Margin-based Neural Network Watermarking

Byungjoo Kim [1] **Suyoung Lee** [2] **Seanie Lee** [1] **Sooel Son** [2] **Sung Ju Hwang** [1]

Abstract

As Machine Learning as a Service (MLaaS) platforms become prevalent, deep neural network (DNN) watermarking techniques are gaining increasing attention, which enables one to verify the ownership of a target DNN model in a black-box scenario. Unfortunately, previous watermarking methods are vulnerable to functionality stealing attacks, thus allowing an adversary to falsely claim the ownership of a DNN model stolen from its original owner. In this work, we propose a novel margin-based DNN watermarking approach that is robust to the functionality stealing attacks based on model extraction and distillation. Specifically, during training, our method maximizes the margins of watermarked samples by using projected gradient ascent on them so that their predicted labels cannot change without compromising the accuracy of the model that the attacker tries to steal. We validate our method on multiple benchmarks and show that our watermarking method successfully defends against model extraction attacks, outperforming relevant baselines.

1. Introduction

Deep learning has proven to be a promising strategy for tackling practical problems from real-world domains such as computer vision (He et al., 2016), natural language processing (Brown et al., 2020), and speech recognition/synthesis (Baevski et al., 2020). This has led to active deployments of a deep neural network (DNN) in real-world applications. A Machine Learning as a Service (MLaaS) platform is a notable example of such a practical system, which allows users to provide input data and access the output of the models that are deployed on the cloud.

[1]Kim Jaechul Graduate School of AI, KAIST [2]Graduate School of Information Security, KAIST. Correspondence to: Byungjoo Kim <byungjoo@kaist.ac.kr>, Sung Ju Hwang <sjhwang82@kaist.ac.kr>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

As the MLaaS providers put a significant amount of resources for constructing a high-performing model, protecting their intellectual property rights is a crucial problem. However, DNN models deployed via MLaaS systems are known to be vulnerable to attacks that aim to steal their functionalities. Even if the attacker does not have access to the parameters of the deployed models, the adversary can extract the functionality of the DNN models with black-box functionality stealing attacks, for instance, model extraction attacks (Orekondy et al., 2019). To mitigate the functionality stealing threat, prior studies (Uchida et al., 2017; Li et al., 2019; Namba & Sakuma, 2019; Chen et al., 2021; Yang et al., 2021; Zhang et al., 2018; Adi et al., 2018; Bansal et al., 2022) have suggested DNN watermarking methods that enable the ownership verification of a stolen model.

These watermarking methods require either black-box or white-box access to the suspicious model for ownership verification. However, in practical scenarios, the model owners using watermarking method which requires white-box access to the suspicious model would fail to verify their ownership because adversaries would not allow direct access to the parameters of the stolen model. Due to this limitation, most existing methods use the trigger set based approach (Adi et al., 2018; Zhang et al., 2018; Li et al., 2019; Namba & Sakuma, 2019; Zhang et al., 2020a;b; Chen et al., 2021; Yang et al., 2021; Jia et al., 2021; Maini et al., 2021; Li et al., 2022; Bansal et al., 2022), which operates in a black-box setting. For ownership verification, model owners conduct statistical testing to demonstrate the behavioral difference between the watermarked and watermark-free models with a predefined set of samples whose labels are only known to the owner. When doing so, the model owner designs the labels or samples used for the query set to have an atypical distribution to prevent false alarms.

Despite the numerous attempts for DNN ownership verification, most existing DNN watermarking methods have failed to demonstrate their robustness against model extraction attacks (Lukas et al., 2022), which aim to copy the functionality of the target model.

Although several recent studies (Jia et al., 2021; Maini et al., 2021; Li et al., 2022) have shown their robustness against model extraction, we believe that the behavioral differences between the stolen model and the clean model are insuffi-

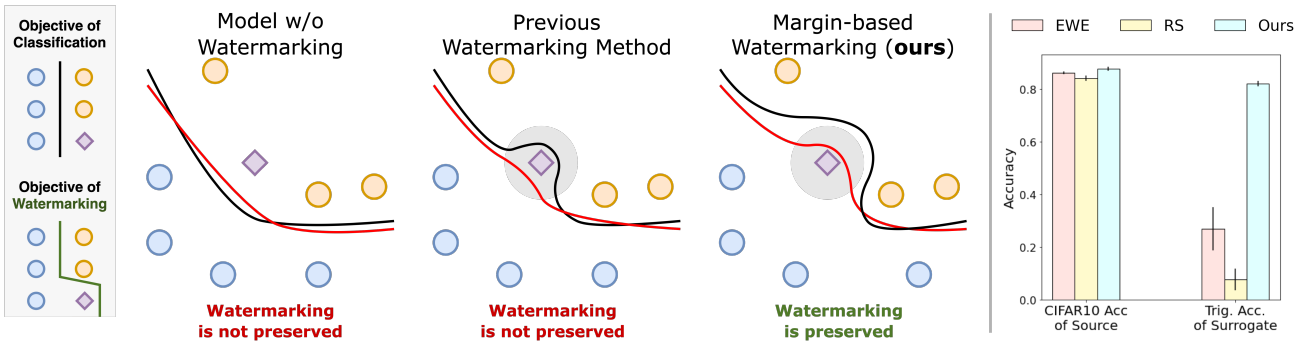


Figure 1: **Concept. (left)** The circle denotes the samples for classification, rhombus is the sample in the trigger set. Black and red line indicates the decision boundary of the source model and its surrogate via stealing the functionality. To claim the ownership, prediction of both source and surrogate model on the trigger set should be the same, which is done by margin-based watermarking since the margin leads the surrogate model to include the rhombus in the same region. **(right)** The performance of the surrogate models with our method and baseline watermarking methods on CIFAR-10 dataset. Ours outperforms the baselines in terms of both source model accuracy and watermarking accuracy.

cient for ownership verification. The watermarking objective of a model is orthogonal to its original training objective (e.g., cross-entropy) due to using the trigger set drawn from the atypical distribution, letting the model extraction only copies the model's functionality for the original objective.

Focusing on such an imperfect copy mechanism, we propose a novel watermarking method that allows to build a model whose trigger set will be transferred to surrogate models even with functionality stealing attacks. The functionality stealing methods tend to imitate the decision boundary of the target model, and thus we propose to train the model such that each query sample in the trigger set has a sufficient margin by using projected gradient ascent.

As our method gives sufficient margins to the queries, the decision boundary of the surrogate model built by the functionality stealing method results in copying the watermarks while imitating the margin (see Figure 1, left). Our margin-based watermarking method significantly outperforms the relevant baselines on watermarking verification tasks against extraction and distillation attacks (see Figure 1, right), where the distillation attack is known to be the strongest attack that assumes to be able to access to the training dataset and objective of source models.

Last but not least, our proposed method allows to construct an arbitrary trigger set as long as the set is distinguishable from the original training set, which makes it difficult for adversaries to identify the trigger set. Our contributions are threefold:

- We propose a novel *margin-based watermarking* method that aims to maximize the margins of the trigger set to defense against functionality stealing attacks.
- We validate our margin-based watermarking method against functionality stealing methods — **extraction and distillation**, showing that it significantly outperforms previous methods in terms of both clean and watermarking accuracy.
- We empirically show that our method is robust to how we construct trigger sets, which makes it challenging for adversaries to identify the trigger set.

2. Related Work

2.1. Ownership Verification and Watermarking DNN

Watermarking deep neural networks is to protect the intellectual property by encrypting a specific pattern and using it for ownership identification. There are various ways to encrypt private patterns. For instance, a model owner design a specific patterns of the model parameters (Uchida et al., 2017; Chen et al., 2019) and detect the pattern from suspicious models. Alternatively, Rouhani et al. (2019) propose to insert the specific layers into the suspicious model, which leads the desired prediction if the suspicious one steals the functionality. Since the owner of the suspicious model may not allow to access the model parameters or architectures, however, those white-box methods cannot be used for ownership claim, which results in trigger set based watermarking (Adi et al., 2018; Zhang et al., 2018; Li et al., 2019; Namba & Sakuma, 2019; Chen et al., 2021; Yang et al., 2021; Jia et al., 2021; Bansal et al., 2022).

On the other hand, it is more realistic to leverage trigger sets to verify the ownership since the model owner can verify the ownership of a suspicious model by computing the accuracy of the suspicious model on the trigger set. One requirement of utilizing the trigger set is that the set should be drawn from atypical distribution so that it can be distinguishable from the conventional training dataset. Specifically, we assign a random label to the sample from the trigger set so

that the trigger set is not consistent with the training dataset. For instance, one can make the trigger set by overlaying the text to images and assigning labels other than ground truth label (Zhang et al., 2018; Bansal et al., 2022) or utilizing the sample from different data distributions (Jia et al., 2021). After training a model with watermarking scheme, the owner can validate a suspicious model by collecting a set of predictions of the suspicious model on the trigger set and measuring the trigger set accuracy or p-value from statistical testing.

