Trapdoor Normalization with Irreversible Ownership Verification

Hanwen Liu [1] **Zhenyu Weng** [2] **Yuesheng Zhu** [2] **Yadong Mu** [1]

Abstract

This paper introduces a deep model watermark with an irreversible ownership verification scheme: **T**rap**d**oor **N**ormalization (`TdN`), inspired by the trapdoor function in traditional cryptography. To protect intellectual property within deep models, the proposed method is able to embed ownership information into normalization layers during training. We argue and empirically validate that relevant methods are vulnerable to ambiguity attacks, where the forged watermarks can cast ambiguity over the ownership verification. The primary trait that distinguishes this work from previous ones, is its design of a *bidirectional* connection between watermarks and deep models. Thereby, `TdN` enables an irreversible ownership verification scheme that is difficult for the adversary to compromise. In this way, the proposed `TdN` can effectively defeat ambiguity attacks. Extensive experiments demonstrate that the proposed method is not only superior to previous state-of-the-art methods in robustness, but also has better efficiency.

1. Introduction

With the increasing demand for deep learning applications, deep neural networks are becoming increasingly larger and more complex (Brown et al., 2020; Ramesh et al., 2022; Rombach et al., 2022; Saharia et al., 2022), which spurs more investment in computing resources and data collection. On the one hand, deep models are regarded as a form of valuable assets. On the other hand, when leaked (or stolen) and under some unexpected malicious use or abuse, deep models may cause serious privacy and security issues (Krishna et al., 2020; Choquette-Choo et al., 2021; Yu et al., 2022). Thus an emerging primary concern in its modern development relates to the need for ownership protection.

Over recent years, deep model watermarks (Liu et al., 2021a) serve as a promising way for model ownership protection. By embedding the specific signature into deep models in the training phase, ownership can be verified when the signature is detected from the suspicious model. Pioneer approaches tried to hide ownership information in the model weights (Uchida et al., 2017) or prediction results (Adi et al., 2018). However, these methods are proved to be vulnerable to the so-called *ambiguity attack*, which is the central concern of this research work. The term refers to the scenario in which the adversary deliberately forges watermarks using adversarial knowledge, thereby making the ownership doubtful since multiple watermarks can now pass the verification. To defeat ambiguity attacks, passport-based methods (Fan et al., 2019; 2022; Zhang et al., 2020) build a correlation between model classification fidelity and the present watermark (*i.e.*, a passport image in this case). Specifically, the original passport is transformed as a part of model parameters that are trained alternatively with other model parameters, to deal with verification and deployment scenarios respectively. During verification, the parameters of the corresponding position will be replaced by those generated by the original passport. Once the adversary forges a fake passport, the ownership cannot be falsely claimed since the fake passport will result in deteriorated model fidelity due to incorrect parameters.

We argue that current passport-based methods are not robust to ambiguity attacks if the adversary has access to the original passport, where the adversary can forge a fake passport using the knowledge about the original one to pass the verification. In the potentially huge parameter space, the adversary can maximize the discrimination between the forged passport and the original one, while keeping the original model fidelity. Optimized from the original passport, the forged passport will look different and generate a set of alternative parameters to pass the fidelity evaluation. For ambiguity robustness, we find that it is insufficient to only build the correlation between the model and the passport, only from a feed-forward direction, as previous works did (Fan et al., 2019; Zhang et al., 2020). To tackle it, this paper proposes a conceptually simple yet effective watermarking method, namely **T**rap**d**oor **N**ormalization (`TdN`), for deep model ownership protection. The fundamental spirit of `TdN` is inspired by the trapdoor function in traditional cryptography (Diffie & Hellman, 1976), where it is extremely difficult

[1]Wangxuan Institute of Computer Technology, Peking University [2]School of Electronic and Computer Engineering, Peking University. Correspondence to: Yadong Mu <myd@pku.edu.cn>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

to find the inverse of the embedded signature (*e.g.*, an image that matches the signature). By binding watermarks with models in a bidirectional manner, even if the adversary obtains the original passport and forges a new one, and passes the fidelity evaluation, the forged passport cannot be verified, because the signature in the watermarked model is the hash fingerprint of the original passport.

The contributions of the paper can be summarized as follows: a) We propose the first normalization-form watermark that is robust against ambiguity attacks with oracle passports, with theoretical analyses; b) The proposed trapdoor in deep models enables an irreversible ownership verification scheme, with only negligible training overheads compared with previous methods; c) Experiments prove that `TdN` outperforms previous methods in terms of robustness and efficiency.

2. Background

Deep model watermarks (Liu et al., 2021a; Bansal et al., 2022; Yu et al., 2022) are a means of ownership protection that verifies whether a trained model was from its authorized owner. To protect digital rights and legitimate interests, ownership information is embedded into models during the training stage. When verifying suspicious models, previous methods used specific matrices (Uchida et al., 2017; Rouhani et al., 2019) or pre-defined images (Adi et al., 2018; Guo & Potkonjak, 2018; Zhang et al., 2018) to check the potential ownership information in the trained models.

Normalization layers (Ba et al., 2016; Ulyanov et al., 2016) have been widely applied in modern deep neural networks (Dosovitskiy et al., 2021; Liu et al., 2021b; Dai et al., 2021) to ease the training difficulty. To normalize intermediate features, Batch Norm (Ioffe & Szegedy, 2015) utilized statistics from the mini-batch samples. Likewise, Group Norm (Wu & He, 2018) aimed to normalize features by dividing them into groups. For the purpose of ownership protection, passport-based methods emerge. By replacing normalization layers, Fan et al. (2019; 2022) proposed Passport Layer to manipulate the classifier performance based on the correctness of present passports. Since forged passports would result in poor model performance, Fan et al. (2019) argued that Passport Layer is robust against ambiguity attacks, whereas previous methods (Uchida et al., 2017; Adi et al., 2018) cannot resist such attacks. To alleviate model fidelity drops caused by Passport Layer, Zhang et al. (2020) used an extra passport-aware branch for ownership protection (*i.e.*, Passport Norm). Given a passport image p, these passport-based methods share a similar formulation:

$$\begin{aligned} \text{Normaliztion:} \quad & \hat{x} = \frac{1}{\sigma_x}(x - \mu_x), \\ \text{Affine:} \quad & z = \gamma_p \hat{x} + \beta_p, \end{aligned} \tag{1}$$

where feature x is the output of a precedent layer. In Eq. (1), the way to compute the mean μ_x and the standard deviation σ_x of x is different, which depends on the particular normalization methods (*e.g.*, according to the mini-batch in Batch Norm). The only difference made by passport-based methods is in the affine transformation, where γ_p and β_p are decided by the present passport p. Fan et al. (2019) used an average pooling layer to compute γ_p and β_p based on p, while Zhang et al. (2020) implemented them using a block similar to SE block (Hu et al., 2018). Ideally, any image that is not p, will lead to incorrect γ_p and β_p, thus making deep models suffer from steep accuracy drops.

