of new trigger set* $\mathbf{T}'$ *and/or signature* $\mathbf{s}'$ *can be reverse-engineered for a given DNN model:*

- *a) the forged* $\mathbf{T}',\mathbf{s}'$ *can be successfully verified with respect to the given DNN weights* $\mathbf{W}$ *i.e.,* $V(I(\mathbb{N}[\mathbf{W},\mathbf{T},\mathbf{s}]),\mathbf{T}',\mathbf{s}',\epsilon_s)=True$*;*
- *b) the fidelity evaluation outcome* $F\big(\mathbb{N}[\mathbf{W},\cdot,\cdot],\mathbf{D}_t,\mathcal{M}_t,\epsilon_f\big)$ *defined in Definition 1.II remains* True.

Remark*: this condition plays an indispensable role in designing the* non-invertible *verification schemes to defeat ambiguity attacks (see Section 4.4).*

V) *If at least one invert process exists for a DNN verification scheme* $\mathcal{V}$*, then the scheme is called an* invertible *scheme and denoted by* $\mathcal{V}^I=(E,F,V,I\neq\emptyset)$*; otherwise, the scheme is called* non-invertible *and denoted by* $\mathcal{V}^\emptyset=(E,F,V,\emptyset)$*.*

The definition as such is abstract and can be instantiated by concrete implementations of processes and functions (All notations are summarized in Table 1). For instance, the following combined loss function (Eq. (1)) generalizes loss functions adopted by both the feature-based and trigger-set based watermarking methods

$$L=L_c\big(f(\mathbf{W},\mathbf{X}_r),\mathbf{y}_r\big)+\lambda^t L_c\big(f(\mathbf{W},\mathbf{X}_T),\mathbf{y}_T\big)+\lambda^r R(\mathbf{W},\mathbf{s}),\tag{1}$$

in which λ^t,λ^r are the relative weight hyper-parameters, $f(\mathbf{W},\mathbf{X}_-)$ are the network predictions with inputs $\mathbf{X}_r$ or $\mathbf{X}_T$.

TABLE 1
All Notations

Notation	Description
E	Embedding process. This process will train a DNN model and embed watermarks into the DNN model.
F	Fidelity evaluation process.
V	Verification process. $\mathbb{E}[V]$ is the expected (average) detection rate of verification results.
I	Invert process.
L_c	Loss function such as *cross-entropy*.
$\mathbf{s}=\{\mathbf{P},\mathbf{B}\}$	**s**. Signature to be embedded, usually include a signature extraction parameter **P** and binary signature string **B**. **P**. In feature-based watermark methods, **P** is a signature extraction parameter to recover hidden signature string **B** from a watermarked DNN. In our passport-based method, **P** is *passport* that is to compute signature string **B** and to modulate the passport layer scale γ and bias β (see Fig. 6a). **B**. A binary signature string where $\mathbf{B}=\{-1,1\}^C$.
$\mathbf{T}$	Trigger set data including inputs $\mathbf{X}_T$ and designated labels $\mathbf{y}_T$.
$\mathbf{D}$	A dataset including inputs $\mathbf{X}$ and labels $\mathbf{y}$.
$\mathbb{N}[\cdot]$	An initial DNN model with architecture specified and weights to be trained.
$\mathbb{N}[\mathbf{W}]$	A trained DNN model that has trained weight $\mathbf{W}$.
$\mathbb{N}[\mathbf{W},\mathbf{s}]$	A trained DNN model that has trained weight $\mathbf{W}$ embedded with signature $\mathbf{B}$, that is extracted by using parameter $\mathbf{P}$. $\mathbf{s}$ consists of both $\mathbf{P}$ and $\mathbf{B}$.
$\mathbb{N}[\mathbf{W},\mathbf{T}]$	A trained DNN model that has trained weight $\mathbf{W}$ embedded with trigger set $\mathbf{T}$.
$\mathbb{N}[\mathbf{W},\mathbf{T},\mathbf{s}]$	A trained DNN model that has trained weight $\mathbf{W}$, embedded with watermarks (which are trigger set $\mathbf{T}$ and signature $\mathbf{s}$).
$\mathcal{M}$	The performance of a model.
$\mathcal{M}_t$	Target inference performance.
ϵ_f	A predefined threshold for discrepancy measurement.
ϵ_s	A predefined Hamming radius for signature string measurement.
ϵ_t	A predefined threshold for false detection rate of trigger set samples.

L_c is the loss function like *cross-entropy* that penalizes discrepancies between the predictions and the target labels $\mathbf{y}_r$ or $\mathbf{y}_T$. Signature $\mathbf{s}=\{\mathbf{P},\mathbf{B}\}$ consists of signature extraction parameter $\mathbf{P}$ and signature string $\mathbf{B}$. The regularization terms could be either $R=L_c(\sigma(\mathbf{W},\mathbf{P}),\mathbf{B})$ as in [1] in which $\sigma()$ is sigmoid function or $R=\mathrm{MSE}(\mathbf{B}-\mathbf{PW})$ as in [3] in which MSE is mean square error.

It must be noted that, for those DNN models that will be used for classification tasks, their inference performance $\mathcal{M}(\mathbb{N}[\mathbf{W},\cdot,\cdot],\mathbf{D}_t)=L_c\big(f(\mathbf{W},\mathbf{X}_t),\mathbf{y}_t\big)$ tested against a dataset $\mathbf{D}_t=\{\mathbf{X}_t,\mathbf{y}_t\}$ is independent of either the embedded signature $\mathbf{s}$ or trigger set $\mathbf{T}$. It is this independence that induces an invertible process for existing watermark-based methods as disclosed by following proposition.

Proposition 1. *(*Invertible process*) propmainclaim For a DNN ownership verification scheme* $\mathcal{V}$ *as in Definition 1, if the fidelity process* $F()$ *is independent of either the signature* $\mathbf{s}=\{\mathbf{P},\mathbf{B}\}$ *or trigger set* $\mathbf{T}$*, then there always exists an invertible process* $I()$ *i.e., the scheme is invertible* $\mathcal{V}^I=(E,F,V,I\neq\emptyset)$*).*

Proof. for a trained network $\mathbb{N}[\hat{\mathbf{W}}, T, s]$ with signature $\mathbf{s}$ and/or trigger set $\mathbf{T}$ embedded, the invert process $I()$ can be constructed with the following steps:

1) maintain the given weights $\hat{\mathbf{W}}$ unchanged;
2) forge the *feature-based* watermarks $\mathbf{s}' = \{\mathbf{P}', \mathbf{B}'\}$ by minimizing the distance $\mathbf{P}' = arg\min_{\mathbf{P}'} H(f_e(\hat{\mathbf{W}}, P'), \mathbf{B}')$.
   Remark: attackers have to take $\mathbf{B}' \neq \mathbf{B}$, and in case that the watermark signature $\mathbf{B}$ is unknown, attackers may assign random signature $\mathbf{B}'$, whose the probability of collision $\mathbf{B}' = \mathbf{B}$ is then exponentially low.
3) forge the trigger set $T' = \{\mathbf{X}'_T, \mathbf{y}'_T\}$ by minimizing the (cross-entropy) loss $\mathbf{X}'_T = arg\min_{\mathbf{X}'_T} L_c\big(f(\hat{\mathbf{W}}, \mathbf{X}'_T), \mathbf{y}'_T\big)$ between the prediction and the target labels.
4) fidelity evaluation is fulfilled i.e., $|\mathcal{M}(\mathbb{N}[\mathbf{W}, \cdot, \cdot], \mathbf{D}_t) - \mathcal{M}_t| \leq \epsilon_f$ since model performance is independent to both the forged signatures and trigger set, thus remain unchanged.

Remark: during the minimization of detection error, there is *no need of training data* which is not used in step 2 at all;

Remark: during the minimization of detection error, the *computational cost is minor* since the dimensionality of the optimization parameters i.e., $\{\mathbf{P}', \mathbf{B}'\}$ or $\mathbf{X}'_T, \mathbf{y}'_T$ is order of magnitude smaller, as compared to the number of DNN weights $\hat{\mathbf{W}}$. □

3.2 Threat Model: Conventional Watermark-Based DNN in the Face of Ambiguity Attacks

In this section, we investigate a number of popular watermark-based DNN ownership verification methods [1], [2] in the face of ambiguity attacks, which aim to cast doubts on ownership verification by forging counterfeit watermarks.

3.2.1 Ambiguity Attacks on Feature-Based Method [1]

Herein, one may train a DNN model embedded with watermarks as described in [1], then the ambiguity attacks are launched as follows. The loss function adopted in [1] uses the following binary cross entropy for the embedding regularizer:

$$R(W) = -\sum_{j=1}^{C}(b_j \log(y_j) + (1 - b_j)\log(1 - y_j)), \tag{2}$$

in which $y_j = \sigma(\sum_i \mathbf{P}_{ji} w_i)$ is the extracted feature with $\sigma(\cdot)$ the sigmoid function. In order to forge watermark $\mathbf{P}$ for a given signature $\mathbf{B} = \{b_1, \ldots, b_C\} \in \{-1, 1\}^C$ and the weights w_i of the watermarked DNN model, the loss (Eq. (2)) is minimized with respect to the new binary signatures $\mathbf{B}'$.

Following [1], we detect watermarks by comparing the extracted binary strings w.r.t. the designated one by measuring the successful detection rate. As summarized in Table 2, for both real/fake watermarks, E(V) = 100%, and F = *True* (since W remains unchanged). It is impossible to tell the real from the counterfeit watermarks. The verification scheme V is invertible. Even after the fine-tuning (a typical removal attack), E(V) for both real and fake watermarks remain 100 percent. It is still impossible to distinguish real from counterfeit watermarks.

TABLE 2
Accuracy of the Classification Task $\mathcal{M}$ and Detection Rate of *Real/Fake* Embedded Watermarks $\mathbb{E}[V]$ (Both in %) With Two Representative Watermark-Based DNN Methods [1], [2], Before (Trained With CIFAR10) and After the DNN Weights are Fine-Tuning for a Transfer Learning Task (i.e., CIFAR100 and Caltech-101)

		Trained with	Fine-tuned with	
AlexNet		CIFAR10	CIFAR100	Caltech-101
Feature based method [1]	$\mathcal{M}$	90.97	64.25	74.08
	$\mathbb{E}[V]$	100/100	100/100	100/100
Trigger-set based method [2]	$\mathcal{M}$	91.03	65.20	75.06
	$\mathbb{E}[V]$	100/100	25.00/27.80	43.60/46.80

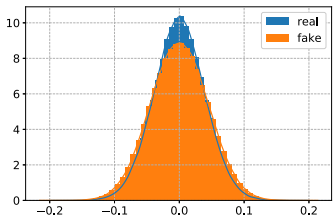
Note that since w_i are fixed, we do not need to include the original (cross-entropy) loss measured with the training images, which is a constant during the optimization. This simplicity allows the forging of P_{ji} converge very rapidly. Note that, the overall optimization took about only 50 iterations in 50 seconds, which merely constitutes a minor fraction (2.5 percent) of the training time for the original task.

Fig. 3a illustrates the distributions of counterfeit watermarks $\mathbf{P}_{ji}$ together with the original watermarks, which are hardly distinguishable from each other. In terms of the extracted features $\sum_i \mathbf{P}_{ji} w_i$, their distributions are different from the original watermarks, but it is still impossible to tell the difference after thresholding for the purpose of ownership verification. Finally, Fig. 3c illustrates that the distribution of $\mathbf{P}_{ji}$ is not much affected by the fine-tuning process which aims to modify the DNN weights for transfer learning purposes (see Table 2).