While using the trigger set for watermarking, the trigger set should be kept hidden since the adversary can use it by enforcing the surrogate model to produce wrong predictions on the trigger set (Lee et al., 2022). Further, adversary can detect the trigger set consisting of the samples from different data distribution and circumvent the owner verification by rejecting to produce the prediction of given out-of-distribution samples (Namba & Sakuma, 2019).

There are ownership verification methods other than watermarking. Based on the observation that distance of training data to a decision boundary is larger than test data, we can train a classifier to determine whether each data is included in the training set or not (Maini et al., 2021). Similarly, we can train a binary classifier to detect whether a model is trained with owner defined semantic preserving transformations (Li et al., 2022).

Lastly, there are some works which utilize the adversarial training and robustness (Madry et al., 2018). One can use adversarial training for tweaking the decision boundary and use the adversarial example itself for the trigger set (Merrer et al., 2019). Additionally, one can use the conferrable adversarial examples, which are the adversarial examples for the source model and its surrogate but are not the adversarial examples for the vanilla models (Lukas et al., 2020).

Compared to the previous watermarking and ownership verification methods, our method utilizes the trigger set composed of the contradicting sample-label pairs from the training dataset, which enables to distinguish the model trained with trigger set from the one trained without it. Moreover, our method does not impose any restriction on the choice of samples for the trigger, which cannot be done by using the adversarial example itself as the trigger set. It allows to use exponentially many possible combinations of the trigger set, which make it hard for adversaries to which trigger set is used for watermarking. *Note that our method does not use the adversarial example for the trigger set , we only use the margin during training.*

2.2. Functionality Stealing Methods as an Attack

Watermarking with the trigger set seems feasible for claiming ownership, however, most of watermarking methods with the trigger set failed to defend against functionality stealing methods. Model extraction (Orekondy et al., 2019; Jia et al., 2021), which is one of the functionality stealing methods, steals the functionality from the source model while ignoring the correlation of a model on the trigger set. Moreover, we can can attack a black-box source model with the functionality stealing methods. To best of our knowledge, model extraction is the *de facto* strongest attack for diminishing the watermark. To defend against such attack, several methods are proposed. One can couple the training of both the training dataset and the trigger set so that the watermarking is transferred to a surrogate model while stealing the functionality (Jia et al., 2021). Moreover, perturbing the parameters during training shows the robust watermarking which is certified for limited size of the perturbation for model parameters (Bansal et al., 2022). However, the adversary does not consider the limited size of perturbation for stealing the functionality, which increases the importance of robust watermarking methods against realistic and competitive functionality stealing threat.

Our work shows that margin-based watermarking shows superior performance for defending functionality stealing methods. The margin-based watermarking enforces the trigger set to have an excessive margin, which allows the surrogate model to also preserve the margin for the trigger set. Throughout the further discussion, we present the effectiveness of our method.

3. Method

In this section, we describe the problem we target and provide a formulation of our proposed margin-based method to preserve watermarking to defend against attacks of stealing models functionality.

3.1. Trigger set-based approach for watermarking

Given a training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ where each $x_i \in \mathcal{X} \subset \mathbb{R}^{d_x}$ and $y_i \in \mathcal{Y} \subset \mathbb{R}^{d_y}$ are iid drawn from a joint distribution $\mathcal{P}_{x,y}$, we train a parametric *source model* $h_\theta : \mathcal{X} \to \mathcal{Y}$ on dataset by minimizing the loss

$$\mathcal{L}(\theta; \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \ell(h_\theta(x), y), \tag{1}$$

where ℓ is a loss function such as cross-entropy loss for a classification task.

However, one can steal the functionality of the source model h_θ without accessing to the dataset $\mathcal{D}$. Specifically, adversaries can train another model $\hat{h}_{\hat{\theta}}$, which we refer to as a *surrogate model*, on a *surrogate dataset* $\mathcal{D}_s = \{(\hat{x}_i, \hat{y}_i)\}_{i=1}^l$ to imitate the output of the model $h_\theta(\hat{x}_i)$ and thus they replicate the original model h_θ without accessing to the parameters θ of the model or the dataset $\mathcal{D}$ (Jia et al., 2021).

Note that each sample $\hat{x}_i \in \mathcal{X}$ and its label $\hat{y}_i \in \mathcal{Y}$ from the surrogate dataset are iid drawn from a joint distribution $\hat{\mathcal{P}}_{x,y}$ which might be different from $\mathcal{P}_{x,y}$.

Model Extraction. Specifically, an adversary can query a sample $\hat{x}_i$ from the surrogate dataset $\mathcal{D}_s$ to the model h_θ, obtain the output of the model $h_\theta(\hat{x}_i)$, and then minimize the KL-divergence between $\hat{h}_{\hat{\theta}}(\hat{x}_i)$ and $h_\theta(\hat{x}_i)$ as follows:

$$\begin{aligned} &\underset{\hat{\theta}}{\text{minimize}}\, \mathcal{L}_{\texttt{ext}}(\hat{\theta}; \theta, \mathcal{D}_s) \\ &\mathcal{L}_{\texttt{ext}}(\hat{\theta}; \theta, \mathcal{D}_s) = \frac{1}{|\mathcal{D}_s|} \sum_{(x,y)\in\mathcal{D}_s} D_{\text{KL}}(h_\theta(x), \hat{h}_{\hat{\theta}}(x)), \end{aligned} \tag{2}$$

which is a special case of knowledge distillation (Hinton et al., 2015). Although a pair of the sample $\hat{x}_i$ and its label $\hat{y}_i$ is drawn from joint distribution $\hat{\mathcal{P}}_{x,y}$ which is different from $\mathcal{P}_{x,y}$, the adversary can still steal functionality of the source model h_θ (Orekondy et al., 2019).

Ownership Identification with a Trigger Set. In order to detect and protect against intellectual property theft, one can embed a watermark into a model and identify the ownership of stolen models. One effective method for watermarking is the use of a trigger set. The owner of the model randomly samples $\{(x_{k_i}, y_{k_i})\}_{i=1}^m$ from $\mathcal{D}$ without replacement, then replaces the label y_{k_i} with a random label y'_{k_i} such that $y'_{k_i} \neq y_{k_i}$, resulting in the trigger set $\mathcal{D}_q = \{(x_{k_i}, y'_{k_i})\}_{i=1}^m$. Since the labels in the trigger set are not consistent with the original training dataset $\mathcal{D}$, the set $\{(x_{k_i}, y_{k_i})\}_{i=1}^m$ is excluded from $\mathcal{D}$, and the remaining set is denoted as $\mathcal{D}_b \coloneqq \mathcal{D} \setminus \{(x_{k_i}, y_{k_i})\}_{i=1}^m$ which we call benign training set. Finally, the owner forces the model h_θ to deliberately mis-classify x_{k_i} into y'_{k_i} by optimizing the parameters θ as follows:

$$\underset{\theta}{\text{minimize}}\, \mathcal{L}(\theta; \mathcal{D}_b) + \mathcal{L}(\theta; \mathcal{D}_q), \tag{3}$$

which we refer to as empirical risk minimization (ERM). If a prediction of a suspicious model on the trigger set $\mathcal{D}_q$ is sufficiently similar to that of h_θ, we can claim that the model is stolen and verify the ownership of the model.

However, the empirical risk minimization with $\mathcal{D}_b$ and $\mathcal{D}_q$ is still vulnerable to functionality stealing. In other words, the prediction for each sample x_{k_i} of the trigger set $\mathcal{D}_q$ made by the surrogate model is not sufficiently similar to the prediction of the source model. Since size of the trigger set is small and the label y'_{k_i} of the sample x_{k_i} from the trigger set $\mathcal{D}_q$ contradicts to the ground truth label y_{k_i}, the trigger set is considered as an outlier. If we train the source model on both training set $\mathcal{D}_b$ and the trigger set $\mathcal{D}_q$ with ERM, the samples from the trigger set are pushed away from the majority in the representation space as illustrated in Figure 2e and they are located close to the decision boundary as shown in Figure 2a. As a result, it is extremely difficult for the surrogate model to generalize to such outlier trigger set by learning from the source model output on the majorities. In the end, as shown in Figures 2b and 2f, the surrogate model mis-classifies the sample from the trigger set into its original ground truth label. In other words, it achieves 0 accuracy on trigger set and successfully removes the watermarking as shown in Table 1.

Table 1: Acc. of the source and surrogate model with ERM.