However, to defeat ambiguity attacks, it is not enough to bind the passport with forward propagation to prevent passport forgery. Since the entire parameter space is huge, it is possible to forge a passport that can generate similar γ_p and β_p, yet looks different from the original one in the aspect of human vision. In the following sections, we prove that it is not trustworthy to rely only on fidelity evaluations to resist ambiguity attacks. Unlike previous passport-based methods, our proposed method not only binds the passport forward based on forward propagation, but also binds the fingerprint of the passport, in a backward manner.

3. Trapdoor Normalization (`TdN`)

Revisiting ambiguity attacks. Previous passport-based methods (Fan et al., 2019; Zhang et al., 2020) proposed the fidelity evaluation to defeat ambiguity attacks, where models verified with randomly forged passports will deteriorate in terms of verification accuracy (*i.e.*, based on incorrect γ_p and β_p there is a significant drop in accuracy during verification). However, we find previous methods are vulnerable to ambiguity attacks when the original passport is available (*i.e.*, the oracle passport scenario), as in Definition 3.1.

Definition 3.1 (Ambiguity attacks with oracle passports)**.** Given the deep neural network $F_\theta(\cdot)$ with parameters θ, the loss function $L_\theta(\cdot)$ *w.r.t.* the model fidelity, and the watermark embedding term E_θ. Let $\mathcal{D}$ and $\delta(p)$ represent the training dataset and a feasible perturbation region *w.r.t* the original passport p, respectively. Ambiguity attacks can be formulated as a bi-level optimization problem:

$$\begin{aligned} \min \quad & L_\theta(\mathcal{D}, \tilde{p}) + E_\theta \\ s.t. \quad & \tilde{p} = \arg\max_{\tilde{p} \in \delta(p)} \mathrm{DST}(\tilde{p}, p), \end{aligned} \tag{2}$$

where $\mathrm{DST}(\cdot)$ is a distance function to measure differences between the forged passport $\tilde{p}$ and the original passport p. The aim is to forge a *visually different* passport that can pass the fidelity evaluation *w.r.t.* $L_\theta(\cdot)$ while keeping valid ownership *w.r.t.* E_θ, without modifying the model.

In fact, it is ordinary that the original passport is exposed to the adversary: a) The adversary could be a former employee and leave the company with the original passport illegally. b) Once public verification happens, there is no chance for a second verification since the original passport is public and known to everyone including the adversary. Unfortunately, as empirically proved later in Table 3, even with a common cosine similarity function as $\mathrm{DST}(\cdot)$ in Definition 3.1, existing passport-based methods are not robust against ambiguity attacks with oracle passports. To this end, we introduce irreversible ownership verification in the proposed `TdN` to resist ambiguity attacks.

Threat model. We assume that the adversary is capable of obtaining deep models illegally. Besides, the adversary intends to modify the trained models to remove the ownership information, or cast doubts on the verification process to make the ownership ambiguous. The following restrictions on the adversary are considered for the own interests of the adversary: a) The adversary cannot attack the embedded watermark by significantly impairing the model performance, which will make the model less valuable. b) The adversary only possesses limited training data or computational resources, since it is meaningless to steal a trained model if the adversary could train a model from scratch.

The goal of watermarking deep models is to embed the identification of the owner (*i.e.*, ownership information) into deep neural networks. The requirements of embedding watermarks into deep models are two-fold: a) The watermark is supposed to have minimal impact on the model performance, and it should be as difficult as possible for the adversary to detect the presence of embedded watermarks. b) The watermark should be robust against various attacks, including removal attacks by adding perturbations on the model parameters (*e.g.*, fine-tuning and pruning), and ambiguity attacks by forging watermarks to falsely claim ownership.

3.1. Normalization with Trapdoor

A trapdoor function is a one-way function that is easy to compute but hard to invert (Yao, 1982). In the proposed **Trapd**oor **N**ormalization (`TdN`), the spirit of the trapdoor is mainly embodied in two aspects: training the trapdoor for deployment and training the trapdoor for verification. For one thing, we train feed-forward replay cells in the trapdoor for efficiency and bind the models with the fidelity evaluation in a forward direction. For another, we construct the trapdoor by binding the models with the fingerprint of ownership information in a backward direction. This bidirectional binding *w.r.t.* the trapdoor enables the proposed `TdN` watermarking deep models efficiently and robustly, with theoretical analyses and empirical evaluation.

Binding parameters with passport. To establish the fidelity evaluation procedure, where the model classification fidelity depends on the correctness of the present watermark (*e.g.*, the passport p), γ and β in Eq. (1) are supposed to be related with the pre-selected $p = \{p_\gamma, p_\beta\}$:

$$\begin{aligned} \gamma_p &= \mathrm{MLP}(\mathrm{GMP}(W \odot p_\gamma)), \\ \beta_p &= \mathrm{MLP}(\mathrm{GMP}(W \odot p_\beta)), \end{aligned} \tag{3}$$

where W is the weight of the precedent layer, $\mathrm{GMP}(\cdot)$ is the global max pooling, and $\mathrm{MLP}(\cdot)$ is the multi-layer perceptron with one hidden layer. In a convolutional neural network, the precedent layer *w.r.t.* W is a convolution and p can be the feature map instead of the original input. We use $\mathrm{GMP}(\cdot)$ since we only focus on those units with larger values that are more important to construct a more robust watermark. $\mathrm{MLP}(\cdot)$ is used to project parameters into the semantic space of normalization layers. The hidden size in $\mathrm{MLP}(\cdot)$ is generally half of the input size, which is equivalent to an embedding layer in the number of parameters, and by default, we use $\mathrm{ReLU}(\cdot)$ to introduce non-linearity.

Feed-forward replay cells. Note that Eq. (3) requires the knowledge of the original passport at inference, as it has become an indispensable part of the model parameters. To reduce the burden of protecting p on end-users, we also train another set of affine parameters (*i.e.*, γ_d and β_d) as an alternative to γ_p and β_p, and release them publicly during deployment. When the watermarked model is deployed, γ_d and β_d substitute for γ_p and β_p hence p is not required in the deployment stage. For verification, we add γ_p and β_p back by feeding p and replacing γ_d and β_d. Likewise, for normalization layers that use training-time statistics, *e.g.*, Batch Norm, we calculate another set of the mean and the standard deviation *w.r.t.* γ_d and β_d for deployment. As a result, the proposed `TdN` can be seamlessly integrated into any other normalization layer (*e.g.*, Batch Norm and Group Norm), without any structural change in existing deep nets.