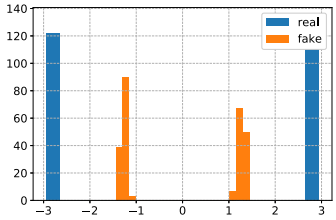
3.2.2 Ambiguity Attacks on Trigger-Set Based Method [2]

One may follow [2] to train the DNN model with trigger set images embedded as watermarks, and then the ambiguity attacks are conducted as follows. In order to construct the adversarial trigger set images by minimizing the cross-entropy loss between the predicted labels and the target labels, one may adopt a simple approach which adds trainable noise components to randomly selected base images using the following steps:

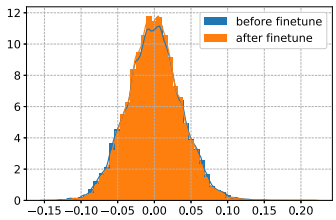
1) Randomly select a set of N base images $\mathbf{X}_b$ as shown in Fig. 4a;
2) Make random noisy patterns of the same size $\mathbf{X}_n$ as trainable parameters;
3) Use the summed components $\mathbf{X}_T = \mathbf{X}_b + \eta \mathbf{X}_n$ as the trigger set images, in which $\eta = 0.04$ to make the noise component invisible;
4) Randomly assign trigger set labels $\mathbf{y}_T$;
(5) Minimize the cross-entropy loss $L_c\big(f(\hat{\mathbf{W}}, \mathbf{X}_T), \mathbf{y}_T\big)$ w. r.t. the trainable parameter $\mathbf{X}_n$ in which $\mathbf{X}_n = arg\min_{\mathbf{X}_n} L_c$.
   Remark: DNN parameters $\hat{\mathbf{W}}$ are fixed during the optimization, and thus, the original training data is not needed.



(a) Distribution of the watermarks.



(b) Histogram of the extracted features.



(c) Distribution of the counterfeit watermarks after fine-tuning.

Fig. 3. A comparison of the distributions of watermarks and extracted features.

Fig. 4c illustrates the final optimized $\mathbf{X}_T$. where all of them are correctly classified as the assigned labels i.e., $\mathbf{y}_T$. Visually, these forged trigger set images (Fig. 4c) are hardly distinguishable from the original ones (Fig. 4a). In terms of histogram distributions, they are indistinguishable too (see Fig. 5). As shown in Table 2, both the trigger set and forged images are 100 percent correctly labeled with assigned adversarial labels. This indistinguishable situation casts doubt on ownership verification by trigger set images alone.

After fine-tuned to other classification tasks, however, the classification accuracies of both trigger set and forged images deteriorated drastically yet the detection rate of forged images is slightly better than that of the original trigger set images. We ascribed this improvement to the invert process, which optimizes $\mathbf{X}_n$ to increase the detection rate. In terms of the computational cost, the overall optimization requires only about 100 epochs of fake trigger set in 100 seconds, which merely constitutes a minor fraction (5 percent) of the training time for the original task.

As a summary, we found out that theoretically, as proved by Proposition 1, one is able to construct forged watermarks for any already watermarked networks. Empirically, we tested the performances of two representative DNN watermarking methods [1], [2], and Table 2 shows that counterfeit watermarks can be forged for the given DNN models with 100 percent detection rate, and 100 percent fake trigger set images can be reconstructed as well in the original task. Given that the detection accuracies for the forged trigger set is slightly better than the original trigger set after fine-tuning, the claim of the ownership is ambiguous and cannot be resolved by neither feature-based nor trigger-set based watermarking methods. Considering that the computational cost to forge counterfeit watermarks is minor, and that fake watermarks are successfully forged without the need of original training data, we see ambiguity attacks pose serious challenges to watermark-based IPR protections.

As a whole, the ambiguity attacks against conventional DNN watermarking methods are effective with minor computational and without the need of original training datasets. We ascribe this loophole to the crux that the loss of the original task i.e., $L_c\big(f(\hat{\mathbf{W}}, \mathbf{X}_r), \mathbf{y}_r\big)$ is *independent* of the forged watermarks. In the next section, we shall illustrate a solution to defeat the ambiguity attacks.

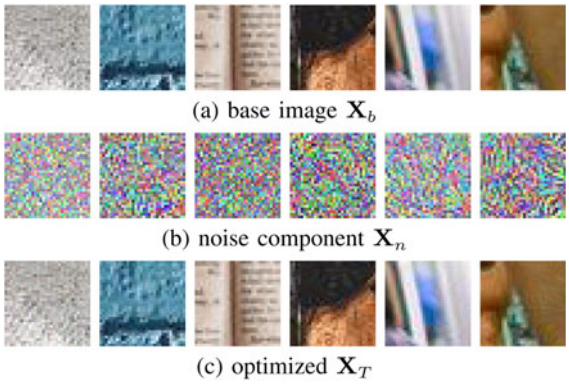


Fig. 4. Sample of the trigger set images used in ambiguity attacks on trigger-set based method in Section 3.2.2.

4 Embedding Passports for DNN Ownership Verification

The main motivation of embedding digital passports is to design and train DNN models in a way such that, their inference performances of the original task (i.e., classification accuracy) will be significantly *deteriorated due to the forged signatures*. We shall illustrate next first how to implement the desired property by incorporating the so called *passport layers*, followed by different ownership protection schemes that exploit the embedded passports to effectively defeat ambiguity attacks.

4.1 Passport Layers

In order to control the DNN model functionalities by the embedded digital signatures i.e., *passports*, we proposed to append after a convolution layer a *passport layer*, whose scale factor $\gamma \in \mathbb{R}^{O_C}$ and bias shift term $\beta \in \mathbb{R}^{O_C}$ are dependent on both the convolution kernels $\mathbf{W}_p \in \mathbb{R}^{O_C \times I_C \times K \times K}$ (in which O_C is number of output channels, I_C is number of input channels, and K is kernel size), and the designated passport $\mathbf{P} \in \mathbb{R}^{N_P \times I_C \times I_S \times I_S}$ (in which N_P is number of passports and I_S is size of image or feature map) as follows:

$$\mathbf{O}^{(l)}(\mathbf{X}_p) = \gamma^{(l)}\mathbf{X}_p^{(l)} + \beta^{(l)} = \gamma^{(l)}(\mathbf{W}_p^{(l)} * \mathbf{X}_c^{(l)}) + \beta^{(l)}, \tag{3}$$

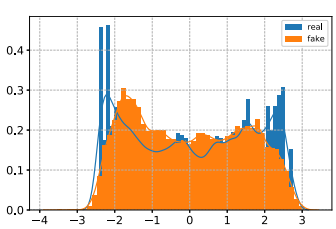
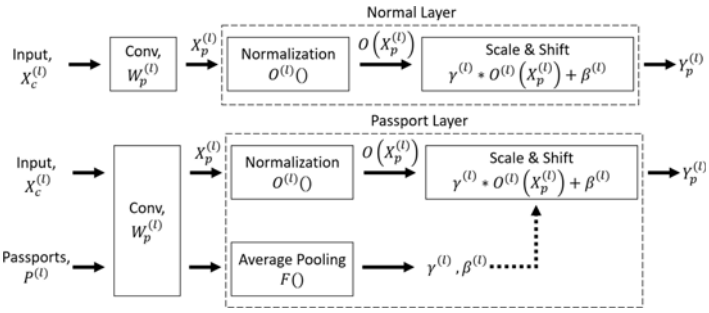
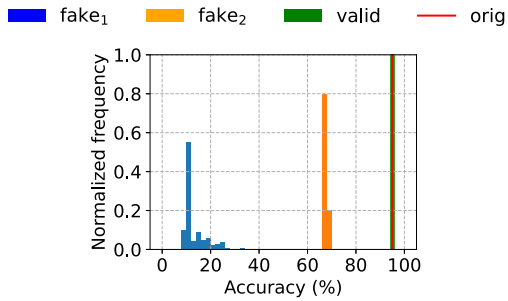


Fig. 5. Distribution of the real $\mathbf{X}_b$ and fake $\mathbf{X}_T$ trigger set images. It shows that the fake trigger set images are hardly distinguishable from the real ones.



(a) Top figure indicates a standard Conv-Norm layer ($\mathbb{N}[\mathbf{W}]$) while bottom figure indicates a passport Conv-Norm layer ($\mathbb{N}[\mathbf{W}, \mathbf{s}_e]$). $\mathbf{P}^{(l)} = \{\mathbf{P}_\gamma^{(l)}, \mathbf{P}_\beta^{(l)}\}$ is the proposed *digital passports* where $\mathcal{F} = \text{Avg}(\mathbf{W}_p^{(l)} * \mathbf{P}_{\gamma,\beta}^{(l)})$ is a passport function to compute the hidden parameters (i.e. $\gamma^{(l)}$ and $\beta^{(l)}$) given in Eq. (3).



(b) A comparison of CIFAR10 classification accuracies given the original DNN, proposed DNN with valid passports, proposed DNN with randomly generated passports ($fake_1$), and proposed DNN with reverse-engineered passports ($fake_2$).

Fig. 6. (a) Passport layer and (b) Classification accuracies modulated by different passports in CIFAR10, e.g., given counterfeit passports, the DNN models performance will be deteriorated instantaneously to fend off illegal usage.

$$\gamma^{(l)} = \text{Avg}(\mathbf{W}_p^{(l)} * \mathbf{P}_\gamma^{(l)}), \qquad \beta^{(l)} = \text{Avg}(\mathbf{W}_p^{(l)} * \mathbf{P}_\beta^{(l)}), \tag{4}$$

in which $*$ denotes the convolution operations, l is the layer number, $\mathbf{X}_p \in \mathbb{R}^{N \times I_C \times I_S \times I_S}$ is the input to the passport layer and $\mathbf{X}_c \in \mathbb{R}^{N \times I_C \times I_S \times I_S}$ is the input (with batch size of N) to the convolution layer. $\text{Avg}()$ is the average pooling function along the dimension of batch size, height and width and $\mathbf{O}()$ is the corresponding linear transformation of outputs, while $\mathbf{P}_\gamma^{(l)}$ and $\mathbf{P}_\beta^{(l)}$ are the passports used to derive scale factor $\gamma^{(l)}$ and bias term $\beta(l)$ respectively. Fig. 6a delineates the architecture of digital passport layers used in a standard Conv-Norm (such as Conv2d-BatchNorm) layer.

Remark: for DNN models trained with passport $\mathbf{s}_e^{(l)} = \{\mathbf{P}_\gamma^{(l)}, \mathbf{P}_\beta^{(l)}\}$, their *inference performances* $\mathcal{M}(\mathbb{N}[\mathbf{W}, \mathbf{s}_e], \mathbf{D}_t, \mathbf{s}_t)$ depend on the running time passports $\mathbf{s}_t$ i.e.,

$$\mathcal{M}(\mathbb{N}[\mathbf{W}, \mathbf{s}_e], \mathbf{D}_t, \mathbf{s}_t) = \begin{cases} \mathcal{M}_{\mathbf{s}_e}, & \text{if } \mathbf{s}_t = \mathbf{s}_e, \\ \mathcal{M}_{\mathbf{s}_t}, & \text{otherwise.} \end{cases} \tag{5}$$

If the genuine passport is not presented $\mathbf{s}_t \neq \mathbf{s}_e$, the running time performance $\mathcal{M}_{\mathbf{s}_t}$ is significantly deteriorated because the corresponding scale factor γ and bias terms β are calculated based on the wrong passports. For instance, as shown in Fig. 6b, a proposed DNN model presented with valid passports (green) will demonstrate almost identical accuracies as to the original DNN model (red). In contrast, the same proposed DNN model presented with counterfeit passports (blue), the accuracy will deteriorate to merely about 10 percent only.