Model	CIFAR-10 Acc.	Trig. Acc.
Source	93.10	100.00
Surrogate	90.60	0.00

3.2. Margin-based watermarking

As previously described, since the way how we label the samples in the trigger set is not consistent with the training dataset $\mathcal{D}_b$, the trigger set is considered to be an outlier in the latent representation of the source model with ERM in Figure 2e. It makes the surrogate model to steal the functionality of the source model easier. In order to tackle these limitation of ERM of Equation (3), as shown in Figure 2c, we propose to maximize the margin of the sample x_{k_i} from the trigger set $\mathcal{D}_q$. Specifically, we perform projected gradient ascent (Madry et al., 2018) to identify the samples with maximum loss around the ϵ-neighborhood of the instances from the trigger set and train the source model h_θ to minimize loss on those samples as well as the loss on the training set $\mathcal{D}_b$ as follows:

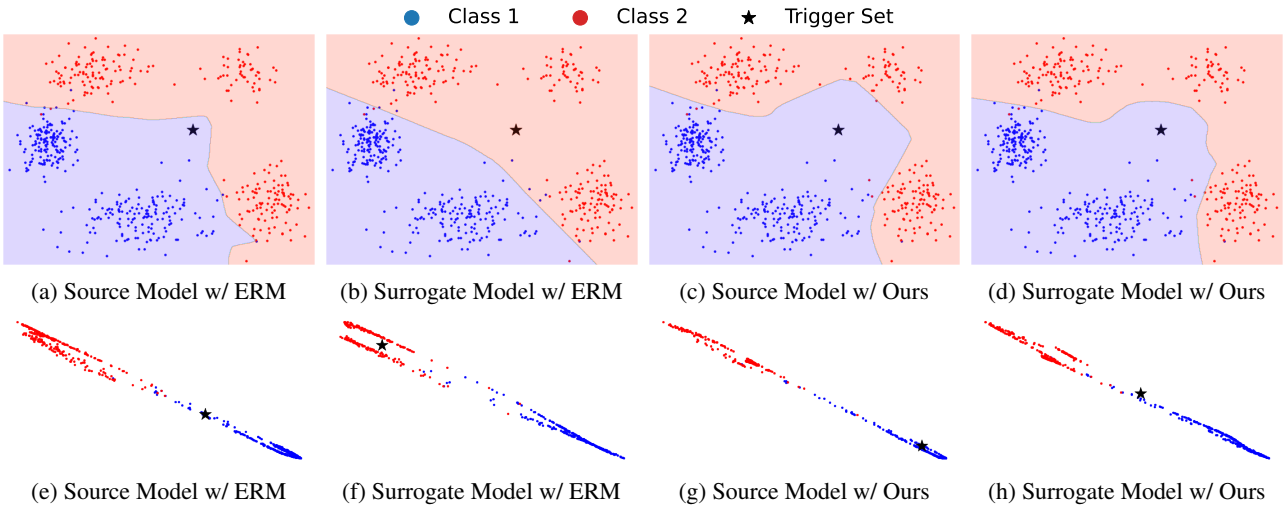
$$\begin{aligned} &\underset{\theta}{\text{minimize}}\, \mathcal{L}(\theta; \mathcal{D}_b) + \lambda \mathcal{L}_{\texttt{wat}}(\theta; \mathcal{D}_q) \\ &\mathcal{L}_{\texttt{wat}}(\theta; \mathcal{D}_q) = \frac{1}{|\mathcal{D}_q|} \sum_{(x,y')\in\mathcal{D}_q} \max_{\|\delta\|_\infty \leq \epsilon} \ell\left(h_\theta(x+\delta), y'\right). \end{aligned} \tag{4}$$

Consequently, as illustrated in Figure 2g, training the source model h_θ with the margin maximization pushes the trigger sample x_{k_i} with its label y'_{k_i} towards majorities of the samples x_j from $\mathcal{D}_b$ with its corresponding label $y_j = y'_{k_i}$, and the output of the source model on the training set $\mathcal{D}_b$ becomes similar to that of the source model on the trigger set $\mathcal{D}_q$. Then the surrogate model learns to replicate the output of the source model on the samples x_j with $y_j = y'_{k_i}$ in training set and it classifies the sample x_{k_i} into the class $y'_{k_i} = y_j$, *i.e.* the watermarking is still embedded in the surrogate model and fails to steal the functionality of the source model h_θ as shown in Figure 2d. We outline our method in Algorithm 1.

4. Experiments

4.1. Experimental Setup

We evaluate our margin-based watermarking method and the other baselines which demonstrate the robustness against



(a) Source Model w/ ERM (b) Surrogate Model w/ ERM (c) Source Model w/ Ours (d) Surrogate Model w/ Ours

(e) Source Model w/ ERM (f) Surrogate Model w/ ERM (g) Source Model w/ Ours (h) Surrogate Model w/ Ours

Figure 2: Visualization of input (the fist row) and hidden representation (the second row) for the synthetic experiments, where the sample (star) from the trigger set is labeled as the class 1 and its original ground truth label is the class 2. **(a) & (e)**: Trigger set is outlier in the source model with ERM and located near decision boundary. **(b) & (f)**: Surrogate model easily mis-classifies the trigger set since the trigger set is an outlier. **(c) & (g)**: As a consequence of margin-based watermarking, the trigger set is not an outlier anymore. **(d) & (h)**: The surrogate model makes a correct prediction about the trigger set as it learns to classify the samples around the trigger set.

Algorithm 1 Margin-based watermarking

Input: The training dataset $\mathcal{D}_t$, trigger set $\mathcal{D}_q$, model h_θ, learning rate η_1, loss function ℓ, step size for projected gradient ascent η_2, maximum bound for projected gradient ascent ϵ, # of iteration K.

Output: Watermarked model h_θ.

while not converge **do**
  Sample a mini-batch $(x_{b_i}, y_{b_i})_{i=1}^{n'} \sim \mathcal{D}_t$
  $L_{\texttt{obj}} \leftarrow \frac{1}{n'} \sum_{i=1}^{n'} \ell(h_\theta(x_{b_i}), y_{b_i})$, $L_{\texttt{wat}} \leftarrow 0$
  for all $i = 1, \ldots, m'$ **do**
    $(x_{\texttt{wat}}, y_{\texttt{wat}}) \sim \mathcal{D}_q$
    $\delta \leftarrow \mathbf{0} \in \mathcal{X}$
    for all $j = 1, \ldots, K$ **do**
      $\delta \leftarrow \delta + \eta_1 \cdot \texttt{sign}\left(\nabla_\delta \ell(h_\theta(x_{\texttt{wat}} + \delta), y_{\texttt{wat}}\right)$
      $\delta \leftarrow \texttt{Proj}(\delta, \epsilon)$
    end for
    $L_{\texttt{wat}} \mathrel{+}= \frac{1}{m'} \ell(h_\theta(x_{\texttt{wat}} + \delta), y_{\texttt{wat}})$
  end for
  $L \leftarrow L_{\texttt{obj}} + \lambda L_{\texttt{wat}}$
  $\theta \leftarrow \theta - \eta_2 \cdot \nabla_\theta L$
end while

functionality stealing methods.[1]

Stealing Methods. Following the previous papers (Jia et al., 2021; Bansal et al., 2022), we train a surrogate model $\hat{h}_{\hat{\theta}}$ on the dataset $\mathcal{D}_b$, *i.e.* $\mathcal{D}_s := \mathcal{D}_b$, with variants of model extraction methods to steal functionality of the model h_θ.

- **Soft-Label.** We train the surrogate model $\hat{h}_{\hat{\theta}}$ to minimizing $\mathcal{L}_{\texttt{ext}}(\hat{\theta}; \theta, \mathcal{D}_b)$ as described in Equation (2).
- **Hard-Label.** Instead of using the soft label $(z_1, \ldots, z_{d_y}) = h_\theta(x)$, we construct a hard label $\tilde{y} = \texttt{one\_hot}(\arg\max_i \{z_i\}_{i=1}^{d_y}) \in \mathbb{R}^{d_y}$ and replace the soft label with the hard label $\tilde{y}$ for the KL-divergence in Equation (2).
- **Regularization with Ground Truth Label.** As an aggressive stealing method, we propose to train the surrogate model $\hat{h}_{\hat{\theta}}$ to minimize empirical loss on the dataset $\mathcal{D}_b$ as well as KL-divergence between the output of the model h_θ and that of the $\hat{h}_{\hat{\theta}}$ as follows:

$$\underset{\hat{\theta}}{\text{minimize}}\, \alpha \mathcal{L}_{\texttt{ext}}(\hat{\theta}; \theta, \mathcal{D}_b) + (1 - \alpha)\mathcal{L}(\hat{\theta}; \mathcal{D}_b), \quad (5)$$

where $0 \leq \alpha \leq 1$ is a hyperparameter to control the importance the loss function. Since how we label the sample x_{k_i} from the trigger set $\mathcal{D}_q$ is not consistent with $\mathcal{D}_b$, *i.e.* its label y'_{k_i} contradicts to the ground truth label y_{k_i}, enforcing the surrogate model to predict ground truth label of x from $\mathcal{D}_b$ can effectively remove the watermarking embedded in the source model h_θ.