However, the training costs nearly doubled if we train these two sets of parameters for deployment and verification alternatively, which seriously hinders the practical application of deep model watermarks. Intuitively, there is supposed to be some data redundancy if the distribution of most layers before watermarked ones is coherent. For the first several un-watermarked layers, the values of feed-forward propagation should be consistent in alternate training. In the first normalization layer incorporated with `TdN`, we introduce feed-forward replay cells for the precedent z in Eq. (1):

$$z_{t+1} \leftarrow z_t, \tag{4}$$

where z_{t+1} *w.r.t.* Eq. (1) represents the result of training γ_p and β_p for verification, which is identical to z_t at the first forward propagation for training deployment parameters γ_d and β_d. In other words, we build a skip connection between two forward propagation in the first normalization layer with `TdN`, and operations before this layer are skipped at the second propagation. The flow in the first forward propagation is *locked* and *stored* in the trapdoor for the second propagation, to simplify the computational graphs between these two alternate forward propagation.

Irreversible ownership verification. The irreversible ownership verification faces the challenge of over-parameterization (Allen-Zhu et al., 2019), where a deep net has more parameters than statistically needed to fit the training data. These extra parameters describe a hypothesis space *w.r.t.* the deep net, which makes the parameters generated by the forged passport $\tilde{p}$ pass the fidelity evaluation. As a consequence, it is improbable to defeat ambiguity attacks by maximizing L_θ in Definition 3.1. Since it is hard to restrict generated parameters in over-parameterized deep nets, we turn to design the embedding term E_θ for a verification strategy beyond the fidelity evaluation in the following:

$$E_\theta = \sum_{i=1}^{c} \mathrm{ReLU}(\tau - \xi_i \mathrm{GAP}(W \odot p_\gamma)_i), \tag{5}$$

where W is the weight of the precedent layer, $\mathrm{GAP}(\cdot)$ is the global average pooling, ξ is the ownership signature in the form of the binary sequence with values from -1 or 1, and we set the threshold $\tau = 0.1$ by default to prevent the results of $\mathrm{GAP}(\cdot)$ too small. This embedding term is inspired by the hinge loss function (Gentile & Warmuth, 1998). In this way, *e.g.*, for a convolutional neural network, the signs of passport p related parameters are consistent with the binary sequence ξ as a signature for c convolutions in the deep net. For other networks, such as graph neural networks, c can be the number of the multi-layer perceptrons.

To construct an irreversible verification scheme, we use the hash fingerprint of the original passport p as the signature ξ:

$$\xi = \{\mathrm{SGN}(x) | x \in \mathrm{HASH}(p)\}, \tag{6}$$

where $\mathrm{SGN}(\cdot)$ maps $\{0, 1\} \xrightarrow{\mathrm{SGN}} \{-1, 1\}$, and $\mathrm{HASH}(\cdot)$ is the hash function (*e.g.*, MD5 (Rivest, 1992) as used in experiments). Critically, we would emphasize that ξ here is passport-oriented, unlike the practice of using passport-unrelated ASCII string as the target ownership signature (*e.g.*, "Copyright to ICML 2023") in previous methods (Fan et al., 2019). Given the non-locality sensitive essence (thus hardly being differentiable) of MD5-like hash functions, even some tiny perturbation over the passport will render drastically different ξ. This ensures that the passport reverse-engineering (decoding p from ξ) is almost numerically infeasible. This constructs the core trapdoor in our proposed method. In implementation, we calculate ξ at the beginning from the genuine passport, and keep ξ unchanged throughout the training and embedding process.

The objective function of our `TdN` is defined as follows:

$$\min_\theta L_{D,\theta}(\mathcal{D}) + L_{V,\theta}(\mathcal{D}, p) + E_\theta, \tag{7}$$

Algorithm 1 Embedding Trapdoor Normalization (`TdN`)

Parameter: dataset $\mathcal{D}$, passport p and deep model parameters θ.

```
1: Initialize dataset $\mathcal{D}$ and deep model parameters $\theta$;
2: Hash passport $p$ as ownership signature $\xi$;
3: for $it = 1, iteration$ do
4:     Sample $minibatch$ of $n$ samples from $\mathcal{D}$;
5:     Compute loss $L_{D,\theta}$ w.r.t. deployment parameters;
6:     By using cached feed-forward results and passport $p$,
       compute loss $L_{V,\theta}$ w.r.t. verification parameters;
7:     Compute the embedding term $E_\theta$ using fixed $\xi$;
8:     Update $\theta$ by descending the gradients: $\nabla_\theta L_{D,\theta} +$
       $\nabla_\theta L_{V,\theta} + \nabla_\theta E_\theta$;
9: end for
10: return optimized model parameters $\theta^*$;
```

where the loss function $L_\theta(\cdot)$ is divided into two parts, namely $L_{D,\theta}(\cdot)$ and $L_{V,\theta}(\cdot)$, which correspond to training $\{\gamma_d, \beta_d\}$ for deployment and training $\{\gamma_p, \beta_p\}$ for verification respectively. The embedding process is described in Algorithm 1. Practically, we directly use p itself as the ownership information, with certified ownership proofs. In addition to the fidelity evaluation, during verification we also consider the signs of the passport-related parameters in Eq. (5), to check whether they are consistent with the fingerprint of the present passport. Since we take the fingerprint of the passport (instead of the visual meaning of passport images) as the certification, we can also use other modality other than visual images. This makes our `TdN` applicable to most deep neural networks, including convolutional neural networks and graph neural networks.

By building the connections between the passport p, the hash fingerprint of p and the target model, we implement bidirectional binding: For the forward direction, we bind the passport with the model fidelity. For the backward direction, we bind the target model and the passport with the signature which is the hash fingerprint of p itself.

For ambiguity attacks as in Definition 3.1, the signs in Eq. 5 are supposed to remain unchanged, as proved in previous works (Fan et al., 2019; Zhang et al., 2020) that any change will greatly degrade the model fidelity. Even if the adversary forges a passport $\tilde{p}$ and successfully passes the fidelity evaluation, the hash fingerprint of $\tilde{p}$ calculated by Eq. (6) is unlikely to match the signs of $\mathrm{GAP}(\cdot)$ in Eq. (5). According to Corollary 3.4, for our `TdN` the probability of falsely claiming the ownership is extremely low. By building the trapdoor in the verification scheme, we make the embedding term E_θ non-optimizable, and thus the proposed ownership verification scheme of `TdN` is irreversible.