Remark: the gist of the proposed passport layer is to enforce *dependence* between scale factor, bias terms and network weights. As shown by the Proposition 2, it is this dependence that validates the required non-invertibility to defeat ambiguity.

Proposition 2. (Non-invertible process) *propnoninvertclaim A DNN ownership verification scheme $\mathcal{V}$ as in Definition 1 is* non-invertible, *if*

I) the fidelity process outcome $F(\mathbb{N}[\mathbf{W}, \mathbf{T}, \mathbf{s}], \mathbf{D}_t, \mathcal{M}_t, \epsilon_f)$ depends either on the presented signature $\mathbf{s}$ or trigger set $\mathbf{T}$,

II) with forged passport $\mathbf{s}_t \neq \mathbf{s}_e$, the DNN inference performance $\mathcal{M}(\mathbb{N}[\mathbf{W}, \mathbf{s}_e], \mathbf{D}_t, \mathbf{s}_t)$ in (Eq. (5)) deteriorates to the extent that the discrepancy is larger than the prescribed threshold i.e., $|\mathcal{M}_{\mathbf{s}_e} - \mathcal{M}_{\mathbf{s}_t}| > \epsilon_f$.

Proof. Since using forged passports the DNN model performance is significantly deteriorated such that $|\mathcal{M}_{\mathbf{s}_e} - \mathcal{M}_{\mathbf{s}_t}| > \epsilon_f$, it immediately follows, from the definition of invertible verification schemes V (see Definition 1.IV), that the scheme in question is non-invertible. □

4.2 Methods to Generate Passports

Public parameters of a passport protected DNN might be easily plagiarized, then the plagiarizer has to deceive the network with certain passports. The chance of success of such an attacking strategy depends on the odds of correctly guessing the secret passports. Fig. 7 illustrates three different types of passports which have been investigated in our work:

1) *random patterns*, whose elements are independently randomly generated according to the uniform distribution between $[-1, 1]$.
2) one selected image is fed through a trained DNN model with the same architecture, and the corresponding feature maps are collected. Then the selected *image* is used at the input layer and the *corresponding feature* maps are used at other layers as passports. We refer to passports generated as such the *fixed image* passport.
3) a set of N^3 selected *images* are fed to a trained DNN model with the same architecture, and N corresponding *feature maps* are collected at each layer. Among the N options, only one is randomly selected as the passport at each layer. Specifically, for a set of N images being applied to a DNN model with L layers, there are altogether N^L possible combinations of passports that can be generated. This option should provide stronger protection as it is very difficult for the attackers to pick up the correct combinations of passports. We refer to

3. N is number of selected images or feature maps.

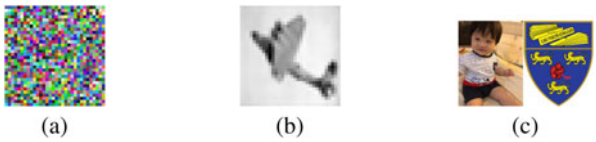


Fig. 7. Example of different types of passports: (a) random patterns, (b) fixed image and (c) candidate images used to generate random shuffled passports (see text in Section 4.2).

passports generated as such the *randomly shuffled* image passports.

Since randomly shuffled passports allow strong protection and flexibility in the passport generation and distribution, we adopt this passport generation method for all the experiments reported in this paper. Specifically, $N = 20$ images are selected and fed to the DNN architectures that are used in our experiments. Feature maps at those corresponding convolution layers are then collected as possible passports. Some example of the features maps selected as the passports at different layers are illustrated in Fig. 8. Note that, in order to enhance the justification of ownership, one can furthermore select either personal identification pictures or organization logos (see Fig. 7c) during the designing of the fixed or random image passports. Also, it must be noted that, using passports as proofs of ownership to stop infringements is the last resort, only if the hidden parameters are illegally disclosed or (partially) recovered. We believe this juridical protection is often not necessary since the proposed technological solution actually provides proactive, rather than reactive, IP protection of deep neural networks.

4.3 Sign of Scale Factors as Signature

During learning the DNN, to further protect the DNN models ownership from insider threat (e.g., a former staff who establish a new start-up business with all the resources copied from originator), one can enforce the scale factor γ to take either positive or negative signs $(+/-)$ as designated, so that it will form a unique signature string. This process is done by adding the following *sign loss* regularization term into the combined loss (Eq. (1))

$$R(\mathbf{P}, \mathbf{B}) = \sum_{i=1}^{C} \max(\alpha - \gamma_i b_i, 0), \tag{6}$$

in which the scale factors $\boldsymbol{\gamma}$ are computed using passports $\mathbf{P}_\gamma$ by Eq. (3), $\mathbf{B} = \{b_1, \ldots, b_C\} \in \{-1, 1\}^C$ consists of the designated binary bits for C convolution kernels, and α is a positive control parameter (0.1 by default unless stated otherwise) to encourage the scale factors have magnitudes greater than α.

It must be highlighted that the inclusion of sign loss (Eq. (6)) enforces the scale factors γ to take either positive or negative values, and the signs enforced in this way remain rather persistent against various adversarial attacks. This feature explains the superior robustness of embedded passports against ambiguity attacks shown in Section 5.2.

Fig. 8. *Randomly shuffled* passports in a 5-layered passport *AlexNet*$_\mathbf{p}$. From left to right: Conv1 to Conv5 layers where the 4 passports in Conv2 to Conv5 corresponding to the first 4 channel of each layer.

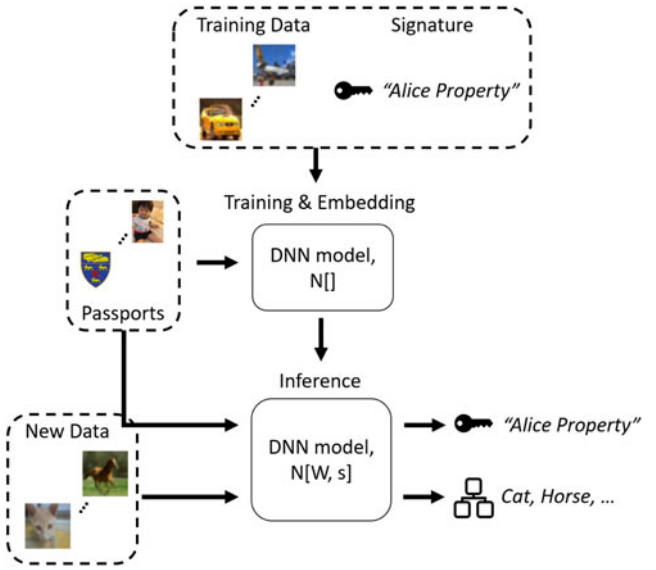


Fig. 9. Ownership verification scheme $\mathcal{V}_1$. $\mathbb{N}[]$ is an untrained network, $\mathbb{N}[\mathbf{W}]$ is a trained network with weights $\mathbf{W}$. $\mathbb{N}[\mathbf{W}, \mathbf{s}]$ is a trained network with weights $\mathbf{W}$ and embedded with signature $\mathbf{s}$.

4.4 Ownership Verification With Passports

Taking advantages of the proposed passport embedding method, we design three ownership verification schemes that are summarized in Figs. 9, 10 and 11 and their respective merits and demerits in Table 3.

$\mathcal{V}_1$: Passport is distributed with the trained DNN model

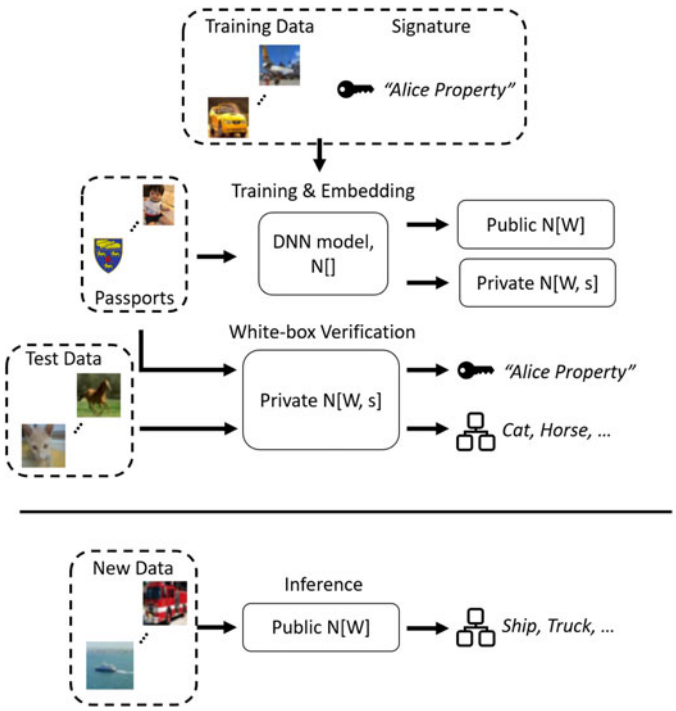


Fig. 10. Ownership verification scheme $\mathcal{V}_2$. Note that the public and private trained network $\mathbb{N}$ share the same weights $\mathbf{W}$.

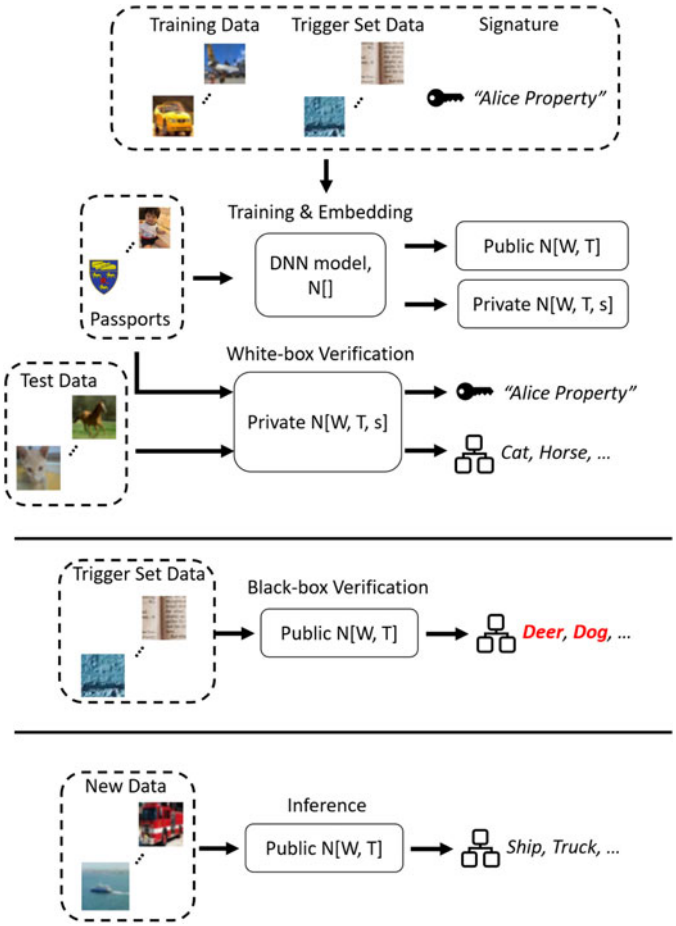


Fig. 11. Ownership verification scheme $\mathcal{V}_3$. $\mathbb{N}[\mathbf{W}, \mathbf{T}]$ is a trained network with weights $\mathbf{W}$ that is embedded with trigger set $\mathbf{T}$. $\mathbb{N}[\mathbf{W}, \mathbf{T}, \mathbf{s}]$ is a trained network with weights $\mathbf{W}$ that is embedded with trigger set $\mathbf{T}$ and signature $\mathbf{s}$. Note that the public and private trained network $\mathbb{N}$ share the same weights $\mathbf{W}$.