Baselines. We compare our margin-based watermarking method against the following baselines.

- **Entangled Watermark Embedding (EWE)** (Jia et al., 2021). It guides the source model by learning with the soft nearest neighbor loss for the trigger set.
- **Randomized Smoothing (RS)** (Bansal et al., 2022). It performs randomized smoothing for certified robustness under the limited size of the modification for the source model parameters θ.

[1] The source codes are available at here.

Table 2: Results for watermarking DNNs against functionality stealing methods. All the models have the trigger set with size 100. For the watermark accuracy of the standard model, the lower is better since the accuracy of the watermarking can have more statistical significance. The result shows that our method strictly outperforms the baselines for ownership verification.

Methods	Metric	Source Model (h_θ)	Surrogate Models ($\hat{h}_{\hat{\theta}}$)		
			Soft-Label	Hard-Label	Reg. w. GT Label
DI (Maini et al., 2021)	CIFAR-10 Acc. (%)	**92.03 ± 0.25**	92.50 ± 0.17	92.27 ± 0.38	92.23 ± 0.59
EEF (Li et al., 2022)		91.86 ± 0.22	87.75 ± 0.37	86.86 ± 0.61	86.04 ± 0.19
EWE (Jia et al., 2021)		86.10 ± 0.54	83.97 ± 1.02	82.22 ± 0.50	88.88 ± 0.35
RS (Bansal et al., 2022)		84.17 ± 1.01	88.93 ± 1.18	89.62 ± 0.97	90.14 ± 0.08
Margin-based (Ours)		87.81 ± 0.76	91.17 ± 0.76	91.88 ± 0.40	93.05 ± 0.20
DI (Maini et al., 2021)	p-value	10^{-3}	10^{-2}	10^{-2}	10^{-2}
EEF (Li et al., 2022)		10^{-7}	10^{-4}	10^{-4}	10^{-3}
Margin-based (Ours)		$\mathbf{10^{-12}}$	$\mathbf{10^{-8}}$	$\mathbf{10^{-8}}$	$\mathbf{10^{-8}}$
EWE (Jia et al., 2021)	Trigger Set Acc. (%)	26.88 ± 8.22	51.01 ± 5.58	36.05 ± 6.48	1.64 ± 1.05
RS (Bansal et al., 2022)		95.67 ± 4.93	7.67 ± 4.04	6.33 ± 1.15	3.00 ± 0.00
Margin-based (Ours)		**100.00 ± 0.00**	**82.00 ± 1.00**	**51.33 ± 4.93**	**72.67 ± 6.66**

Methods	Metric	Source Model (h_θ)	Surrogate Models ($\hat{h}_{\hat{\theta}}$)		
			Soft-Label	Hard-Label	Reg. w. GT Label
DI (Maini et al., 2021)	CIFAR-100 Acc. (%)	**70.97 ± 0.74**	72.70 ± 0.26	71.33 ± 0.31	72.87 ± 0.59
EWE (Jia et al., 2021)		55.11 ± 1.67	53.00 ± 1.57	46.78 ± 1.00	63.73 ± 0.40
RS (Bansal et al., 2022)		59.87 ± 2.78	65.66 ± 1.53	65.79 ± 0.39	64.99 ± 0.30
Margin-based (Ours)		62.13 ± 4.36	67.66 ± 0.36	70.65 ± 0.49	70.24 ± 0.46
DI (Maini et al., 2021)	p-value	10^{-3}	10^{-2}	10^{-2}	10^{-2}
Margin-based (Ours)		$\mathbf{10^{-10}}$	$\mathbf{10^{-7}}$	$\mathbf{10^{-6}}$	$\mathbf{10^{-6}}$
EWE (Jia et al., 2021)	Trigger Set Acc. (%)	68.14 ± 10.16	30.90 ± 11.34	15.10 ± 5.64	5.73 ± 3.42
RS (Bansal et al., 2022)		99.00 ± 1.73	2.67 ± 1.53	4.33 ± 4.16	2.00 ± 1.00
Margin-based (Ours)		**100.00 ± 0.00**	**70.67 ± 7.57**	**40.00 ± 8.89**	**62.66 ± 10.12**

Table 3: Accuracy of the model trained with CIFAR-10 on the trigger sets constructed by each method.

Trigger Set	EWE	RS	Ours
Trigger Acc	2.00 ± 2.00	1.67 ± 0.00	0.00 ± 0.00

- **Dataset Inference (DI)** (Maini et al., 2021). It measure a proxy margin between sample x from $\mathcal{D}_b$ and decision boundary of each class, resulting in $\mathbf{c} \in \mathbb{R}^{d_y}$, and leverage the margin $\mathbf{c}$ to represent x as an embedding. Then it extracts the margin embedding $\mathbf{c}$ from unseen dataset $\mathcal{D}_{\texttt{test}}$ and train a binary meta classifier that distinguishes the margin embedding of D_b from that of the unseen dataset. After that, it extracts margin embedding of the suspicious model from $\mathcal{D}_b$ and $\mathcal{D}_{\texttt{test}}$ and perform t-test between the confidence score of the classifier on the margin embedding from $\mathcal{D}_b$ and $\mathcal{D}_{\texttt{test}}$ to verify the suspicious model is the counterfeit of the source model.
- **Embedded External Features (EFF)** (Li et al., 2022). It samples pairs of samples and labels from $\mathcal{D}_b$ and applies semantic preserving transformations to the samples while using the same label to construct the set $\mathcal{D}_t$. Then it trains a source model on $\mathcal{D}_b \cup \mathcal{D}_t$ and a benign model on $\mathcal{D}_b$. Gradient w.r.t both model parameter on each sample from $\mathcal{D}_t$ is computed and a binary meta classifier is trained to distinguish gradient of the source model from that of the benign model. Finally, it performs t-test between gradient of suspicious model on $\mathcal{D}_t$ and that of the source model to verify the ownership of the suspicious model.

Evaluation Metric. For the watermarking based approaches — EWE and RS, we measure accuracy of the surrogate model $\hat{h}_{\hat{\theta}}$ on the trigger set $\mathcal{D}_q$ to evaluate the performance of watermarking. For the statistical testing based methods — DI and EEF, we compute p-value from the t-test performed by each method.

Settings We use the CIFAR-10 or CIFAR-100 dataset (Krizhevsky et al., 2009) as $\mathcal{D}$ and split the training set of the CIFAR-10 dataset into a train set $\mathcal{D}_b$, and

Table 4: Watermarking against adversary with different surrogate dataset and architecture.

Model	CIFAR-10 Acc.	Trigger Set Acc.
Surrogate Dataset with SVHN		
Source	87.81 ± 0.76	100.0 ± 0.00
Surrogate	63.99 ± 3.90	72.00 ± 6.08
Surrogate Model with VGG11		
Source	87.81 ± 0.76	100.0 ± 0.00
Surrogate	86.00 ± 2.17	32.00 ± 7.21

a validation set. We *randomly sample* 100 images from the validation set and assign random labels to them such that the new label is different from their original ground truth label, resulting in the trigger set $\mathcal{D}_q$. ResNet34 (He et al., 2016) is used for all the source and surrogate models. Note that all the surrogate models are randomly initialized since adversaries cannot access to the parameters of the source model. As a functionality stealing, we use the training dataset $\mathcal{D}_b$ as the surrogate dataset $\mathcal{D}_s$ for training the surrogate model to imitate the source model since it is the *de facto* strongest attack for diminishing the watermark or confusing the ownership verification procedure (Jia et al., 2021).

Hyperaparemters For our margin-based watermarking method, we set the trigger set size and batch size of the trigger set to 100 and 25, respectively. We perform inner maximization in Equation (4) for 5 steps with a step size $\eta_1 = 1/255$ and $\epsilon = 5/255$. For attack with regularized by ground truth label Equation (5), we set α to 0.3. We run 3 multiple experiments with a different random seed ,and report the mean and standard deviation for accuracy and harmonic mean for p-values.

4.2. Results on Functionality Stealing

In Table 2, we present our experimental result on CIFAR-10 and CIFAR-100 dataset. Notably, source model trained with our margin-based watermarking method achieves perfect accuracy on the trigger set, which is one of the most important goals for watermarking since we identify the ownership of the source model by the accuracy on the trigger set. In contrast, all the baselines make some error on the trigger set and the vanilla model trained without any ownership verification method consistently shows 0 accuracy across all the trigger sets as shown in Table 3.