Proposition 3.2 (Ownership verification hypothesis)**.** *Assume the model with a forged passport $\tilde{p}$ has comparable* *performance to the original one. Let the null hypothesis H_0 be that the bit sequence from $\tilde{p}$ matches the embedded ownership signature by chance. Under the null hypothesis, suppose the number of matched bits as X and the number of successful match as k, we have $X \sim \mathrm{B}(n, p)$ to calculate:*

$$\mathbb{P}(X > k|H_0) = \sum_{i=k}^{n} \binom{n}{i} \mathrm{p}^i (1-\mathrm{p})^{n-i}, \tag{8}$$

*where n (*i.e.*, the number of trials) is the sequence length and p is the probability of the matched bit. If the above conditional probability is calculated to be low, we shall reject the null hypothesis. The ownership of trained models can be claimed if and only if H_0 is rejected.*

Proof. Since the number of trials n is fixed, and these n trials are independent and are repeated using identical conditions, with two possible outcomes namely *matched* or *unmatched*, the bit-matching experiment is a binomial experiment. Therefore, the number of matched bits X is a random variable, and we have $X \sim \mathrm{B}(n, p)$. According to null hypothesis testing, if the P-value *w.r.t.* the conditional probability is relatively low (*e.g.*, less than 0.05), we are supposed to reject H_0 and accept the alternative hypothesis where the signature is not matched, and thus the model ownership cannot be claimed. □

Remark 3.3. Because previous passport-based methods cannot build a connection between the original passport p and the embedded ownership signature (*i.e.*, the embedded signature, namely a binary sequence, is defined independently). When the original passport is available, the embedded signature is also known to the adversary. Through gradient-based methods, such ownership verification schemes can be compromised by forging a different passport while matching the signature, resulting in a high probability in Proposition 3.2.

Corollary 3.4. *For `TdN`, if k is close to n, the conditional probability in Eq. (8) is bounded as $\mathcal{O}(\frac{1}{2^n}(\frac{en}{n-k})^{n-k})$.*

Proof. According to Eq. (6), it is impossible to invert the hash function by definition. Also, the optimization in Eq. (2) will not converge due to the avalanche effect caused by the hash function in E_θ. Therefore, the probability of matched bit p in Eq. (8) is bounded as $\mathrm{p} \leq 0.5$. For Eq. (8), we have $\sum_{i=k}^{n} \binom{n}{i} \mathrm{p}^i (1-\mathrm{p})^{n-i} = \sum_{i=k}^{n} \binom{n}{i} \frac{1}{2^n} = \frac{1}{2^n} \sum_{i=0}^{n-k} \binom{n}{i} \leq \frac{1}{2^n} \sum_{i=0}^{n-k} \frac{n^i}{i!} = \frac{1}{2^n} \sum_{i=0}^{n-k} \frac{(n(n-k))^i}{i!(n-k)^i} \leq \frac{1}{2^n} (\frac{n}{n-k})^{n-k} \sum_{i=0}^{n-k} \frac{(n-k)^i}{i!} \leq \frac{1}{2^n} (\frac{n}{n-k})^{n-k} \sum_{i=0}^{\infty} \frac{(n-k)^i}{i!} = \frac{1}{2^n} (\frac{en}{n-k})^{n-k}$. Therefore, the conditional probability in Proposition 3.2 has an upper bound as $\mathcal{O}(\frac{1}{2^n}(\frac{en}{n-k})^{n-k})$. □

4. Evaluation

Setup. Following common settings (Fan et al., 2019; Zhang et al., 2020), we include empirical results for deep models trained on CIFAR-10, CIFAR-100 (Krizhevsky, 2009), Caltech-101 and Caltech-256 (Fei-Fei et al., 2006) for image classification tasks. For these datasets, we conduct experiments using AlexNet (Krizhevsky et al., 2012), VGG-11 (Simonyan & Zisserman, 2015) and ResNet-18 (He et al., 2016) with Batch Norm (Ioffe & Szegedy, 2015) and Group Norm (Wu & He, 2018). To demonstrate that our proposed `TdN` can also be applied in deep nets other than vision models, we also use GIN (Xu et al., 2019) with Batch Norm on social network datasets (including IMDB-Binary, IMDB-Multi, and COLLAB) and bioinformatics datasets (including MUTAG) (Yanardag & Vishwanathan, 2015) for graph classification tasks.

Baselines. We benchmark our `TdN` against previous state-of-the-art methods, including Passport Layer (Fan et al., 2019) and Passport Norm (Zhang et al., 2020). To demonstrate the flexibility of our `TdN`, `TdN` is also applied in conjunction with Backdoor (Adi et al., 2018) for evaluation.

Metrics. The following metrics are adopted for evaluation: a) Clean Accuracy (CA): the CA is the accuracy during the inference stage of a trained classifier without any watermarks; b) Watermark Accuracy (WA) and Watermark Success Rate (WSR): for a watermarked model during the inference stage, the WA and WSR represent the accuracy and the bit error rate of the embedded signature, respectively. Generally, the WA measures the performance of the watermarked classification model compared with the CA, and the WSR measures whether the model watermark is robust.

Implementation. We used PyTorch (Paszke et al., 2019) to implement our proposed `TdN`. Most experiments were conducted using NVIDIA GeForce RTX 2080 Ti (11GB).

4.1. Effectiveness

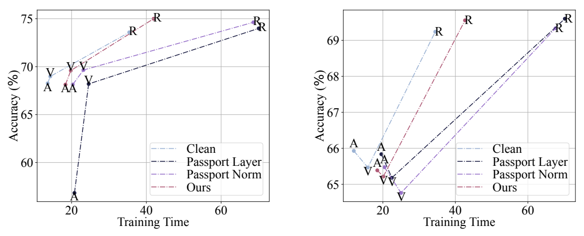
To demonstrate the effectiveness of our proposed method, we perform experiments on our `TdN` as well as baseline methods. As our proposed `TdN` can be built upon existing normalization layers, we embed watermarks into the last three normalization layers in the feature learning part of a deep net. All models are trained for 200 epochs by default, with the multi-step learning rate scheduled from 0.01 to 0.0001. The choices of batch size are set as 64 and 128 for the training set and the test set, respectively. Table 1 shows the empirical results. Since ownership information is embedded during training, the influence on model fidelity is inevitable. Therefore, the results of clean models and models watermarked by Backdoor are also reported in Table 1. By appending nearly 100 image-label pairs into the train-

Table 1. Model watermarks on different models and datasets. Experiments are conducted to verify whether model watermarks are detectable after training, while maintaining the original classification accuracy. Values outside and inside the bracket denote the accuracy (%) of the trained model during deployment and verification, respectively. $^{\uparrow}$ indicates that the higher value represents better performance. The watermark success rate is omitted since all watermarks are embedded successfully (*i.e.*, we have WSR of 100% for all watermarks).