Hereby, the *learning* process aims to minimize the combined loss function (Eq. (1)), in which $\lambda_t = 0$ since trigger set images are not used in this scheme and the sign loss (Eq. (6)) is added as the regularization term. The trained DNN model together with the passport $\mathbb{N}[\mathbf{W}, \mathbf{s}]$ are then distributed to legitimate users, who perform network *inferences* with the given *passport* fed to the passport layers as shown in Fig. 6a. The network ownership is automatically verified by the distributed passports. As shown by Table 7 and Fig. 12, this ownership verification is robust to DNN model modifications. Also, as shown in Fig. 16, ambiguity attacks are not able to forge a set of passport and signature that can maintain the DNN inference performance.

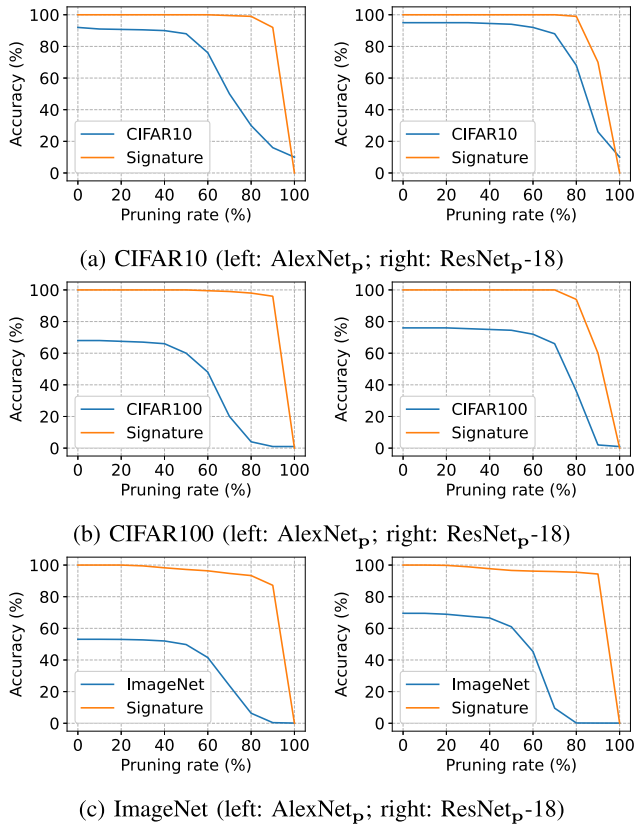


Fig. 12. Removal attack (Model Pruning): Classification accuracy of our passport-based DNN models on CIFAR10, CIFAR100 and ImageNet and signature detection accuracy against different pruning rates. The results are obtained from scheme $\mathcal{V}_1$.

The downside of this scheme is the requirement to use passports during inferencing, which leads to extra computational cost by about 10 percent (see Section 5.5). Also the distribution of passports to the end-users is intrusive and imposes additional responsibility of guarding the passports safely.

$\mathcal{V}_2$: Private passport is embedded but not distributed

Herein, the *learning* process aims to simultaneously achieve *two goals*, of which the first is to minimize the original task loss (e.g., classification accuracy discrepancy) when *no passport* layers included ($\mathbb{N}[\mathbf{W}]$); and the second is to minimize the combined loss function (Eq. (1)) with passports regularization included ($\mathbb{N}[\mathbf{W}, \mathbf{s}]$).

Algorithm-wise, this *multi-task learning* is achieved by alternating between the minimization of these two goals. The successfully trained DNN model $\mathbb{N}[\mathbf{W}]$is then distributed to end-users, who may perform network inference *without the need of passports*. Note that this is possible since

TABLE 3
A Comparison of the Three Passport-Based Ownership Verification Schemes Depicted in Section 4.4

	Passport Needed		Trigger set Needed	Inference	Fidelity Evaluation F		Verification V	
	Training	Inference			if $\mathbf{s}_e = \mathbf{s}_t$	Otherwise	White-box	Black-box
$\mathcal{V}_1$	✓	✓	×	$\mathbb{N}[\mathbf{W}, \mathbf{s}]$	$\leq \epsilon_f$	$> \epsilon_f$	$V(\mathbb{N}[\mathbf{W}, \mathbf{s}])$	-
$\mathcal{V}_2$	✓	×	×	$\mathbb{N}[\mathbf{W}]$	$\leq \epsilon_f$	$> \epsilon_f$	$V(\mathbb{N}[\mathbf{W}, \mathbf{s}])$	-
$\mathcal{V}_3$	✓	×	✓	$\mathbb{N}[\mathbf{W}, \mathbf{T}]$	$\leq \epsilon_f$	$> \epsilon_f$	$V(\mathbb{N}[\mathbf{W}, \mathbf{T}, \mathbf{s}])$	$V(\mathbb{N}[\mathbf{W}, \mathbf{T}])$

See Definition (1) for fidelity evaluation process F and verification process V. White-box verification needs access to DNN weights while black-box verification can be done through remote API call.

passport layers are not included in the distributed networks. The ownership verification will be carried out only upon requested by the law enforcement, by adding the passport layers to the network in question $\mathbb{N}[\mathbf{W}, \mathbf{s}]$and detecting the embedded sign signatures with unyielding the original network inference performances.

Compared with scheme $\mathcal{V}_1$, this scheme is easy to use for end-users since no passport is needed and no extra computational cost is incurred. In the meantime, this ownership verification is robust to both removal attacks and ambiguity attacks. The downside, however, is the requirement to access the DNN weights and to append the passport layers for ownership verification, i.e., the disadvantages of white-box protection mode as discussed in [2]. Therefore, we propose to combine it with trigger-set based verification that will be described next.

$\mathcal{V}_3$: Both the private passport and trigger set are embedded but not distributed

This scheme only differs from scheme $\mathcal{V}_2$ in that, a set of trigger images is embedded in addition to the embedded passports. The advantage of this, as discussed in [2] is to probe and claim ownership of the suspect DNN model ($\mathbb{N}[\mathbf{W}, \mathbf{T}]$) through remote calls of service APIs. This capability allows one, first to claim the ownership in a black-box mode, followed by reassertion of ownership with passport verification in a white box mode.[4] Algorithm-wise, the embedding of trigger set images is jointly achieved in the same minimization process that embeds passports in scheme $\mathcal{V}_2$. Finally, it must be noted that the embedding of passports in both $\mathcal{V}_2$ and $\mathcal{V}_3$ schemes are implemented through *multi-task learning tasks* where we adopted group normalisation [17] instead of batch normalisation [18] that is not applicable due to its dependency on running average of batch-wise training samples.

4.4.1 Algorithms

Pseudo-code of the three verification schemes are illustrated in this section. For reproducibility of this work, we have make publicly available all source codes as well as the training / test datasets that are used in this paper in https://github.com/kamwoh/DeepIPR.

Algorithm 1. Forward Pass of a Passport Layer

procedure forwardX_c, W_p, P_γ, P_β, γ_{publ} β_{publ}, idx
  if $idx = 0$ **then** ▷ Scheme $\mathcal{V}_{23}$
    $X_p \leftarrow W_p * X_c$
    $Y_p \leftarrow \gamma_{publ} * O(X_p) + \beta_{publ}$ ▷ γ_{publ} and β_{publ} is a public parameter
  else ▷ Scheme $\mathcal{V}_1$
    $\gamma \leftarrow Avg(W_p * P_\gamma)$
    $\beta \leftarrow Avg(W_p * P_\beta)$
    $X_p \leftarrow W_p * X_c$
    $Y_p \leftarrow \gamma * O(X_p) + \beta$ ▷ O is a linear transformation such as BatchNorm
  return Y_p

Algorithm 2. Sign Loss

procedure sign loss$B^{(l)}$, $W_p^{(l)}$, $P_\gamma^{(l)}$, α
  $\gamma^{(l)} \leftarrow Avg(W_p^{(l)} * P_\gamma^{(l)})$
  $loss \leftarrow max(\alpha - \gamma^{(l)} * B^{(l)}, 0)$ ▷ α is a positive constant, equals 0.1 as by default
  return loss

Algorithm 3. Signature Detection

procedure signature detectionW_p, P_γ
  $\gamma \leftarrow Avg(W_p * P_\gamma)$
  $signature \leftarrow sign(\gamma)$
  convert *signature* into binary
  decode binarized *signature* into desired format e.g., ascii
  match decoded *signature* with target signature

4.4.2 Multi-Task Learning With Private Passports and/or Trigger Set Images

The multi-task learning algorithms used for embedding passports in schemes $\mathcal{V}_2$ and $\mathcal{V}_3$ are summarized in Algorithm 4.

Algorithm 4. Training Step

initialize a passport model $\mathbb{N}$ with desired number of passport layers, N_{pass}
if enable trigger set **then** ▷ for scheme $\mathcal{V}_3$
  initialize trigger sets **T**
  initialize passport keys **P** in $\mathbb{N}$ using **T**
else
  initialize passport keys **P** in $\mathbb{N}$
encode desired *signature* string **B** into binary to be embedded into signs of γ_p of all passport layers
for number of training iterations **do**
  sample minibatch of m samples **X** $\{\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(m)}\}$ and labels **y** $\{\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(m)}\}$
  if enable backdoor **then**
    sample t samples of **T** and backdoor labels $\mathbf{y}_T$ ▷ $t = 2$, default by [2]
    concatenate **X** with **T**, **y** with $\mathbf{y}_T$
  for idx in 0 1 **do** ▷ Only 1 if scheme $\mathcal{V}_1$
    if $idx = 0$ **then**
      compute cross-entropy loss L_c using **X**, **y** and γ_{publ}
    else
      compute cross-entropy loss L_c using **X** and **y**
      for l in N_{pass} **do**
        compute sign loss $R^{(l)}$ using $s^{(l)}$ and $\gamma_p^{(l)}$
  $R \leftarrow \sum_l^{N_{pass}} R^{(l)}$
  compute combined loss L using L_c and R
  backpropagate using L and update $\mathbb{N}$

It must be noted that the practical choice of formula (Eq. (3)) is inspired by the well-known *Batch Normalization* (BN) layer which essentially applies the channel-wise linear transformation to the inputs [18]. Nevertheless BN is not applicable to multi-task learning tasks because of its dependency on running average of batch-wise training samples. When BN is used for multi-task learning, the test accuracy is significantly reduced even though the training accuracy seems optimized. We therefore adopted *group normalization* (GN) [17] in the baseline DNN model for schemes $\mathcal{V}_2$ and $\mathcal{V}_3$ reported in Table 7.

4. In other words, the black-box verification is to collect enough evidences (e.g., high detection rate) from suspected candidates, then report the suspect to related department with the collected evidences to invoke a more certain – white-box verification.