Our margin-based watermarking is the most effective owner verification method compared to other baselines. Firstly, all the surrogate models trained with soft-label, hard-label, or Reg. w. GT label fail to remove our margin-based watermarking and thus they show significantly high accuracy on the trigger set. In contrast, the other watermarking baselines fail to prevent surrogate models from stealing the functionality of the source model and thus surrogate models show low accuracy on the trigger set. Moreover, the source model trained with our margin-based watermarking achieves the best accuracy on CIFAR-10 / CIFAR-100 compared to the source models trained with other watermarking baselines. Based on these results, our method achieve the best trade-off between accuracy on CIFAR-10 / CIFAR-100 and the trigger set among the watermarking baselines EWE and RS.

Lastly, our method also outperforms other non-watermarking baselines DI and EEF in terms of statistical testing. The source model trained with our margin-based watermarking achieves the lowest p-value compared to other baselines. Moreover, all the surrogate models fail to diminish the watermarking of our source model and show an extremely small p-value, which means it is highly likely that the surrogate model is counterfeit of the source model.

4.3. Heterogeneous Surrogate Dataset & Architecture

In this section, we assume that either the training dataset $\mathcal{D}_b$ or neural network architecture of the source model h_θ is *unknown* to adversaries, which is a more realistic and challenging scenario than the previous experimental setup. First, we utilize SVHN dataset (Netzer et al., 2011) as a surrogate dataset $\mathcal{D}_s$ to train a surrogate model with the soft-label (Equation 2) of the source model which is trained with our margin-based watermarking. Since SVHN is dissimilar to CIFAR-10 dataset, distilling the knowledge of the source model into the surrogate model is imperfect. Consequently, it degrades the accuracy of the surrogate model on both CIFAR-10 dataset and trigger set as shown in Table 4. Despite of the performance degradation, our margin-based watermarking method still outperforms the other baselines evaluated at much easier tasks, where trigger accuracy of EWE and RS is 51.05 and 7.67, respectively.

Additionally, we replace the neural network architecture of the surrogate model with VGG11 (Simonyan & Zisserman, 2015) and train the surrogate model with CIFAR-10 dataset as a surrogate dataset, *i.e.* $\mathcal{D}_s = \mathcal{D}_b$, to imitate the soft-label of the source model trained with our margin-based watermarking. Since we train the surrogate model with the same dataset used for training the source model, as shown in Table 4, the surrogate model can easily achieve as the same accuracy as the source model on CIFAR-10. However, the accuracy of the surrogate model on the trigger set drops to 32.00. Since its architecture is different from that of the source model, it is challenging for the surrogate model to generalize to the trigger set.

4.4. Labeling Strategy for Trigger Set

In this section, we show that our margin-based watermarking method is robust to how we construct the trigger set. Our method allows owners to choose arbitrary samples from

Table 5: Upper bounds of margin for the source model and its surrogates. The sufficient margin induces the high trigger set accuracy of surrogates and also the margin of the source model is transferred to surrogates. We treat the maximum and the minimum intensity of the CIFAR-10 as 1 and 0.

Model	Source (h_θ)	Surrogates		
		Soft-Label	Hard-Label	Reg. w. GT Label
Upper bound of Margin	0.0232 ± 0.0011	0.0094 ± 0.0006	0.0078 ± 0.0008	0.0080 ± 0.0005

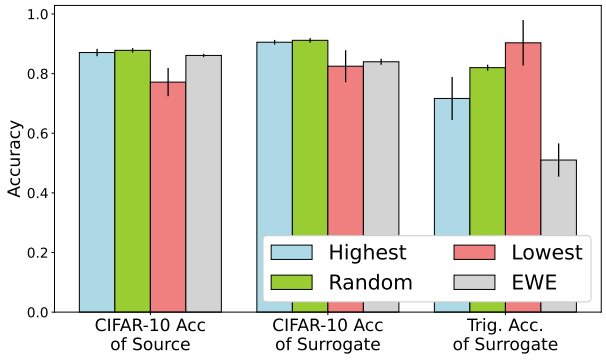


Figure 3: Accuracy of the source model and the surrogate model on CIFAR-10, and accuracy of the surrogate model on trigger sets constructed by different methods (highest, lowest, and random). Our margin-based watermarking method with any type of trigger set outperforms the baseline EWE.

the given CIFAR-10 dataset $\mathcal{D}$ and can assign any labels to the samples other than ground truth labels. This randomness is critical for protecting the ownership of the model since it is difficult for adversaries to identify such random trigger set and remove the watermarking.

Firstly, we randomly samples 100 instances from the whole dataset $\mathcal{D}$ and arbitrarily choose a label out of 9 excluding the ground truth label for each sample. This results in 9^{100} possible different trigger sets for the given 100 samples. We repeat this process to construct multiple trigger sets and train the source model with each trigger set. As shown in Figure 3, our margin-based watermarking method shows consistent performance across all the trigger sets.

Lastly, we consider the following two extreme cases for labeling the trigger set. Given the source model trained on CIFAR-10, we compute the confidence score of the model on the sample from the trigger set and either choose the label with the *highest* or *lowest* confidence among nine excluding the ground truth label. As shown in Figure 3, our margin based watermarking with such extreme trigger set still significantly achieves better trigger set accuracy than the baseline EWE. Note that the source model trained with our margin-based watermarking achieves perfect accuracy on both trigger sets.

4.5. Upper Bound of Margin

In this section, we empirically verify that the surrogate model partially copy the margin of the source model. In order to measure the upper bound of the margin of each sample from the trigger set, we perform projected gradient ascent while varying the maximum margin bound as an approximation (Jia et al., 2021). As shown in Table 5, we observe that the surrogate model still keeps margin for some samples. Moreover, high margin is correlated with high trigger set accuracy which validates our margin-based watermarking method.

5. Conclusion

In this work, we proposed margin-based watermarking to protect intellectual property rights. The margin-based watermarking trains the trigger set with a sufficient margin and achieves the robust decision boundary of the each sample of the trigger set. We validate the margin-based watermarking against the functionality stealing method for the most competitive settings. For all the cases, the margin-based watermarking outperforms the baseline watermarking and ownership verification methods. Moreover, our method ensures to use any samples and any labels for the trigger set which prevents adversaries aim from identifying the trigger set and circumventing the ownership identification. Lastly, we show that the margin is transferred to surrogate models with quantitative and qualitative analysis. Our results suggest that there exist trade-offs between watermarking and the performance of the model, and we aim to seek a better method for improving both of them as a future work.

Acknowledgements

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2019-0-00075, Artificial Intelligence Graduate School Program(KAIST)) and Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2020-0-00153).

References

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

Baevski, A., Zhou, Y., Mohamed, A., and Auli, M. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33, 2020.

Bansal, A., Chiang, P.-y., Curry, M. J., Jain, R., Wigington, C., Manjunatha, V., Dickerson, J. P., and Goldstein, T. Certified neural network watermarks with randomized smoothing. In *International Conference on Machine Learning*, pp. 1450–1465. PMLR, 2022.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., 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., 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 Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020.

Chen, H., Rouhani, B. D., Fu, C., Zhao, J., and Koushanfar, F. DeepMarks: A secure fingerprinting framework for digital rights management of deep learning models. In *Proceedings of the ACM International Conference on Multimedia Retrieval*, pp. 105–113, 2019.

Chen, X., Chen, T., Zhang, Z., and Wang, Z. You are caught stealing my winning lottery ticket! making a lottery ticket claim its ownership. In *Proceedings of the Advances in Neural Information Processing Systems*, 2021.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Jia, H., Choquette-Choo, C. A., Chandrasekaran, V., and Papernot, N. Entangled watermarks as a defense against model extraction. In *Proceedings of the USENIX Security Symposium*, pp. 1937–1954, 2021.

Krizhevsky, A. et al. Learning multiple layers of features from tiny images. *arXiv preprint*, 2009.

Lee, S., Song, W., Jana, S., Cha, M., and Son, S. Evaluating the robustness of trigger set-based watermarks embedded in deep neural networks. *IEEE Transactions on Dependable and Secure Computing*, 2022.

Li, Y., Zhu, L., Jia, X., Jiang, Y., Xia, S.-T., and Cao, X. Defending against model stealing via verifying embedded external features. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

Li, Z., Hu, C., Zhang, Y., and Guo, S. How to prove your model belongs to you: A blind-watermark based framework to protect intellectual property of DNN. In *Proceedings of the Annual Computer Security Applications Conference*, pp. 126–137, 2019.

Liu, K., Dolan-Gavitt, B., and Garg, S. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pp. 273–294. Springer, 2018.