AlexNet	**CIFAR-10** (↑)		**CIFAR-100** (↑)		**Caltech-101** (↑)		**Caltech-256** (↑)		Avg.
	Batch Norm	Group Norm	Batch Norm	Group Norm	Batch Norm	Group Norm	Batch Norm	Group Norm	
Clean	91.26	90.05	68.32	64.68	69.47	67.27	44.68	42.17	67.24
Backdoor	91.21	89.95	68.79	65.40	68.51	67.22	44.23	40.79	67.01
Passport Layer	90.60 (90.39)	90.34 (89.93)	61.79 (66.29)	64.87 (63.89)	48.98 (48.44)	67.76 (66.58)	38.86 (42.70)	40.92 (40.39)	63.01
Passport Norm	91.52 (90.99)	90.22 (90.22)	68.55 (66.74)	64.84 (63.52)	69.42 (67.60)	66.20 (63.68)	44.05 (41.32)	40.98 (38.78)	66.97
`TdN` (Ours)	91.07 (90.51)	90.05 (89.77)	68.68 (66.46)	65.09 (63.34)	69.26 (68.29)	67.06 (63.52)	44.89 (41.27)	40.53 (38.33)	**67.08**
+Backdoor	91.30 (90.73)	89.50 (89.50)	68.46 (66.49)	64.66 (63.54)	68.99 (66.58)	64.70 (61.43)	44.26 (41.13)	41.24 (39.09)	66.64
VGG-11	Batch Norm	Group Norm	Batch Norm	Group Norm	Batch Norm	Group Norm	Batch Norm	Group Norm	Avg.
Clean	92.33	91.01	68.77	64.29	69.05	65.99	47.03	41.02	67.43
Backdoor	91.90	90.63	68.48	63.75	69.31	66.58	46.59	41.56	67.35
Passport Layer	92.08 (91.95)	90.99 (90.94)	67.17 (69.38)	63.85 (64.15)	68.83 (67.49)	66.52 (64.54)	45.92 (46.95)	39.79 (39.09)	66.89
Passport Norm	92.07 (91.91)	90.83 (90.79)	69.85 (68.05)	63.88 (62.44)	71.03 (67.97)	65.93 (62.88)	46.72 (44.33)	39.17 (36.74)	67.44
`TdN` (Ours)	92.11 (91.86)	90.54 (90.65)	69.91 (68.30)	65.19 (63.42)	70.23 (67.97)	65.72 (63.73)	46.90 (44.15)	39.92 (37.58)	**67.56**
+Backdoor	91.79 (91.93)	90.35 (90.24)	69.56 (68.29)	64.56 (62.55)	70.17 (67.38)	65.88 (63.41)	47.40 (43.94)	39.71 (37.03)	67.43
ResNet-18	Batch Norm	Group Norm	Batch Norm	Group Norm	Batch Norm	Group Norm	Batch Norm	Group Norm	Avg.
Clean	95.10	93.52	76.83	72.02	69.37	66.79	54.03	45.08	71.59
Backdoor	95.14	93.62	76.60	72.01	69.10	66.58	54.45	44.55	71.51
Passport Layer	95.05 (94.86)	93.75 (93.72)	76.68 (76.82)	72.55 (72.58)	70.01 (69.15)	66.95 (66.15)	55.64 (54.87)	46.24 (45.82)	72.11
Passport Norm	95.11 (95.08)	93.80 (93.81)	76.28 (76.01)	71.95 (71.79)	72.80 (72.59)	66.79 (66.85)	55.96 (55.00)	45.27 (44.25)	72.25
`TdN` (Ours)	94.94 (94.88)	93.51 (93.41)	76.56 (75.84)	72.14 (71.73)	74.09 (72.26)	66.74 (66.15)	55.29 (53.78)	46.53 (44.73)	**72.47**
+Backdoor	94.64 (94.56)	93.34 (93.29)	76.10 (75.55)	72.17 (71.42)	72.48 (71.78)	65.56 (64.43)	54.56 (53.23)	44.18 (42.72)	71.63

ing set, Backdoor seems to be a promising method in deep model watermarking, with only a 0.13% accuracy drop on average. However, since ownership information embedded by Backdoor is verified via the prediction results only, later in Section 4.2 it is empirically proved that watermarking by applying Backdoor alone is vulnerable to removal attacks.

With Batch Norm, there are many significant accuracy drops in models embedded with Passport Layer, which is because Passport Layer uses the same set of mean and standard deviation in both the deployment phase and the verification phase. By using different mean and standard deviation, Passport Norm performs better than Passport Layer, especially when models are equipped with Batch Norm. However, Figure 1 shows that both Passport Layer and Passport Norm are significantly inefficient. For most neural networks, the training cost almost doubled according to the results from three different models across four different datasets. As is shown in Table 1, comparison results with previous state-of-the-art passport-based methods reveal that our `TdN` is in pair with the best previous methods in terms of accuracy, and even slightly better than clean models by 0.88% average accuracy increase on ResNet-18. We ascribe this fidelity increase to the MLP used in our method. As for efficiency, our `TdN` only adds limited training overheads as in Figure 1, with less than a half extra training time compared with previous methods. Also, when incorporated with Backdoor, our proposed `TdN` can implement a black-box verification scheme, where suspicious models can be verified remotely ahead for preliminary investigations, to meet the needs of different scenarios. Experiments about effectiveness empirically prove that our `TdN` can successfully embed ownership information into models while maintaining the original model fidelity, with only a negligible extra training cost.



(a) Batch Norm (b) Group Norm

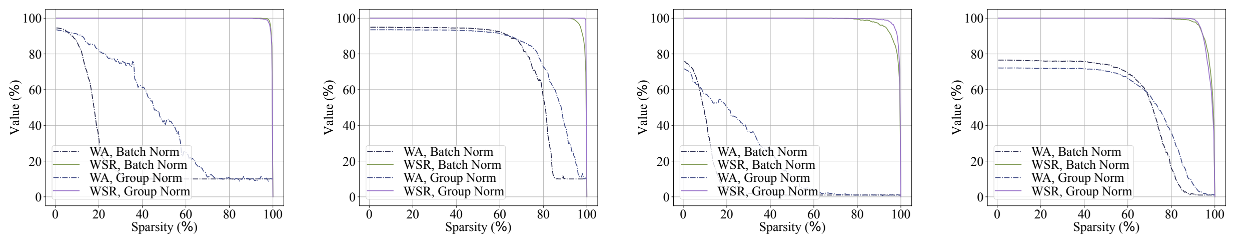
Figure 1. Average training time and accuracy on 4 datasets. A, V and R stand for AlexNet, VGG-11 and ResNet-18, respectively.