TABLE 4
(Left:) AlexNet$_\mathbf{p}$ Architecture

Layer Name	Output Size	Weight Shape	Padding
Conv1	32 × 32	64 × 3 × 5 × 5	2
MaxPool2d	16 × 16	2 × 2	-
Conv2	16 × 16	192 × 64 × 5 × 5	2
Maxpool2d	8 × 8	2 × 2	-
Conv3	8 × 8	384 × 192 × 3 × 3	1
Conv4	8 × 8	256 × 384 × 3 × 3	1
Conv5	8 × 8	256 × 256 × 3 × 3	1
MaxPool2d	4 × 4	2 × 2	-
Linear	10	10 × 4096	-

Layer Name	Output Size	Weight Shape	Padding
Conv1	32 × 32	64 × 3 × 3 × 3	1
Conv2_x	32 × 32	$\begin{bmatrix} 64\times64\times3\times3 \\ 64\times64\times3\times3 \end{bmatrix}\times 2$	1
Conv3_x	16 × 16	$\begin{bmatrix} 128\times128\times3\times3 \\ 128\times128\times3\times3 \end{bmatrix}\times 2$	1
Conv4_x	8 × 8	$\begin{bmatrix} 256\times256\times3\times3 \\ 256\times256\times3\times3 \end{bmatrix}\times 2$	1
Conv5_x	4 × 4	$\begin{bmatrix} 512\times512\times3\times3 \\ 512\times512\times3\times3 \end{bmatrix}\times 2$	1
Average pool	1 × 1	4 × 4	-
Linear	10	10 × 512	-

(Right): ResNet$_\mathbf{p}$-18 architecture. We modify the architectures from PyTorch to adapt input size of 32 × 32.

5 Experiment Results

This section illustrates the empirical study of passport-based DNN models, with focuses on *convergence* and *effectiveness* of passport layers. The inference performances of various schemes are also compared in terms of *robustness* to both removal attacks and ambiguity attacks. The network architectures we investigated include the well-known AlexNet and ResNet-18 and in order to avoid confusion to the original AlexNet and ResNet models, we denote AlexNet$_\mathbf{p}$ and ResNet$_\mathbf{p}$-18 as our proposed passport-based DNN models. Three publicly datasets - CIFAR10, CIFAR100 and ImageNet classification tasks are employed because these public datasets allow us to perform extensive tests of the DNN model performances. For CIFAR10 and CIFAR100, we are using input size of 32 × 32 and input size of 224 × 224 for ImageNet. Tables 4 and 5 show the detailed architecture and hyper-parameters for both AlexNet$_\mathbf{p}$ and ResNet$_\mathbf{p}$-18 that employed in all the experiments on CIFAR10 and CIFAR100, while architectures with input size of 224 × 224 are using original AlexNet and ResNet18.[5] Unless stated otherwise, all experiments are repeated 5 times and tested against 50 fake passports to get the mean inference performance.

In the next section, we present several types of results:

- *Removal attacks* as a threat that intent to accomplish removing watermarks from the host image. The techniques include both fine-tuning and model pruning (see Section 5.1).
- *Ambiguity attacks* as a threat that aims to puzzle the detector by generating fake watermark from a watermarked image. The techniques include random attack and reverse-engineering attack (see Section 5.2).
- *Internal attacks* as a threat where (former) staffs exposed/stole all the resources from the originator and it is a special case of ambiguity attacks (see Section 5.3).

5.1 Robustness Against Removal Attacks

5.1.1 Fine-Tuning

In this experiment, we repeatedly trained each model five times with designated scale factor signs embedded into both AlexNet$_\mathbf{p}$ and ResNet$_\mathbf{p}$-18 networks. Table 7 shows that the passport signatures are detected at near to 100 percent accuracy for all the ownership verification schemes in the original task. Even after fine-tuning the proposed DNN models for a new task (e.g., from CIFAR10 to Caltech-101), almost 100 percent detection rates of the embedded passport are still maintained. Although fine-tuning from ImageNet to CIFAR100 or Caltech-101 at worst have only 71.56 percent detection rates for AlexNet$_\mathbf{p}$ scheme $\mathcal{V}_2$ and $\mathcal{V}_3$, but in a ResNet$_\mathbf{p}$-18 is getting almost 100 percent detection rates for all datasets or schemes. Note that a detected signature is claimed only *iff* all the binary bits are exactly matched. We ascribe this superior robustness to the unique controlling nature of the scale factors — in case that a scale factor value is reduced near to zero, the channel output will be virtually zero, thus, its gradient will vanish and lose momentum to move towards to the opposite value. Empirically we have not observed counter-examples against this explanation.[6]

Table 6 shows the trigger set image detection rate before and after fine-tuning. Note that passports are not used in this experiment, therefore, the detection rate of the trigger set labels deteriorated drastically after fine-tuning. Nevertheless, trigger set images can still be used in scheme $\mathcal{V}_3$ to complement the white-box passport-based verification approach.

5.1.2 Model Pruning

The aim of model pruning is to reduce redundant parameters without compromise the performance. Here, we adopt the *class-blind* pruning scheme in [19], and test our proposed DNN models with different pruning rates. Fig. 12 shows that, in general, our proposed DNN models still maintained near to 100 percent detection rate even if 60 percent parameters are pruned, by which performances of original networks about to drop around 3-24 percent for all datasets. Even if we prune 90 percent parameters, the accuracy of our proposed DNN models are still much higher than the accuracy of testing data. As said, we ascribe the robustness against model pruning to the superior persistence of signatures embedded in the scale factor signs (see Section 4.3).

5. https://pytorch.org/docs/stable/torchvision/models.html

6. A rigorous proof of this argument is under investigation and will be reported elsewhere.

TABLE 5
Training Parameters for CIFAR10/100 on AlexNet$_\mathbf{p}$ and ResNet$_\mathbf{p}$-18, Respectively († the Learning Rate is Scheduled as 0.01, 0.001 and 0.0001 Between Epochs [1-100], [101-150] and [151-200] Respectively)

Network Architecture	AlexNet$_\mathbf{p}$	ResNet$_\mathbf{p}$-18
Activation Function	ReLU	
Optimization Method	SGD	
Momentum	0.9	
Learning Rate	0.01, 0.001, 0.0001†	
Learning Rate (ImageNet)	0.1, 0.01, 0.001†	
Batch Size	64	
Batch Size (ImageNet)	256	
Passport Layers	Conv3,4,5	Conv5_x

For ImageNet, learning rates are 0.1, 0.01 and 0.001 between epochs [1-30], [31-60], and [61-90] respectively.

5.2 Resilience Against Ambiguity Attacks

As shown in Fig. 13, the accuracy of our proposed DNN models trained on CIFAR10, CIFAR100 and ImageNet classification task is significantly depending on the presence of either valid or counterfeit passports — the proposed DNN models presented with valid passports demonstrated almost identical accuracies as to the original DNN model. Contrary, the same proposed DNN model presented with invalid passports (in this case of $fake_1$ = random attack) achieved only 10 percent accuracy for CIFAR10 which is merely equivalent to a random guessing. In the case of $fake_2$, we assume that the adversaries have access to the original training dataset, and attempt to reverse-engineer the scale factor and bias term by freezing the trained DNN weights. It is shown that in Fig. 13, reverse-engineering attacks are only able to achieve, for CIFAR10, at best 84 percent accuracy on AlexNet$_\mathbf{p}$ and 70 percent accuracy on ResNet$_\mathbf{p}$-18. While in CIFAR100, for $fake_1$ case, attack on both our proposed DNN models achieved only 1 percent accuracy; for $fake_2$ case, this attack only able to achieve 44 percent accuracy for AlexNet$_\mathbf{p}$ and 35 percent accuracy for ResNet$_\mathbf{p}$-18. Even in a more challenged dataset ImageNet, $fake_1$ attack is still achieved only random guessing result which is 0.1 percent for both models and $fake_2$ attack achieved at best 50 percent for AlexNet$_\mathbf{p}$ and 60 percent for ResNet$_\mathbf{p}$.

TABLE 6
Detection Rate (in %) of the Trigger Set Images (Before and After Fine-Tuning) Used in Scheme $\mathcal{V}_3$ to Complement Passport-Based Verifications

	Trained with		Fine-tuned with		
			CIFAR10	CIFAR100	Caltech-101
AlexNet$_\mathbf{p}$	CIFAR10	100	-	24.67	57.67
	CIFAR100	100	36.00	-	78.67
	ImageNet	100	1.00	1.00	10.00
ResNet$_\mathbf{p}$-18	CIFAR10	100	-	12.50	13.67
	CIFAR100	100	6.33	-	4.67
	ImageNet	100	32.50	14.50	4.00

TABLE 7
Removal Attack (Fine-Tuning): $\mathcal{M}$ Denotes Model Classification Accuracy (in %)

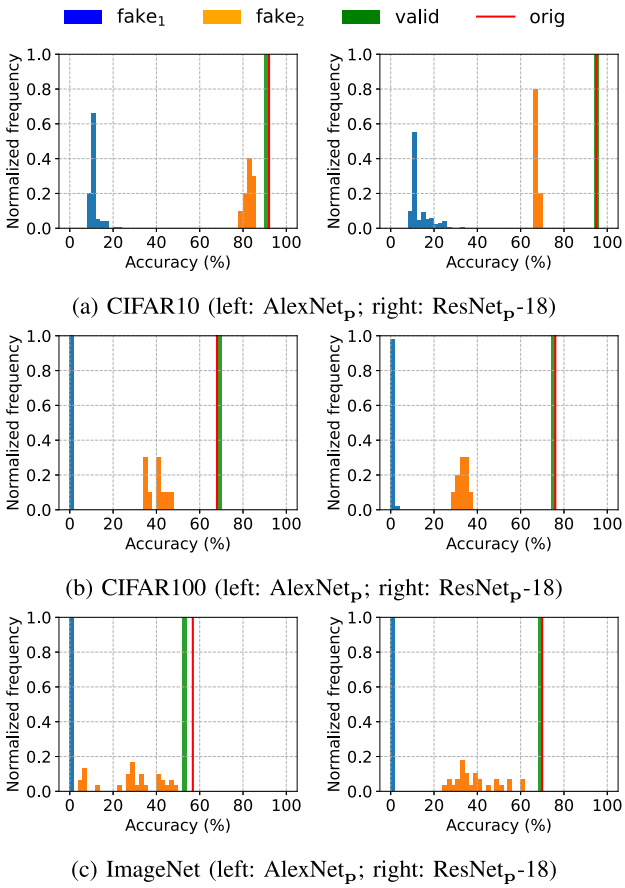
		Trained with	Fine-tuned with	
AlexNet$_\mathbf{p}$		**CIFAR10**	CIFAR100	Caltech-101
$\mathcal{V}_1$	$\mathcal{M}$	90.91 (-0.21)	64.64 (-0.89)	73.03 (-3.30)
	$\mathbb{E}[V]$	100	100	100
$\mathcal{V}_2$	$\mathcal{M}$	89.44 (-1.44)	59.31 (-2.86)	70.87 (-2.41)
	$\mathbb{E}[V]$	100	99.91	100
$\mathcal{V}_3$	$\mathcal{M}$	89.15 (-1.73)	59.41 (-2.76)	71.37 (-1.91)
	$\mathbb{E}[V]$	100	99.96	100
ResNet$_\mathbf{p}$-18		**CIFAR10**	CIFAR100	Caltech-101
$\mathcal{V}_1$	$\mathcal{M}$	94.62 (-0.23)	69.63 (-2.99)	72.13 (-6.85)
	$\mathbb{E}[V]$	100	100	100
$\mathcal{V}_2$	$\mathcal{M}$	93.41 (-0.24)	63.84 (-5.56)	71.07 (-4.01)
	$\mathbb{E}[V]$	100	100	100
$\mathcal{V}_3$	$\mathcal{M}$	93.26 (-0.39)	63.61 (-5.79)	72.00 (-3.08)
	$\mathbb{E}[V]$	100	99.98	99.99
		Trained with	Fine-tuned with	
AlexNet$_\mathbf{p}$		**CIFAR100**	CIFAR10	Caltech-101
$\mathcal{V}_1$	$\mathcal{M}$	68.31 (+0.05)	89.07 (-0.39)	78.83 (-0.83)
	$\mathbb{E}[V]$	100	100	100
$\mathcal{V}_2$	$\mathcal{M}$	64.09 (-1.00)	87.47 (-0.83)	76.31 (-1.77)
	$\mathbb{E}[V]$	100	100	100
$\mathcal{V}_3$	$\mathcal{M}$	63.67 (-1.42)	87.46 (-0.84)	75.89 (-2.19)
	$\mathbb{E}[V]$	100	100	100
ResNet$_\mathbf{p}$-18		**CIFAR100**	CIFAR10	Caltech-101
$\mathcal{V}_1$	$\mathcal{M}$	75.52 (-0.73)	95.28 (+2.06)	79.27 (-3.61)
	$\mathbb{E}[V]$	100	100	99.99
$\mathcal{V}_2$	$\mathcal{M}$	72.15 (+0.09)	90.94 (-0.89)	77.34 (-1.81)
	$\mathbb{E}[V]$	100	100	100
$\mathcal{V}_3$	$\mathcal{M}$	72.10 (+0.04)	91.30 (-0.53)	77.46 (-1.69)
	$\mathbb{E}[V]$	100	100	100
		Trained with	Fine-tuned with	
AlexNet$_\mathbf{p}$		**ImageNet**	CIFAR100	Caltech-101
$\mathcal{V}_1$	$\mathcal{M}$	53.21 (-3.81)	74.27 (-2.10)	85.43 (-2.28)
	$\mathbb{E}[V]$	100	100	100
$\mathcal{V}_2$	$\mathcal{M}$	49.68 (-6.64)	74.02 (-1.30)	86.05 (-1.13)
	$\mathbb{E}[V]$	100	71.56	80.37
$\mathcal{V}_3$	$\mathcal{M}$	49.93 (-6.39)	70.24 (-5.08)	86.21 (-0.97)
	$\mathbb{E}[V]$	100	73.86	79.11
ResNet$_\mathbf{p}$-18		**ImageNet**	CIFAR100	Caltech-101
$\mathcal{V}_1$	$\mathcal{M}$	69.51 (-0.42)	81.29 (-0.16)	92.01 (+0.44)
	$\mathbb{E}[V]$	100	100	100
$\mathcal{V}_2$	$\mathcal{M}$	65.73 (-2.59)	78.68 (-1.06)	91.42 (+1.51)
	$\mathbb{E}[V]$	100	99.77	99.93
$\mathcal{V}_3$	$\mathcal{M}$	65.51 (-2.81)	78.45 (-1.29)	90.56 (+0.65)
	$\mathbb{E}[V]$	100	100	100