Liu, W., Wang, X., Owens, J. D., and Li, Y. Energy-based out-of-distribution detection. *arXiv preprint arXiv:2010.03759*, 2020.

Lopes, R. G., Fenu, S., and Starner, T. Data-free knowledge distillation for deep neural networks. *arXiv preprint arXiv:1710.07535*, 2017.

Lukas, N., Zhang, Y., and Kerschbaum, F. Deep neural network fingerprinting by conferrable adversarial examples. In *International Conference on Learning Representations*, 2020.

Lukas, N., Jiang, E., Li, X., and Kerschbaum, F. SoK: How robust is image classification deep neural network watermarking? In *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 52–69, 2022.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Maini, P., Yaghini, M., and Papernot, N. Dataset Inference: Ownership resolution in machine learning. In *Proceedings of the International Conference on Learning Representations*, 2021.

Merrer, E. L., Pérez, P., and Trédan, G. Adversarial frontier stitching for remote neural network watermarking. *Neural Computing and Applications*, 2019.

Namba, R. and Sakuma, J. Robust watermarking of neural network with exponential weighting. In *Proceedings of the ACM Asia Conference on Computer and Communications Security*, pp. 228–240, 2019.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Orekondy, T., Schiele, B., and Fritz, M. Knockoff nets: Stealing functionality of black-box models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4954–4963, 2019.

Rouhani, B. D., Chen, H., and Koushanfar, F. DeepSigns: An end-to-end watermarking framework for ownership protection of deep neural networks. In *Proceedings of the ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 485–497, 2019.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015.

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

Yang, P., Lao, Y., and Li, P. Robust watermarking for deep neural networks via bi-level optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14841–14850, 2021.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Zhang, J., Gu, Z., Jang, J., Wu, H., Stoecklin, M. P., Huang, H., and Molloy, I. Protecting intellectual property of deep neural networks with watermarking. In *Proceedings of the ACM Asia Conference on Computer and Communications Security*, pp. 159–172, 2018.

Zhang, J., Chen, D., Liao, J., Fang, H., Zhang, W., Zhou, W., Cui, H., and Yu, N. Model watermarking for image processing networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020a.

Zhang, J., Chen, D., Liao, J., Zhang, W., Hua, G., and Yu, N. Passport-aware normalization for deep model protection. In *Proceedings of the Advances in Neural Information Processing Systems*, pp. 22619–22628, 2020b.

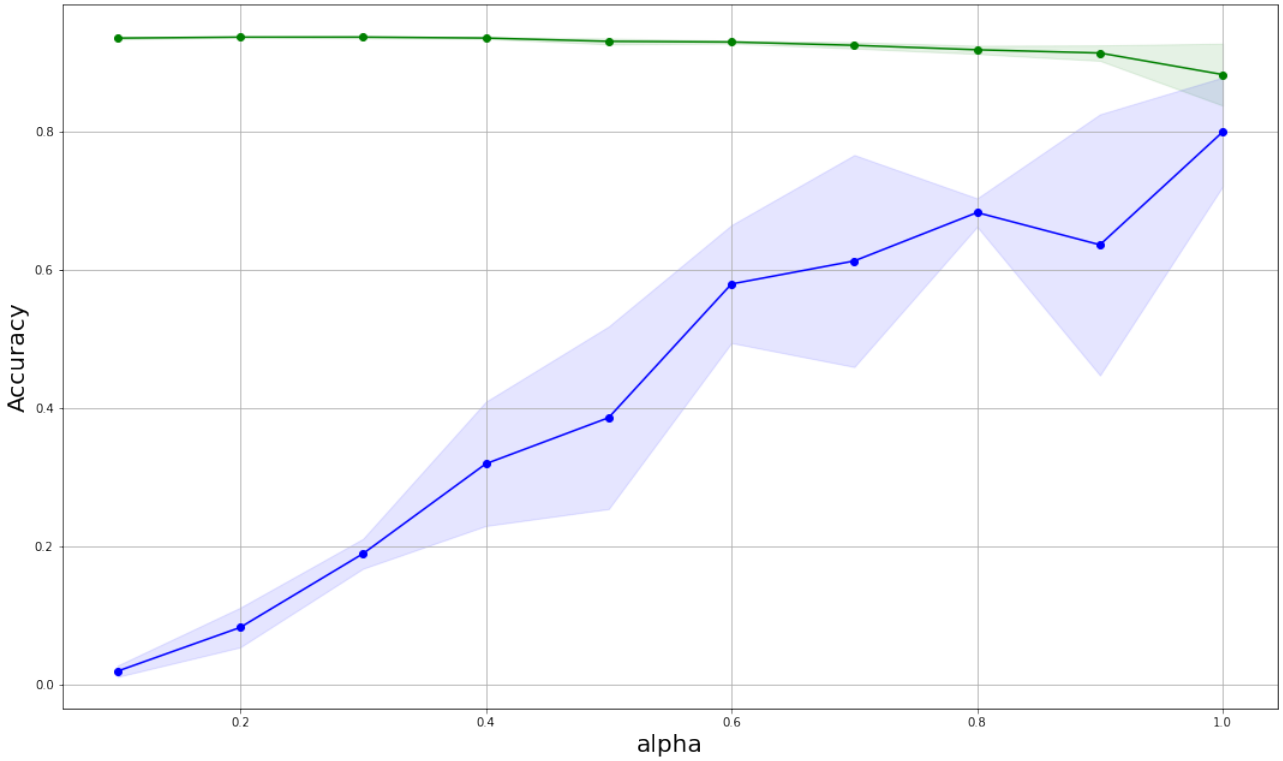


Figure 4: CIFAR-10 (green) and trigger set (blue) accuracies of the surrogate model from distillation while varying α.

A. Summary

The appendix is constructed as follows: we provide experimental settings, the meaning of the distillation for evaluating the watermarking method, the result of the choice of the query and statistical significance comparison in Section B, visualization of the margin-based watermarking in Section C and additional threat models in Section D.

B. Additional Analysis

B.1. Experimental settings

Here we produce the detailed experimental settings. We use ResNet-34 for all the watermarked model with SGD optimizer, learning rate of 0.1, weight decay of 0.0001, learning rate decay of 0.1 for 100 and 150 epoch. The training was done for 200 epochs. For distillation and extraction, we used the same optimizer of training the source model. When varying the number of iterations K in Algorithm 1 and Equation 3, we also vary the maximum perturbation size ϵ, for instance, when $K = 3$, $\epsilon = 3/255$. For baseline experiments, we perform all the experiments with the authors' source codes. The source codes are available at: https://github.com/matbambbang/margin-based-watermarking.

B.2. Distillation for evaluating the watermarking method

The purpose of distillation is to distill the knowledge from the high-performing teacher model to the small student model. Thus we can see the distillation as functionality stealing since it leverages the student model to imitate the output of the teacher model for a given input. There exist various approaches to distill from the teacher model such as data-free distillation (Lopes et al., 2017), but we used the basic knowledge distillation method introduced in (Hinton et al., 2015). From the distribution introduced in (Hinton et al., 2015), the dataset for distillation also needs to be sampled from the same data distribution of the teacher model. Thus, in practice, the adversary cannot perform the distillation to diminish the watermarking since the adversary cannot have a corresponding label for each sample. However, due to the distinction between the label from the trigger set and the training set, the distillation can be used for stronger threats with the proper

choice of hyperparameters. The watermarking methods using a trigger set mainly focus on the response or prediction of the model, where the response should be distinguishable from the common models so that the victim can verify whether the suspicious model steals the source model. In order to eliminate the watermark, we can steal the functionality by using the source dataset of the watermarked model to produce the response of the query as same as the common models. We empirically show how to achieve the aforementioned statement by using distillation.

The training objective of distillation used in our experiment is given by,

$$\min_{\hat{\theta}} \mathbb{E}_{(x,y)\sim\mathcal{D}}\Big[(1-\alpha)\cdot \mathbf{CE}\big(\sigma(\mathbf{z}_s), y\big) + \alpha \cdot \mathbf{KL}\big(\sigma(\mathbf{z}_s/T), \sigma(\mathbf{z}_t/T)\big)\Big], \tag{6}$$

where σ is the softmax function, $\mathbf{z}_s, \mathbf{z}_t$ indicates the logit of the student and teacher model and α is a hyperparameter. We used $\alpha = 0.7$ to evaluate the watermarking in Table 2 which is also used in (Hinton et al., 2015). Intuitively, increasing the ratio of the cross-entropy term in Equation 6 can produce a model which may be more similar to the model without any watermarking. To verify this, we perform the distillation for our watermarking method while varying α. The results are shown in Figure 4.