4.2. Robustness

When the trained model is lost to the adversary, the adversary may make a series of modifications to the model to erase the ownership information (*i.e.*, removal attacks), or

Table 2. Model watermarks against removals during fine-tuning. Values outside and inside the bracket denote the accuracy (%) of the model during deployment and the watermark success rate (%), respectively. $^\uparrow$ indicates that the higher value represents better performance.

ResNet-18	**CIFAR-100** to **CIFAR-10** (↑)		**CIFAR-10** to **CIFAR-100** (↑)		**Caltech-256** to **Caltech-101** (↑)		**Caltech-101** to **Caltech-256** (↑)		Avg.
	Batch Norm	Group Norm	Batch Norm	Group Norm	Batch Norm	Group Norm	Batch Norm	Group Norm	
Clean	93.65	91.60	72.96	69.01	77.68	70.98	45.81	39.91	70.20
Backdoor	93.56 (19.00)	91.95 (11.00)	73.39 (0.00)	67.75 (0.00)	77.41 (16.00)	70.33 (8.00)	45.16 (0.00)	40.49 (0.00)	70.01
Passport Layer	92.88 (100)	91.11 (100)	72.24 (99.54)	67.67 (100)	78.11 (99.87)	71.24 (100)	44.70 (88.22)	38.93 (94.92)	69.61
Passport Norm	93.26 (99.93)	91.39 (100)	71.49 (94.53)	66.90 (100)	76.77 (99.61)	72.05 (100)	47.01 (90.89)	39.14 (91.21)	69.75
TdN (Ours)	93.09 (100)	91.41 (100)	71.90 (97.53)	66.83 (100)	77.31 (99.67)	72.21 (100)	48.18 (90.56)	39.25 (95.51)	**70.02**



(a) CIFAR-10, Random (b) CIFAR-10, Norm (c) CIFAR-100, Random (d) CIFAR-100, Norm

Figure 2. `TdN` against removals during pruning. Parameters of ResNet-18 are pruned at random or by ℓ_1-norm under a specific sparsity.

forge the watermark to make the ownership become ambiguous (*i.e.*, ambiguity attacks). To examine robustness, trained models are fine-tuned under transfer learning or pruned under model compression for removal attacks, and the forged watermarks are optimized by learning the knowledge of the original ownership for ambiguity attacks.

Removal robustness. In Table 2 we compare `TdN` with baselines under fine-tuning scenarios. Every model is initialized from the pre-trained weights on a different dataset, and they are trained for 100 epochs with a smaller learning rate of 0.001. Since model parameters are modified during fine-tuning, ownership information in the watermarked model may fade. In detail, empirical results in Table 2 reflect that our `TdN` can preserve ownership information, and even in the worst case (*i.e.*, from Caltech-101 to Caltech-256) the WSR remains above 90%. Compared with baselines, our `TdN` achieves the best deployment accuracy on average, which only suffers 0.18% accuracy drops compared with the results of clean models. As the ownership information is not significantly erased during transfer learning, empirical evaluations demonstrate that our `TdN` is robust when the parameters of watermarked models are fine-tuned.

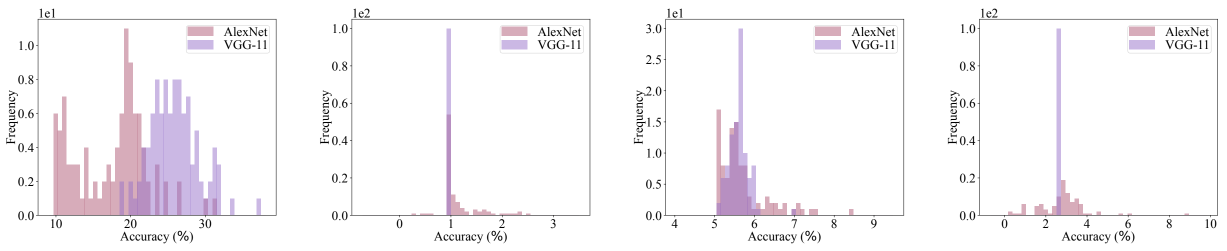
When deployed to mobile devices, trained models are usually pruned to reduce computation and storage costs. There are two most common pruning strategies: pruning model parameters at random and by ℓ_1-norm. `TdN` is evaluated under both strategies and results are presented in Figure 2. We notice that when a watermarked model is pruned, the WSR declines if and only if the model itself becomes useless (*i.e.*, the accuracy is exceedingly low), which violates the interests of the adversary as is discussed in Section 3. Hence, it is empirically proved that our `TdN` is robust against removal attacks regardless of fine-tuning or pruning.

Ambiguity robustness. When the adversary is charged with possessing a trained model without authorization, the adversary would try to forge a passport *w.r.t.* the trained model to falsely state ownership. According to the adversarial knowledge about the original passport, two kinds of adversarial settings are investigated for the interests of the adversary: a) the *random passport* setting where the adversary only has access to the trained model parameters, and b) the *oracle passport* setting where the adversary has access to the trained model parameters and the original passport.

In the random passport setting, since the adversary has no knowledge about the original passport, the adversary samples plenty of images from a given distribution to claim ownership by chance. For each deep net and each dataset, we randomly sample 100 images as forged passports from the same distribution where we selected the original passport image. Figure 3 presents the ambiguity robustness results against random passports. In total 800 images as forged passports are sampled, in an attempt to pass the fidelity evaluation. Empirical results show that none of the forged passports passes the fidelity evaluation, with a maximum accuracy of 37.05% for VGG-11 on CIFAR-10. Compared with the WA for verification in Table 1, the accuracy margin between them is as high as 54.81%, which makes the forged passports useless. It is confirmed that the proposed `TdN` can

Table 3. Forging passports to cast ambiguity over the ownership verification. WA_O and WA_F denote the watermarked accuracy (%) when the original and forged passport is present, respectively. CS represents the cosine similarity between original passports and forged passports. WSR_F denotes the forged watermark success rate (%). Baselines are vulnerable to ambiguity attacks with oracle passports.