Value outside bracket is the model accuracy $\mathcal{M}$ and inside is the accuracy discrepancy (κ) between $\mathcal{M}$ and an unprotected baseline model $\mathcal{M}_t$ i.e., $\kappa = \mathcal{M} - \mathcal{M}_t$ (see fidelity process in Definition 1 and Fig. 2b). $\mathbb{E}[V]$ denotes the mean success rate (in %) of detecting the embedded signature (see verification process in Definition 1 and Fig. 2c). $\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3$ denote three different verification schemes described in Section 4.4.

5.2.1 Random Attacks

The following experiments aim to disclose the dependence of the original task performances with respect to the crucial parameter *scale factors*, and specifically, its positive/negative signs.

In the first experiment, for the passport-embedded DNN models, we assume that adversaries don't have the passport



(a) CIFAR10 (left: AlexNet$_\mathbf{p}$; right: ResNet$_\mathbf{p}$-18)

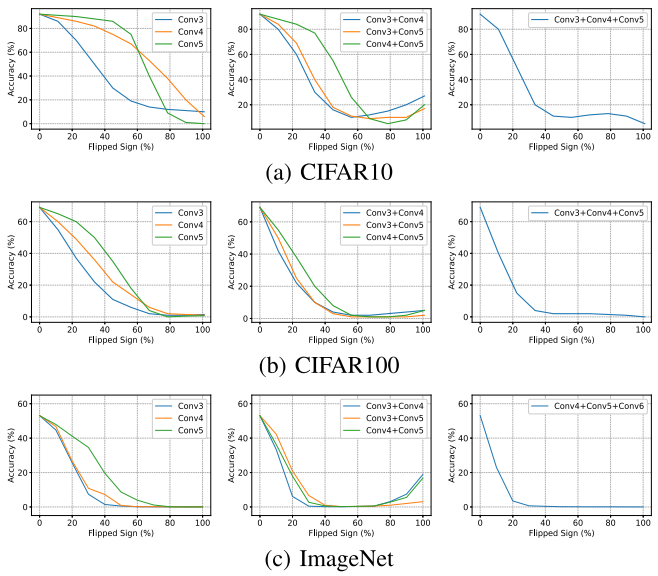
(b) CIFAR100 (left: AlexNet$_\mathbf{p}$; right: ResNet$_\mathbf{p}$-18)

(c) ImageNet (left: AlexNet$_\mathbf{p}$; right: ResNet$_\mathbf{p}$-18)

Fig. 13. Ambiguity attack: Classification accuracy of our passport networks with valid passport, *random attack* ($fake_1$) and *reversed-engineering attack* ($fake_2$) on CIFAR10 (Top), CIFAR100 (Middle) and ImageNet (Bottom). Note that, the accuracies of our passport networks with valid passports and original DNN (without passport) are too close to separate in histograms. The results are obtained using scheme $\mathcal{V}_1$.

(e.g., Fig. 8) and hence cannot use the model properly. Adversaries have to generate their passport by their own. We simulate random attacks by randomly assigning the passport to compute the scale factors and bias. The performances of the model drop significantly to random guessing which are 10-30, 1-3 and 0.1-0.3 percent for CIFAR10, CIFAR100 and ImageNet respectively (see blue bar in Fig. 13).

In the second experiment, we assume that adversaries have the model and also the scale factors computed from the passport. The attacker wishes to remove the embedded signature on the sign of scale factors, and therefore we simulate random attacks by flipping the signs of certain randomly selected scale factors (i.e., to have a certain dissimilarity with original signature) and then measure the performance. It turns out that the final performance are sensitive to the change of signs — majority of the DNN model performances drop significantly as long as more than (at least) 50 percent of scale factors have flipped signs as shown in Figs. 14 and 15, respectively. The deteriorated performances are more pronounced when the passports are embedded in either all the three convolution layers (3-4-5) in AlexNet$_\mathbf{p}$ (right-most column in Fig. 14) or the last blocks in ResNet$_\mathbf{p}$-18 (Fig. 15), whose performances drop to about 10, 1 and 0.1 percent for CIFAR10, CIFAR100 and ImageNet respectively.



(a) CIFAR10

(b) CIFAR100

(c) ImageNet

Fig. 14. Ambiguity attack (Random): It can be seen that the performance of AlexNet$_\mathbf{p}$ $\mathcal{V}_1$ deteriorates with randomly flipped scale factor signs. Left to right: flip one layer, two layers and three layers, respectively. Top row is CIFAR10, middle row is CIFAR100 and bottom row is ImageNet.

5.2.2 *Reversed-Engineering Attacks*

In this experiment, we further assume the adversaries have the access to original training data, knowing there is a signature embedded and thus are able to maximize the original task performance by reverse-engineering scale factors. The trained AlexNet$_\mathbf{p}$/ResNet$_\mathbf{p}$-18 are used for this experiment, and it turns out the best performance the adversary can achieve is no more than 84/70 percent for CIFAR10, 40/38 percent for CIFAR100 and 50/60 percent for ImageNet respectively (see Fig. 16) if the adversary hope not to have exactly the original signature.

Summary. Extensive empirical studies show that it is impossible for adversaries to maintain the original DNN model performances by using fake passports, regardless of the fake passports are either randomly generated or reverse-engineered with the use of original training datasets. Table 8 summarize passport model performances under three different ambiguity attack modes depending on attackers' knowledge of the protection mechanism. It shows that all, unless the original signature is used, the corresponding passport-based DNN models accuracies are deteriorated to various extents. The ambiguous attacks are therefore defeated according to the fidelity evaluation process, $F()$.

It must be noted that even under the most adversary condition, i.e., freezing weights, maximizing the distance from the original passport P, and minimizing the accuracy loss ($fake_3$), attackers are still unable to use new (modified) scale signs without compromising the network accuracies. As shown in Fig. 17, with 10 and 50 percent of the original scale signs are modified,[7] the CIFAR100 classification accuracy drops about 5 and 50 percent, respectively. Based on these empirical studies, we decide to set the threshold ϵ_f in Definition 1 as 3 percent for AlexNet$_\mathbf{p}$

7. In case that the original scale sign remains unchanged, the DNN model ownership can be verified by the pre-defined string of signs.

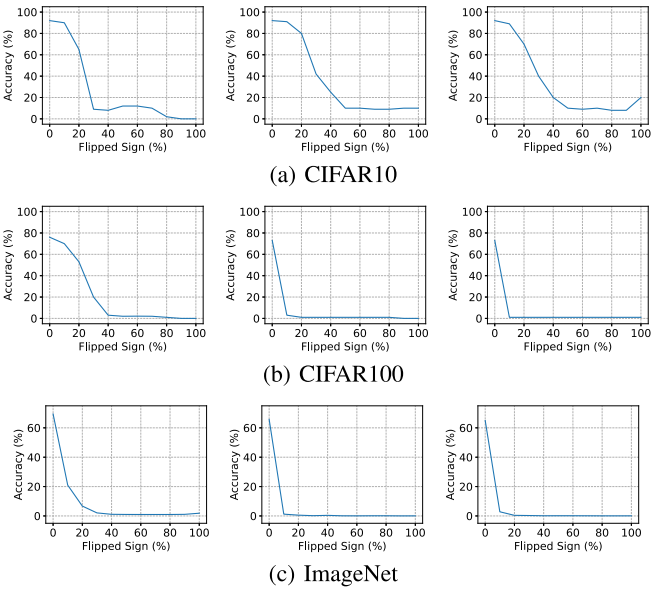


Fig. 15. Ambiguity attack (Random): It can be seen that the performance of ResNet$_\mathbf{p}$-18 deteriorates with randomly flipped scale factor signs. Left to right: Scheme $\mathcal{V}_1$, $\mathcal{V}_2$ and $\mathcal{V}_3$, respectively.

and 10 percent for ResNet$_\mathbf{p}$-18, respectively. By this fidelity evaluation process, any potential ambiguity attacks are effectively defeated.

In summary, extensive empirical studies have shown that it is impossible for adversaries to maintain the original DNN model accuracies by using counterfeit passports, regardless of they are either randomly generated or reverse-engineered with the use of original training datasets. This passport dependent performances play an indispensable role in designing secure ownership verification schemes that are illustrated in Section 4.4.

5.3 Internal Attacks

In this section, we show how the sign $(+/-)$ of scale factor γ can be used to encode a signature s such as ASCII code to defeat internal threat. Table 9 shows an example of the learned scale factors and its respective sign when we embed a signature $s = \{this\ is\ an\ example\ signature\}$ into the Conv5 of AlexNet$_\mathbf{p}$ by using sign loss (Eq. (6)). Note that the maximum size of an embedded signature is depending on the number of the channels in a DNN model. For instance, in this paper, the Conv5 of AlexNet$_\mathbf{p}$ as shown in Table 4 has 256 channels, so the maximum signature capacity is 256bits.