The low α suppresses the knowledge transfer from the watermarked model to the surrogate model and the watermark accuracy drops. Even though there is no targeted elimination method for the watermark, however, distillation is still effective for erasing the watermark. In conclusion, the distillation used in our experiment can be used as one benchmark to evaluate the watermarking method.

Mixup Images	automobile, airplane	bird, horse	frog, truck	cat, truck	ship, truck
Target Label	ship	cat	bird	deer	frog

Figure 5: The examples of the Mixup query with two Mixup components for a trigger set.

B.3. The effect of the choice of the trigger set

As we have demonstrated before, our method does not have any constraints on the choice of the trigger set, except that they should not be contained in the training set $\mathcal{D}$. In the previous experiments, we choose the samples randomly and assign random labels to them, that are different from their ground truth labels. Further, we perform the experiments of the Mixup (Zhang et al., 2017) queries, where one Mixup query is the average of the Mixup components and the corresponding label is randomly selected except the true label of the Mixup components, see Figure 5. We set the number of Mixup queries to 100 and the number of Mixup components of 2 and 5. Since one Mixup query contains multiple correlations of the Mixup components, the false correlation of the trigger set might be stronger than of our previous settings.

Even with the stronger false correlation with the original objective, the result in Table 6 shows the Mixup query can be used for watermarking. However, when increasing the number of Mixup components for each query, the performance of the original objective decreases due to the stronger false correlation of the trigger set. The experiment shows that margin-based watermarking can be used for any given query, but the method with strong false correlation cannot guarantee the performance on the original objective.

Table 6: Comparison of the number of Mixup queries.

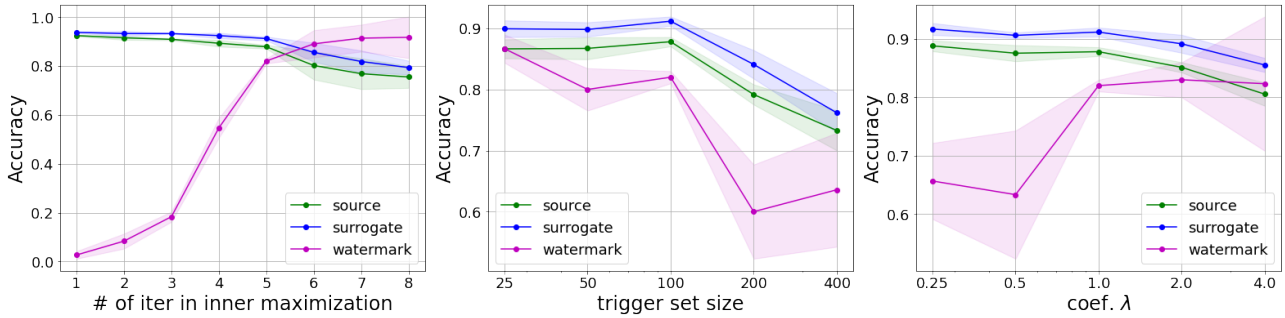
# of Mixup		2	5
Source Model	CIFAR-10. Acc.	79.28	44.80
	Trigger. Acc.	**100.00**	**100.00**
Distillation	CIFAR-10. Acc.	90.30	89.76
	Trigger. Acc.	**42.00**	10.00
Extraction (CIFAR10)	CIFAR-10. Acc.	80.06	44.40
	Trigger. Acc.	67.00	**83.00**
Extraction (CIFAR100)	CIFAR-10. Acc.	75.24	39.66
	Trigger. Acc.	**89.00**	85.00

B.4. In-distribution Analysis

To prevent one scenario in which the adversary rejects the queries to reject the ownership verification process, we analyze the additional quantity: in-distribution analysis. Suppose one watermarking method achieves reasonable performance for ownership verification. However, if the adversary can reject the querying process beforehand, the method cannot be applied to verify the ownership. One simple rejection procedure for the adversary is to confirm whether the given query is the in-distribution sample for the surrogate model or not. We analyze if the surrogate models from the watermarked model confirm the query as the in-distribution sample or not.

Table 7: In-distribution analysis by using energy-based OOD detection. The energy of the success queries are even lower than of the samples from the original objective. This means that the queries of the trigger set is probable to be treated as the in-distribution samples.

Energy	CIFAR-10 Acc.	Energy of $\mathcal{D}$	Success queries in $\mathcal{D}_q$	Failed queries in $\mathcal{D}_q$
Source Model	0.8414	-7.7472	**-15.1316**	N/A
Reg. w/ GT Label	0.9200	**-7.8103**	-7.2842	-6.9181
soft label w/ same source	0.8662	-7.2169	**-7.3657**	-6.2308
soft label w/ diff. source	0.8010	-6.7147	**-8.0391**	-6.1190



(a) Margin size vs Acc. (b) Trigger set size vs Acc. (c) Coefficent of $\mathcal{L}_{\texttt{wat}}$ vs Acc.

Figure 6: Accuracy of as a function of hyperparameters. The green line indicates the accuracy of the source model on CIFAR-10, the blue line shows the accuracy of the surrogate model on CIFAR-10, and the pink line denotes the accuracy of the surrogate model on the trigger set.

To check the query which is an in-distribution sample or not, we use the energy-based out-of-distribution detection method (Liu et al., 2020). In a nutshell, the lower energy indicates that the sample is probable to the in-distribution sample and the other is not. The results are shown in Table 7. The results show that the energy of the success queries in $\mathcal{D}_q$ have a lower energy than of the samples from the original objective $\mathcal{D}$. Thus, the queries would be considered as the in-distribution samples and the adversary cannot reject the queries by OOD sample rejection procedure.

B.5. Further Analysis

In this section, we empirically analyze how our margin-based watermarking leads to the surrogate which shows high accuracy on the trigger set as varying the hyperparameters — the margin size, the number of the trigger set, and the coefficient λ of $\mathcal{L}_{\texttt{wat}}$ in eq. 4. We present the experimental result in Figure 6, where the green line indicates the accuracy of the source model on CIFAR-10 $\mathcal{D}_b$, the blue line shows the accuracy of the surrogate model on CIFAR-10, and the pink line denotes the accuracy of the surrogate model on the trigger set. Note that we do not present the trigger set accuracy of our source model with margin-based watermarking since it always achieves perfect accuracy on the trigger set.

Margin size The size of the margin is an important hyperparameter of our method. We hypothesize that a larger margin leads to better accuracy of the surrogate model on the trigger set while it degrades the performance of the source model on the CIFAR-10 dataset $\mathcal{D}_b$. In order to verify the hypothesis, we measure the accuracy of the source model and surrogate model as varying the margin size. Specifically, we change the number of iterations K from 1 to 8.

As shown in Figure 6a, training a source model with small margin results in the surrogate model with low trigger set accuracy while the small margin leads to better accuracy of the source model on CIFAR-10 dataset $\mathcal{D}_b$. On the other hand, if we train the source model with a large number of iterations K, the surrogate model cannot remove watermarking and shows higher trigger set accuracy. Although we perform more inner maximization step ($K = 8$), the surrogate model cannot obtain perfect accuracy on the trigger set since it is hard for the source model to assign a sufficiently large margin to every sample in the trigger set.

Size of the trigger set In this experiment, we show how the size of the trigger set affects to the accuracy on the CIFAR-10 dataset $\mathcal{D}_b$ and the trigger set $\mathcal{D}_q$. We hypothesize that it is hard for the source model with fixed size model capacity to minimize the loss on the dataset $\mathcal{D}_b$ along with a large trigger set since the trigger set is an outlier compared to the dataset

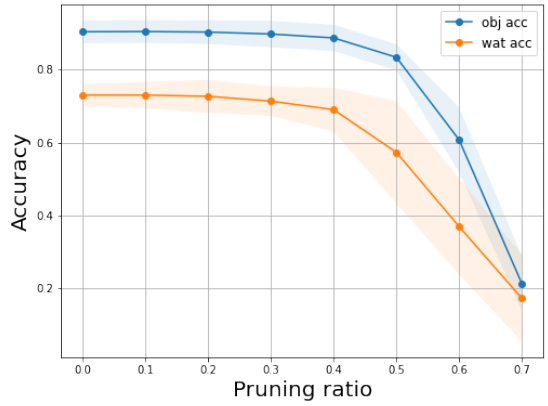


Figure 8: Results against pruning attack. Even with pruning, the stolen model still maintain the watermark accuracy. When the watermark accuracy largely drops, the objective accuracy also largely drops.

Table 8: Results of Equation 7. Since the extraction covers various types of perturbation δ, the stolen model captures the decision boundary of watermarked model more precisely, and finally achieves superior watermark accuracy.