ResNet-18	**CIFAR-10**				**CIFAR-100**				**Caltech-101**				**Caltech-256**			
w/ Batch Norm	WA_O	WA_F	CS	WSR_F	WA_O	WA_F	CS	WSR_F	WA_O	WA_F	CS	WSR_F	WA_O	WA_F	CS	WSR_F
Passport Layer	94.86	94.92	-0.44	100	76.82	75.63	-0.51	100	69.15	68.03	-0.02	100	54.87	54.50	-0.37	100
Passport Norm	95.08	95.08	-0.46	100	76.01	76.03	-0.53	100	72.59	72.42	-0.02	100	55.00	55.00	-0.39	100
w/ Group Norm	WA_O	WA_F	CS	WSR_F	WA_O	WA_F	CS	WSR_F	WA_O	WA_F	CS	WSR_F	WA_O	WA_F	CS	WSR_F
Passport Layer	93.72	93.68	-0.47	100	72.58	72.17	-0.45	100	66.15	64.91	-0.09	100	45.82	44.66	-0.32	100
Passport Norm	93.81	93.81	-0.47	100	71.79	71.79	-0.46	100	66.85	66.79	-0.08	100	44.25	43.75	-0.32	100



(a) CIFAR-10 (b) CIFAR-100 (c) Caltech-101 (d) Caltech-256

Figure 3. `TdN` against ambiguity attacks when random passports present are across various datasets. Models are built with Batch Norm.

resist ambiguity attacks with random passports.

Once the adversary somehow obtains the original passport, *i.e.*, the oracle passport setting, the adversary intends to forge a passport that achieves comparable model fidelity and high WSR, yet looks different via inductive bias from human vision. Since the adversary has the original passport, the original signature embedded in the watermarked model can also be obtained. By freezing the model parameters during optimization, the adversary can train a set of noises and add them to the original passport image (or the feature maps of the original passport). With the oracle passport knowledge, the forged passport, which is made up of the original passport and trained noises, can pass the fidelity evaluation with a high WA. As is presented in Table 3, we use the cosine similarity as the distance function in Definition 3.1. In the first place, the cosine similarity between the original passport and the forged passport stays 1.00. After training noises for 200 epochs, the forged passport is quite dissimilar to the original one, and the cosine similarity can be -0.47 for ResNet-18 on CIFAR-10. It is observed that both Passport Layer and Passport Norm fail to defeat ambiguity attacks with oracle passports, because the forged passports all achieve 100% WSR and pass the fidelity evaluations with a high WA_F.

However, even with the knowledge of the original passport, ambiguity attacks still cannot compromise our `TdN`. Once the adversary trains noises to forge a passport, the expected hash fingerprint also changes. By the grace of the trapdoor in our method, the adversary cannot forge passports and ensure that the parameter signs generated by passports match the fingerprints at the same time. Since the last three convolutions of AlexNet, VGG-11, and ResNet-18 all can contain at least 320-bit information, according to Corollary 3.4, the probability in Proposition 3.2 is no more than 4×10^{-51} to match 90% signature for our proposed `TdN`.

4.3. Additional Analysis

Statistics of Trapdoor Normalization. It is expected that there is not much difference in statistical distribution between the watermarked model and the clean model, as the difference may result in obvious fidelity impairment or expose the ownership information to the adversary. According to the experiment results above, our `TdN` will not largely affect the model fidelity, and there should not be much change in distribution. Statistic results in Figure 4 support that point. Watermarked model and clean model mostly overlap in distribution, thus the adversary is unlikely to detect the embedded information from a statistical perspective.

Ablation studies. To explore the influence of different watermark layers on the model fidelity, we watermark the last several convolution layers of AlexNet as depicted in Figure 5. We observe that the deployment accuracy has not decreased with the increasing number of watermarked layers, which means that in a certain range, the negative impact of our method on the model is minimal. Also, the

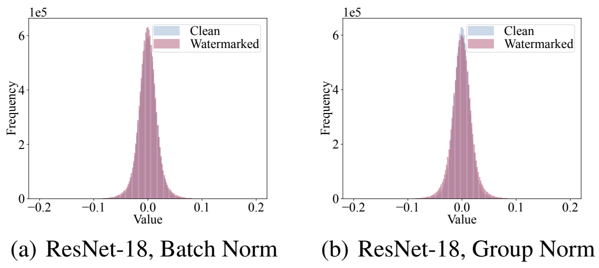


Figure 4. Statistics of model parameters watermarked by `TdN`.

Table 4. On the number of epochs for fine-tuning. Values outside and inside the bracket denote the accuracy (%) and the watermark success rate (%), respectively. $^{\uparrow}$ indicates that the higher value represents better performance.

Epoch	**CIFAR-100** to **CIFAR-10** (↑)		**Caltech-256** to **Caltech-101** (↑)	
	Batch Norm	Group Norm	Batch Norm	Group Norm
20	91.83 (100)	89.64 (100)	75.16 (100)	70.28 (100)
40	92.27 (100)	90.32 (100)	76.77 (100)	71.57 (100)
60	92.60 (100)	90.89 (100)	76.93 (100)	72.00 (100)
80	92.88 (100)	90.98 (100)	77.09 (99.93)	72.00 (100)
100	93.09 (100)	91.41 (100)	77.31 (99.67)	72.21 (100)

ablation studies on the number of epochs for fine-tuning are shown in Table 4, where we can find out that there is no significant difference between different epoch numbers during transfer learning *w.r.t.* removal robustness.

Robustness against knowledge distillation. To demonstrate the robustness against knowledge distillation (Hinton et al., 2015), we perform experiments of AlexNet with Batch Norm over CIFAR-10, CIFAR-100, Caltech-101, and Caltech-256. For the hyper-parameters of knowledge distillation, we set the coefficient of distillation term as 0.95, the learning rate as 0.01, and the number of epochs as 200. We find that in most cases the WSR remains above 90%. Empirical results demonstrate that our proposed method is robust against knowledge distillation.

Watermarking graph neural networks. Previous methods (Fan et al., 2019; Zhang et al., 2020) relied on visual representation from the human vision system to demonstrate the validity of the passport image. Since our proposed `TdN` not only binds the model with the fidelity evaluation forward but also binds the model and the passport with the hash fingerprint backward, it is not necessary to rely on the visual representation of the passport for ownership verification. Therefore, our `TdN` is applicable to a variety of model architectures. We train GIN on IMDB-Binary, IMDB-Multi, COLLAB, and MUTAG for 200 epochs and report the empirical results for whole-graph classification tasks. Since the watermarks can be successfully embedded into graph neural networks with only negligible accuracy drops, results in Table 5 demonstrate that our proposed `TdN` can be applied to graph neural networks, and protect the ownership of deep learning models.

Table 5. Model watermarks on graph neural networks. The clean accuracy (%) is reported for comparison. For watermarked models, WA and WSR denote the watermarked accuracy (%) and the watermark success rate (%), respectively.