For ownership verification, the embedded signature s can be revealed by decoding the learned sign of scale factors. By using Algorithm 3, we can extract s from model $\mathbb{N}$. We can then decode s into desired format such as ASCII code. For example, in Table 9, every 8bits of the scale factor sign is decoded into ASCII code as follow:

1. {-1,1,1,1,-1,1,-1,-1} $\rightarrow$ 116 $\rightarrow$ t
2. {-1,1,1,-1,1,-1,-1,-1} $\rightarrow$ 104 $\rightarrow$ h
3. {-1,1,1,-1,1,-1,-1,-1} $\rightarrow$ 105 $\rightarrow$ i
4. {-1,1,1,1,-1,-1,1,1} $\rightarrow$ 115 $\rightarrow$ s
5. {-1,1,1,-1,1,-1,-1,1} $\rightarrow$ 105 $\rightarrow$ i
6. {-1,1,1,1,-1,-1,1,1} $\rightarrow$ 115 $\rightarrow$ s

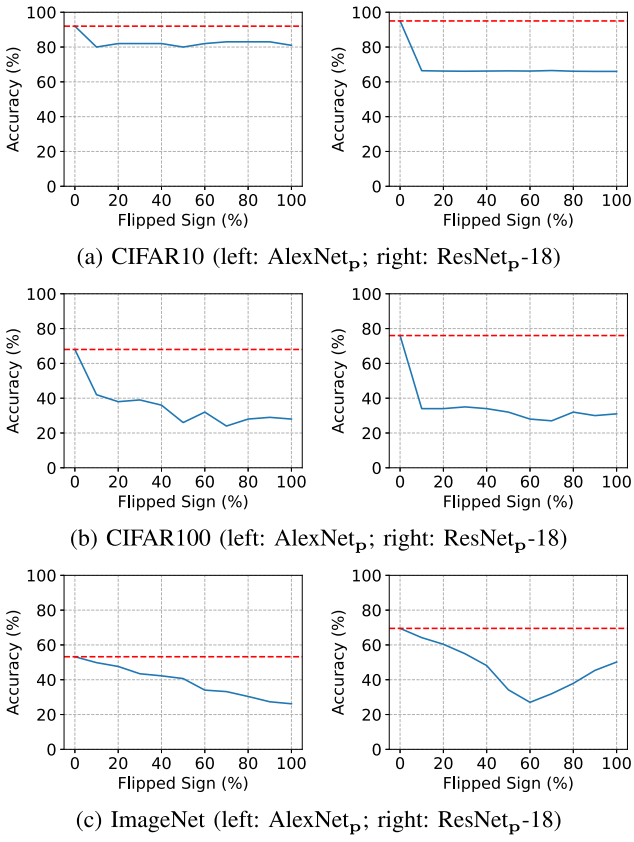


Fig. 16. Ambiguity attack (Reverse Engineer): Performance of (a) CIFAR10, (b) CIFAR100 and (c) ImageNet when adversaries try to forge a new signature by a certain % of dissimilarity with the original signature. The results are obtained from scheme $\mathcal{V}_1$.

Note that, in this proposed method, similar character (e.g., {i} and {s}) appears in different position of a string will have different scale factors. Table 10 shows a comparison result when a correct signature, partial correct signature or total wrong signature is used in CIFAR10 classification task with AlexNet$_\mathbf{p}$. It is shown that when a correct signature is used (i.e this is an example signature), the classification accuracy reached 90.89 percent, while for a partial correct signature, the performance is dropped to 82.23 percent, and a totally wrong signature will obtain a meaningless accuracy (11.44 percent). Based on the

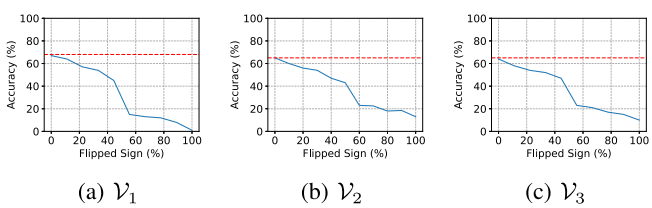


Fig. 17. Ambiguity attack: Classification accuracy on CIFAR100 under insider threat ($fake_3$) on three verification schemes. It is shown that when a correct signature is used, the classification accuracy is intact, while for a partial correct signature (sign scales are modified around 10 percent), the performance will immediately drop around 5 percent, and a totally wrong signature will obtain a meaningless accuracy (1-10 percent). Based on the threshold $\leq \epsilon_f$ = 3% for AlexNet$_\mathbf{p}$ and by the fidelity evaluation process F, any potential ambiguity attacks (even with partially correct signature) are effectively defeated.

TABLE 8
Summary of Overall Passport Model Performances Under Three Different Ambiguity Attack Modes, $fake$

	Attackers have access to	Ambiguous passport forging methods	Verification V	Fidelity Evaluation F	Model Performance
$fake_1$	W	Random passport P_r	*True*	*False*	Large accuracy dropping
$fake_2$	W, $\{D_r;D_t\}$	Reverse engineer passport P_e	*True*	*False*	Moderate accuracy dropping
$fake_3$	$\{D_r;D_t\}$, $\{P, B\}$	Reverse engineer passport $\{P_e;B_e\}$ by exploiting original passport P as initialization of P_e	*True*	*False*	Accuracy dropping depending on percentage of correct signature signs (see Fig. 17 and caption)

Noted that when the attacker using $B_e = B$ in $fake_3$, even though the attacker passed the fidelity evaluation F, the ambiguity is resolved by the original signature.

TABLE 9
Sample of the Learned Scale Factor γ and Respective Signs (+/-) From the 48 Out of 256 Channels From Conv5 of AlexNet$_\mathbf{p}$ When we Embed Signature s = {this} and {is}

t			h			i			s			i			s		
γ	+/-	bit	γ	+/-	bit	γ	+/-	bit	γ	+/-	bit	γ	+/-	bit	γ	+/-	bit
-0.11	-	0	-0.17	-	0	-0.10	-	0	-0.20	-	0	-0.17	-	0	-0.23	-	0
0.23	+	1	0.33	+	1	0.39	+	1	0.27	+	1	0.17	+	1	0.29	+	1
0.25	+	1	0.19	+	1	0.43	+	1	0.16	+	1	0.46	+	1	0.22	+	1
0.49	+	1	-0.12	-	0	-0.11	-	0	0.25	+	1	-0.27	-	0	0.19	+	1
-0.10	-	0	0.16	+	1	0.44	+	1	-0.13	-	0	0.38	+	1	-0.11	-	0
0.39	+	1	-0.18	-	0	-0.15	-	0	-0.19	-	0	-0.18	-	0	-0.15	-	0
-0.12	-	0	-0.27	-	0	-0.11	-	0	0.23	+	1	-0.11	-	0	0.23	+	1
-0.34	-	0	-0.20	-	0	0.19	+	1	0.20	+	1	0.16	+	1	0.31	+	1

threshold $\epsilon_f = 3\%$ for AlexNet$_\mathbf{p}$ and by the fidelity evaluation process, any potential ambiguity attacks (even with partially correct signature) are effectively defeated.

5.4 Ablation Study

5.4.1 Convergence

In this section, we showed that the introduction of the proposed passport layers does not hinder the convergence of DNN learning process. As shown in Fig. 18, we observe that the test accuracies converge in synchronization with the network weights, and computed linear transformation parameters γ and β which all stagnate in the later learning phase when the learning rate is reduced from 0.01 to 0.0001.

5.4.2 Effectiveness

With the introduction of the passport layers, we essentially separate the DNN parameters into two types: the *public* convolution layer parameters $\mathbf{W}$ and the *hidden*[8] passport layer - i.e., scale factor γ and bias terms β (see Eq. (5)). The learning of each of these parameter types are different too. On one hand, the distribution of the convolution layer weights seems identical to that of the original DNN without passport layers (Fig. 19a). However, we must emphasize that information about the passports are embedded into weights $\mathbf{W}$ in the sense that following constraints are enforced once the learning is done

$$\mathrm{Avg}(\mathbf{W}_p^{(l)} * \mathbf{P}_\gamma^{(l)}) = c_\gamma^{(l)}, \qquad \mathrm{Avg}(\mathbf{W}_p^{(l)} * \mathbf{P}_\beta^{(l)}) = c_\beta^{(l)}, \tag{7}$$

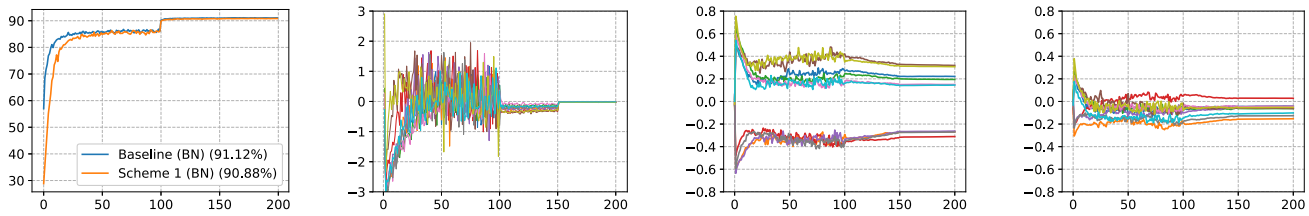
where $c_\gamma^{(l)}, c_\beta^{(l)}$ are two constants of converged parameters $\gamma^{(l)}, \beta^{(l)}$.

On the other hand, the distribution of the hidden parameters are affected by the adoption of sign loss (Eq. (6)). Clearly the scale factors are enforced to take either positive or negative values far from zero (Fig. 19b). We also observe that the sign of scale factors remain rather persistent against various adversarial attacks. An

TABLE 10
A Comparison of the Accuracy of AlexNet(s) in CIFAR10 Classification Task When a Correct (Top), Partially Correct (Middle) or Totally Wrong (Bottom) Signature is Used

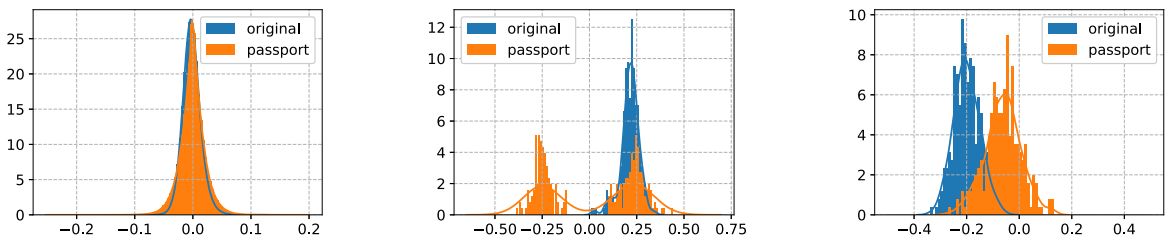
	Signature s	Accuracy (%)
AlexNet (baseline)	-	91.12
	$this\ is\ an\ example\ signature$	90.89
AlexNet$_\mathbf{p}$	$thhs\ iB\ an\ xxxpxX\vert\ sigjature$	82.83
	$qpCA2J^{O}Ec\Delta o * 1ay$	11.44

8. In this work, traditional hidden layer parameters are considered as public parameters.



(a) Test accuracies (b) Weights update (c) Scale factors (d) Bias terms

Fig. 18. (a) Convergences of test accuracies, (b) weight updates, (c) scale factors, and (d) bias terms of first 10 channels in Conv4 of AlexNet$_p$ $\mathcal{V}_1$. x-axis: training epochs; y-axis: see captions of subfigures.