Method		Ours
Source Model	CIFAR-10 Acc.	89.47 ± 0.66
	Trigger. Acc.	$\mathbf{100.00 \pm 0.00}$
Extraction (margin)	CIFAR-10 Acc.	84.94 ± 0.48
	Trigger. Acc.	$\mathbf{98.33 \pm 0.58}$

$\mathcal{D}_b$. As shown in Figure 6b, if we use a moderate size (from 25 to 100) of the trigger set for watermarking, the source model achieves high accuracy on the CIFAR-10 dataset $\mathcal{D}_b$. On the other hand, training the source model with a trigger set of size more than 100 degrades the performance on CIFAR-10. Since how we label the samples from the trigger set contradicts a pair of samples and label in the dataset $\mathcal{D}_b$ drawn from $\mathcal{P}_{x,y}$, the source model fails to generalize to test set drawn from $\mathcal{P}_{x,y}$ if it starts to focus on the trigger set.

Coefficient of the Watermarking Objective Lastly, we control the coefficient λ as 0.25 to 4, and the results are shown in Figure 6c. For the margin-based watermarking with the small λ, the surrogate model shows the lower trigger set accuracy since the watermarking objective is less considered during training. While increasing λ, the source model achieves a sufficient margin for the trigger set and leads the surrogate model to achieve high trigger set accuracy. However, the bigger λ makes the model pay less effort for training data, which results in the lower CIFAR-10 accuracy of the source model. Also, the larger λ shows a high variance of the trigger set accuracy for the surrogate model which is also shown in the bigger margin size cases that the model failed to get an excessive margin for every sample in the trigger set.

Overall, the margin-based watermarking shows the trade-off between the trigger set accuracy of the surrogate model and the CIFAR-10 accuracy of the source model, since the trigger set is composed of contradictions compared to the training data and that makes the learning hard to get excessive margin for each sample of the trigger set. Because of the contradiction, the trade-off is inevitable. However, the margin-based watermarking with a reasonable margin size and the size of the trigger set produces strong evidence for the ownership claims via the high trigger set accuracy of the surrogate model.

C. t-SNE visualization

We provide the additional t-SNE visualization in Figure 7 with perplexity of 10 where the figure does not represent the margin area. In Figure 7, we perform t-SNE for two different perplexities with 5 (on the top of the figures) and 10 (on the bottom of the figures).

D. Additional Threat Model

Since we discuss about the adversary with the functionality stealing attacks, distillation and extraction, we additionally evaluate our method on the other attacks.

Table 9: Results on self fine-tuning. All the surrogate model shows high trigger set accuracy, which is sufficient for ownership claim.

Model	Source Model	Fine-tuned Model
CIFAR-10 Acc.	87.81 ± 0.76	90.64 ± 0.34
Trig. Acc.	100.00 ± 0.00	82.33 ± 2.52

D.1. Pruning

Pruning the deep neural networks is to obtain the small neural network which has similar performance of the original model. Since pruning neglects sort of parameters, the watermarking can also be neglected and can decrease the watermark accuracy (Liu et al., 2018). To investigate the effect of the pruning, we prune the extracted model where the pruning is done for less activated neurons. Figure 8 shows the objective accuracy and the watermark accuracy with respect to the pruning ratio. The results show that the pruning cannot effectively diminish the watermark, and when the pruning becomes effective, the performance for the original task largely drops even the stolen model cannot be applicable. The results show that our margin-based watermarking is still effective against pruning attacks.

D.2. Fine-tuning

Fine-tuning the source model itself is also known for diminishing the watermark. We perform the experiments with fine-tuning for 100 epochs, learning rate of 0.1, and the results are shown in Table 9. Even we give excessive margin for the trigger set, some of samples failed to achive the sufficient margin. Thus, during fine-tuning, the samples with less margin drops and some of the trigger set still shows the high trigger set accuracy for ownership claim.

D.3. Extraction with margin

Our margin-based watermarking gives margin to the trigger set solely, so we observe the new type of extraction, the extraction with a margin which is given by,

$$\min_{\hat{\theta}} \mathbb{E}_{(\tilde{x},\tilde{y})\sim\tilde{\mathcal{D}}}\left[\tilde{\ell}\big(\hat{h}_{\hat{\theta}}(\tilde{x}+\tilde{\delta}), h_\theta(\tilde{x}+\tilde{\delta})\big)\right], \quad \tilde{\delta} = \arg\max_{\|\delta\|\leq\epsilon} \tilde{\ell}(h_\theta(\tilde{x}+\delta), h_\theta(\tilde{x})). \tag{7}$$

The stolen model with the extraction of Equation 7 can achieve more similar decision boundary because the existence of δ makes extraction to experience more various input images and its corresponding predictions. Table 8 shows the results, and surprisingly the stolen model achieves near perfect watermark accuracy. This also supports our claim, that the functionality stealing attack imitates the decision boundary of watermarked model, and more precise functionality stealing can copy most of the watermark.

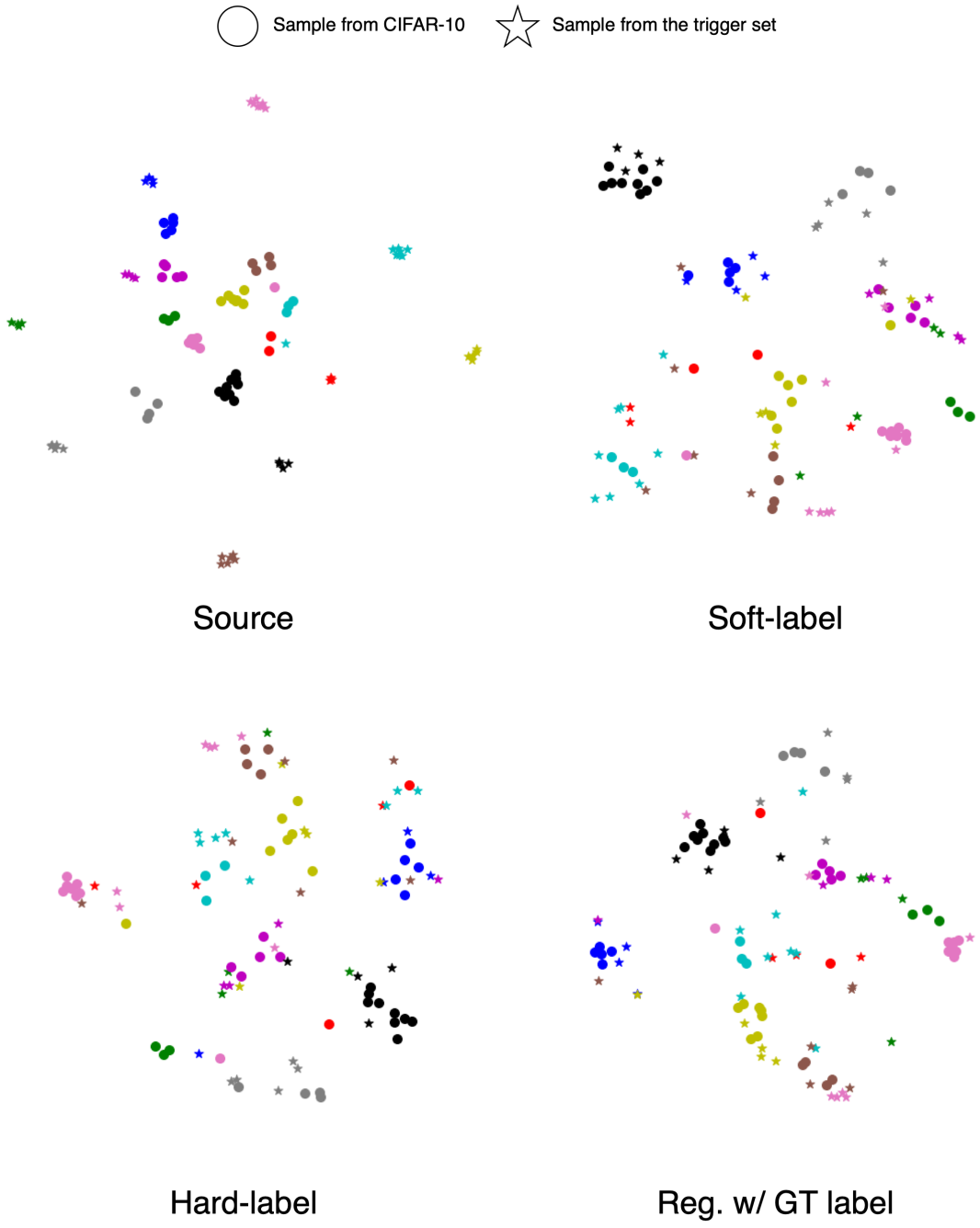


Figure 7: t-SNE visualization of the watermarked model and the surrogate models for CIFAR10.