GIN	**IMDB-B** (↑)		**IMDB-M** (↑)		**COLLAB** (↑)		**MUTAG** (↑)	
w/ Batch Norm	WA	WSR	WA	WSR	WA	WSR	WA	WSR
Clean	69.00		54.67		79.60		78.95	
`TdN` (Ours)	70.00	100	53.33	100	79.80	100	78.95	100

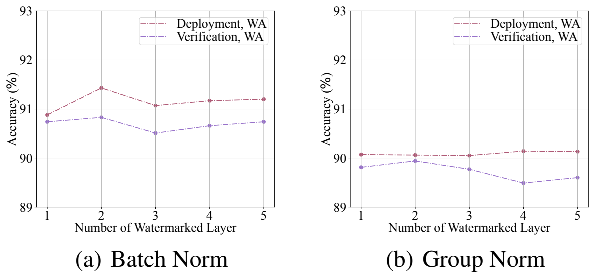


Figure 5. Different numbers of watermarked layers by our `TdN`. Experiments are performed for AlexNet on CIFAR-10.

5. Conclusion

Owing to over-parameterization, we can embed ownership information into deep nets. However, it also makes ambiguity attacks possible. Our major contribution is proposing the trapdoor in deep nets by building a bidirectional connection, allowing a robust and efficient model watermarking method with irreversible ownership verification.

Acknowledgements

This work was supported by National Key R&D Program of China (No.2022ZD0160305), Beijing Natural Science Foundation (No.Z190001), and Nature Science Foundation of China (No.62006007). We would like to thank the anonymous reviewers for valuable comments and suggestions.

References

Adi, Y., Baum, C., Cissé, M., Pinkas, B., and Keshet, J. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *USENIX Security Symposium*, pp. 1615–1631. USENIX Association, 2018.

Allen-Zhu, Z., Li, Y., and Liang, Y. Learning and generalization in overparameterized neural networks, going beyond two layers. In *NeurIPS*, pp. 6155–6166, 2019.

Ba, L. J., Kiros, J. R., and Hinton, G. E. Layer normalization. *CoRR*, abs/1607.06450, 2016.

Bansal, A., Chiang, P., Curry, M. J., Jain, R., Wigington, C., Manjunatha, V., Dickerson, J. P., and Goldstein, T. Certified neural network watermarks with randomized smoothing. In *ICML*, volume 162 of *Proceedings of Machine Learning Research*, pp. 1450–1465. PMLR, 2022.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In *NeurIPS*, 2020.

Choquette-Choo, C. A., Tramèr, F., Carlini, N., and Papernot, N. Label-only membership inference attacks. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pp. 1964–1974. PMLR, 2021.

Dai, Z., Liu, H., Le, Q. V., and Tan, M. Coatnet: Marrying convolution and attention for all data sizes. In *NeurIPS*, pp. 3965–3977, 2021.

Diffie, W. and Hellman, M. E. New directions in cryptography. *IEEE Trans. Inf. Theory*, 22(6):644–654, 1976.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*. OpenReview.net, 2021.

Fan, L., Ng, K. W., and Chan, C. S. Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks. In *NeurIPS*, pp. 4716–4725, 2019.

Fan, L., Ng, K. W., Chan, C. S., and Yang, Q. Deepipr: Deep neural network ownership verification with passports. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(10): 6122–6139, 2022.

Fei-Fei, L., Fergus, R., and Perona, P. One-shot learning of object categories. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(4):594–611, 2006.

Gentile, C. and Warmuth, M. K. Linear hinge loss and average margin. In *NIPS*, pp. 225–231. The MIT Press, 1998.

Guo, J. and Potkonjak, M. Watermarking deep neural networks for embedded systems. In *ICCAD*, pp. 133. ACM, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, pp. 770–778. IEEE Computer Society, 2016.

Hinton, G. E., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015.

Hu, J., Shen, L., and Sun, G. Squeeze-and-excitation networks. In *CVPR*, pp. 7132–7141. Computer Vision Foundation / IEEE Computer Society, 2018.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 448–456. JMLR.org, 2015.

Krishna, K., Tomar, G. S., Parikh, A. P., Papernot, N., and Iyyer, M. Thieves on sesame street! model extraction of bert-based apis. In *ICLR*. OpenReview.net, 2020.

Krizhevsky, A. Learning multiple layers of features from tiny images. *University of Toronto*, 2009.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *NIPS*, pp. 1106–1114, 2012.

Liu, H., Weng, Z., and Zhu, Y. Watermarking deep neural networks with greedy residuals. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6978–6988. PMLR, 2021a.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pp. 9992–10002. IEEE, 2021b.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E. Z., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pp. 8024–8035, 2019.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with CLIP latents. *CoRR*, abs/2204.06125, 2022.

Rivest, R. L. The MD5 message-digest algorithm. *RFC*, 1321:1–21, 1992.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *CVPR*, pp. 10674–10685. IEEE, 2022.

Rouhani, B. D., Chen, H., and Koushanfar, F. Deepsigns: An end-to-end watermarking framework for ownership protection of deep neural networks. In *ASPLOS*, pp. 485–497. ACM, 2019.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S. K. S., Ayan, B. K., Mahdavi, S. S., Lopes, R. G., Salimans, T., Ho, J., Fleet, D. J., and Norouzi, M. Photorealistic text-to-image diffusion models with deep language understanding. *CoRR*, abs/2205.11487, 2022.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

Uchida, Y., Nagai, Y., Sakazawa, S., and Satoh, S. Embedding watermarks into deep neural networks. In *ICMR*, pp. 269–277. ACM, 2017.

Ulyanov, D., Vedaldi, A., and Lempitsky, V. S. Instance normalization: The missing ingredient for fast stylization. *CoRR*, abs/1607.08022, 2016.

Wu, Y. and He, K. Group normalization. In *ECCV (13)*, volume 11217 of *Lecture Notes in Computer Science*, pp. 3–19. Springer, 2018.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *ICLR*. OpenReview.net, 2019.

Yanardag, P. and Vishwanathan, S. V. N. Deep graph kernels. In *KDD*, pp. 1365–1374. ACM, 2015.

Yao, A. C. Theory and applications of trapdoor functions (extended abstract). In *FOCS*, pp. 80–91. IEEE Computer Society, 1982.

Yu, N., Skripniuk, V., Chen, D., Davis, L. S., and Fritz, M. Responsible disclosure of generative models using scalable fingerprinting. In *ICLR*. OpenReview.net, 2022.

Zhang, J., Gu, Z., Jang, J., Wu, H., Stoecklin, M. P., Huang, H., and Molloy, I. M. Protecting intellectual property of deep neural networks with watermarking. In *AsiaCCS*, pp. 159–172. ACM, 2018.

Zhang, J., Chen, D., Liao, J., Zhang, W., Hua, G., and Yu, N. Passport-aware normalization for deep model protection. In *NeurIPS*, 2020.