(a) Weight distribution (b) Scale factor distribution (c) Bias term distribution

Fig. 19. Comparison of the distributions of (a) network weights, (b) scale factors, and (c) bias terms between the original and passport DNN (Conv4 of AlexNet$_p$ $\mathcal{V}_1$).

additional benefit of enforcing non-zero magnitudes of scale factors is to ensure the non-zero channel outputs and slightly improve the performances. Correspondingly the distribution of bias terms becomes more balanced with the sign loss regularization (Eq. (6)) included, whereas the original bias terms are mainly negative valued (Fig. 19c).

5.5 Network Complexity

Table 11 shows the training and inference time of each scheme on AlexNet$_p$ and ResNet$_p$-18, respectively using one NVIDIA Titan V. In both of the proposed DNN architectures, the inference time of the baseline, scheme $\mathcal{V}_2$, scheme $\mathcal{V}_3$ are almost the same as to the execution time because all of them did not use passport to calculate γ and β. However, scheme $\mathcal{V}_1$ is slightly slower (about 10 percent) compared to the baseline because of the extra computational cost of γ and β calculation from the passport. Training time of scheme $\mathcal{V}_1$, scheme $\mathcal{V}_2$ and scheme $\mathcal{V}_3$ are slower than the baseline about 18%(ResNet$_p$-18)/27%(AlexNet$_p$), 116%(ResNet$_p$-18)/125% and 127%(ResNet$_p$-18)/153%, respectively. Scheme $\mathcal{V}_2$ and scheme $\mathcal{V}_3$ are slower about 2x than scheme $\mathcal{V}_1$ due to the multi task training scheme. Nonetheless, we tested a larger network (i.e., ResNet$_p$-50) and its training time increases 10, 182 and 191 percent respectively for $\mathcal{V}_1$, $\mathcal{V}_2$ and $\mathcal{V}_3$ schemes. This increase is consistent with those smaller models i.e., AlexNet$_p$ and ResNet$_p$-18.

Table 12 summarizes the complexity of passport networks in various schemes. We believe that it is the computational cost at the inference stage that should be reduced, since network inference is going to be performed frequently by the end users. While extra costs at the training and verification stages, on the other hand, are not prohibitive since they are performed by the network owners, with the motivation to protect the DNN model ownerships.

6 Discussions and Conclusions

Considering billions of dollars have been invested by giant and start-up companies to explore new DNN models virtually every second, we believe it is imperative to protect these inventions from being stolen. While ownership of DNN models might be resolved by registering the models

TABLE 11
Training (T) and Inference (I) Time of Each Scheme on AlexNet$_p$ (a) and ResNet$_p$-18 (b) Using One *NVIDIA Titan V*

	CIFAR10	
	T	I
AlexNet (Baseline)	8.445	0.834
AlexNet$_p$ $\mathcal{V}_1$	10.745	0.912
AlexNet$_p$ $\mathcal{V}_2$	19.010	0.830
AlexNet$_p$ $\mathcal{V}_3$	21.372	0.881
(a) AlexNet$_p$		

	CIFAR10	
	T	I
ResNet (Baseline)	31.09	1.71
ResNet$_p$-18 $\mathcal{V}_1$	36.67	1.94
ResNet$_p$-18 $\mathcal{V}_2$	67.21	1.87
ResNet$_p$-18 $\mathcal{V}_3$	70.69	1.88
(b) ResNet$_p$-18		

The values are in seconds/epoch.

TABLE 12
Summary of Our Proposed Passport Networks Complexity for $\mathcal{V}_1$, $\mathcal{V}_2$ and $\mathcal{V}_3$ Schemes

		Schemes		
		$\mathcal{V}_1$	$\mathcal{V}_2$	$\mathcal{V}_3$
Training	Passport layers	✓	✓	✓
	Passport needed	✓	✓	✓
	Trigger set needed	×	×	✓
	Training time	1.15x-1.30x	2x-2.25x	2x-2.5x
Inferencing	Passport needed	✓	×	×
	Inference time	1.1x	1	1
Verification	Black-box (by trigger)	×	×	✓
	White-box (by passport)	✓ (verify on the go)	✓ (passport layers needed)	✓ (passport layers needed)

with a centralized authority, it has been recognized that these regulations are inadequate and technical solutions are urgently needed to support the law enforcement and juridical protections. It is this motivation that highlights the unique contribution of the proposed method in unambiguous verification of DNN models ownerships.

Methodology-wise, our empirical studies re-asserted that over-parameterized DNN models can successfully learn multiple tasks with arbitrarily assigned labels and/or constraints. While this assertion has been theoretically proved [20] and empirically investigated from the perspective of network generalization [21], its implications to network security in general remain to be explored. We believe the proposed modulation of DNN performance based on the presented passports will play an indispensable role in bringing DNN behaviours under control against adversarial attacks, as it has been demonstrated for DNN ownership verifications.

ACKNOWLEDGMENTS

The authors would like to thank support of NVIDIA Corporation with the donation of Titan V GPU used for this research. This work was supported in part by the National Key Research and Development Program of China under Grant 2020YFB1805501 and in part by Fundamental Research Grant Scheme (FRGS) MoHE under Grant FP021-2018A from the Ministry of Education Malaysia.

REFERENCES

[1] Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh, "Embedding watermarks into deep neural networks," in *Proc. ACM Int. Conf. Multimedia Retrieval*, 2017, pp. 269–277.

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

[3] H. Chen, B. Darvish Rohani , and F. Koushanfar, "Deepmarks: A digital fingerprinting framework for deep neural networks," 2018, *arXiv: 1804.03648*.

[4] J. Zhang *et al.*, "Protecting intellectual property of deep neural networks with watermarking," in *Proc. Asia Conf. Comput. Commun. Secur.*, 2018, pp. 159–172.

[5] B. Darvish Rouhani , H. Chen, and F. Koushanfar, "Deepsigns: An end-to-end watermarking framework for ownership protection of deep neural networks," in *Proc. 24th Int. Conf. Architectural Support Program. Lang. Oper. Syst.*, 2019, pp. 485–497.

[6] L. Fan, K. W. Ng, and C. S. Chan, "Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 4714–4723.

[7] E. Le Merrer, P. Perez, and G. Trédan, "Adversarial frontier stitching for remote neural network watermarking," *2017, arXiv: 1711.01894*.

[8] G. Jia and M. Potkonjak, "Watermarking deep neural networks for embedded systems," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, 2018, pp. 1–8.

[9] J. Zhang, D. Chen, J. Liao, W. Zhang, G. Hua, and N. Yu, "Passport-aware normalization for deep model protection," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 22619–22628.

[10] X. Xu, Y. Li, and C. Yuan, "Identity Bracelets" for deep neural networks," *IEEE Access*, vol. 8, pp. 102065–102074, 2020.

[11] A. Pyone, M. Maung, and H. Kiya, "Training DNN model with secret key for model protection," in *Proc. IEEE 9th Glob. Conf. Consum. Electron.*, 2020, pp. 818–821.

[12] N. M. Jebreel, J. Domingo-Ferrer , D. SÃ*a*nchez, and A. Blanco-Justicia , "KeyNet: An asymmetric key-style framework for watermarking deep learning models," *Appl. Sci.*, vol. 11, no. 3, 2021, Art. no. 999.

[13] F. Boenisch, "A survey on model watermarking neural networks," 2020, *arXiv:2009.12153*.

[14] Q. Li and E.-C. Chang, "Zero-knowledge watermark detection resistant to ambiguity attacks," in *Proc. 8th Workshop Multimedia Secur.*, 2006, pp. 158–163.

[15] H. T. Sencar and N. D. Memon, "Combatting ambiguity attacks via selective detection of embedded watermarks," *IEEE Trans. Inf. Forensics Secur.*, vol. 2, no. 4, pp. 664–682, Dec. 2007.

[16] S. Craver, N. Memon, B. Yeo, and M. M. Yeung, "Resolving rightful ownerships with invisible watermarking techniques: Limitations, attacks, and implications," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 4, pp. 573–586, May 1998.

[17] Y. Wu and K. He, "Group normalization," *Int. J. Comput. Vis.*, vol. 128, no. 3, pp. 742–755, 2020.

[18] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[19] A. See, M.-T. Luong, and C. D. Manning, "Compression of neural machine translation models via pruning," in *Proc. 20th SIGNLL Conf. Comput. Natural Lang. Learn.*, 2016, pp. 291–301.

[20] Z. Allen-Zhu , Y. Li, and Z. Song, "A convergence theory for deep learning via over-parameterization," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 242–452.

[21] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," 2016, *arXiv:1611.03530*.

Lixin Fan (Senior Member, IEEE) is currently the principal scientist of artificial intelligence with WeBank. He was the inventor of more than 100 patents filed in USA, Europe, and China. He was with Nokia Research Center and Xerox Research Center Europe. His research interests include the well-known bag of keypoints image classification method, machine learning and deep learning, computer vision and pattern recognition, and image and video processing. He has participated in the NIPS/NeurIPS, ICML, CVPR, ICCV, ECCV, IJCAI and other top artificial intelligence conferences for a long time. He was the area chair of the AAAI and organized workshops in various technical fields. He is currently the chairman of the IEEE P2894 Explainable Artificial Intelligence Standard Working Group.

Kam Woh Ng (Member, IEEE) received the BCS degree from the Faculty of Computer Science and Information Technology, University of Malaya, Malaysia, in 2019. He is currently working toward the PhD degree with the University of Surrey, U.K., under the supervision of Prof. Tao Xiang and Dr. Yizhe Song. He was an AI researcher from WeBank, China, and a lab member of the Center of Image and Signal Processing, Universiti Malaya, Malaysia. His research interests include deep learning, computer vision, representation learning, and their applications.

Chee Seng Chan (Senior Member, IEEE) is currently an associate professor with the Faculty of Computer Science and Information Technology, Universiti Malaya, Kuala Lumpur, Malaysia. From 2020 to 2022, he is seconded to the Ministry of Science and Technology and Innovation, Malaysia, as an under-secretary. His research interests include computer vision and machine learning with focus on scene understanding, interplay between language and vision, and generating sentential descriptions about complex scenes. He was the founding chair of the IEEE Computational Intelligence Society Malaysia Chapter, the organising chair for the Asian Conference on Pattern Recognition in 2015, the general chair for the IEEE Workshop on Multimedia Signal Processing in 2019, and the *IEEE Visual Communications and Image Processing* in 2013. He is a currently a chartered engineer registered under Engineering Council, U.K. He was the recipient of several notable awards, including the Young Scientist Network-Academy of Sciences Malaysia in 2015 and Hitachi Research Fellowship in 2013.

Qiang Yang (Fellow, IEEE) is currently a chief artificial intelligence officer with WeBank and the chair professor with CSE Department, Hong Kong University of Science and Technology. His research interests include transfer learning and federated learning. He is the conference chair of AAAI-21, the president of the Hong Kong Society of Artificial Intelligence and Robotics, the President of Investment Technology League, and the former president of IJCAI from 2017 to 2019. He is the founding editor-in-chief of the *IEEE Transactions on Big Data* and the *ACM Transactions on Intelligent Systems and Technology*. He is a fellow of the AAAI, the ACM, the IEEE, and the AAAS.