MARGIN-BASED NEURAL NETWORK WATERMARKING

Anonymous authors
Paper under double-blind review

ABSTRACT

As Machine Learning as a Service (MLaaS) platforms become prevalent, deep neural network (DNN) watermarking has gained 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 recent 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 the deep neural network (DNN) models in real-world artificial intelligence systems. 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.

Considering that the MLaaS providers put a significant amount of resources for constructing a high-performing model, their intellectual property rights should be protected. 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., 2021b; Yang et al., 2021; Zhang et al., 2018; Adi et al., 2018; Fan et al., 2019) 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 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 approach (Adi et al., 2018; Zhang et al., 2018; Fan et al., 2019; Li et al., 2019; Namba & Sakuma, 2019; Zhang et al., 2020a;b; Chen et al., 2021b; Yang et al., 2021; Jia et al., 2021; Maini et al., 2021; Li et al., 2022), which operates in a black-box setting. For ownership verification, the model owner conducts statistical testing to demonstrate the behavioral difference between the watermarked and watermark-free models with a predefined set of samples whose labels are 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 to the attacker's model. Although several recent studies (Jia et al., 2021; Maini et al., 2021; Li et al., 2022) have shown their robustness

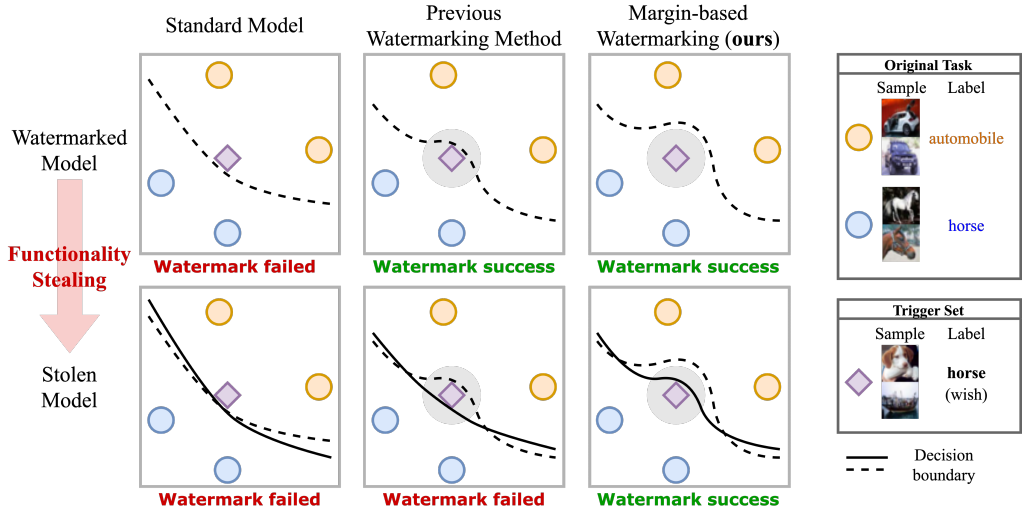


Figure 1: **Concept.** The circle and rhombus shapes indicate samples from the original objective and the trigger set for watermarking. The dotted line shows the decision boundary of the standard or the watermarked model and the solid line shows the decision boundary of the surrogate model, which obtained by the functionality stealing methods. The figure represents that margins of each trigger sample contributes to watermarked model to becoming robust against a functionality stealing attack.

against model extraction, we believe that the behavioral differences between the stolen model and the clean model were insufficient for ownership verification. The watermarking objective of a model is orthogonal to its original objective 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 the 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 partially copies the margin (see Figure 1). Our margin-based watermarking method outperforms the previous baseline on watermarking verification tasks against extraction and distillation attacks, where the latter is the strongest attack that assumes knowledge of the original objective. Last but not least, our method allows using any kind of the trigger sets as long as the set has distinguishable characteristics compared to the original task.

Our contributions are threefold:

- We propose a margin-based watermarking method that aims to maximize the margins of the in-distribution trigger set to mitigate functionality stealing attacks.
- We validate our margin-based watermarking method against two functionality stealing methods (*i.e.*, extraction and distillation), showing that it achieves the state-of-the-art watermarking performance.
- We show that the model trained by margin-based watermarking transfers the margin to the surrogate models.

2 Related Work

2.1 Watermarking Deep Neural Networks

DNN watermarking algorithms can be broadly categorized into feature-based (Uchida et al., 2017; Chen et al., 2019; Rouhani et al., 2019) and trigger set-based methods (Adi et al., 2018; Zhang et al., 2018; Fan et al., 2019; Li et al., 2019; Namba & Sakuma, 2019; Zhang et al., 2020a;b; Chen et al., 2021b; Yang et al., 2021; Jia et al., 2021). However, a vast majority of DNN watermarking takes a

trigger set-based approach, as feature-based watermarking assumes a white-box scenario where we have access to the parameters of a stolen model for ownership verification.

Zhang et al. (2018), which is the first work on trigger set-based watermarking, propose to watermark a model by training it over a trigger set, which consists of query images and their target labels. Note that a target label is a *false* label with respect to its ground-truth label, and the model owner intentionally teaches the model to output this target label when a query image is given. Accordingly, the owner's model would classify a query image as its predefined target label, whereas other unwatermarked models would predict a query image as its ground-truth label. The owner can then verify the ownership of a model by leveraging this gap (*i.e.*, conducting a statistical test or measuring a trigger set recall). Other trigger set-based schemes also share a similar approach, but slightly differ in how they create query images, assign target labels, and train the watermarked model.

To evaluate the robustness of the watermarking schemes, researchers have proposed several attacks that aim to thwart ownership verification. For instance, Adi et al. (2018) proposed to overwrite watermarks embedded in a DNN model by re-training; Fan et al. (2019) introduced an adversary who reverse-engineers query images to claim the ownership; Chen et al. (2021a) attempted to fine-tune the watermarked model to remove watermarks; Namba & Sakuma (2019) demonstrated an attacker who detects queries that contain query images on the fly; and Jia et al. (2021) evaluated whether watermarks survive against model extraction attacks, but the surrogate model exhibited an insufficient statistical difference to claim the model owner's ownership. Similar to our work, the adversarial examples can be used for verifying the ownership (Lukas et al., 2020; Le Merrer et al., 2020). In our case, we construct the trigger set before watermarking, and train the model with a margin which is differ from directly using adversarial examples for verification.

2.2 Model extraction attacks

Since building a high-performing DNN model requires considerable efforts in designing the model architecture, constructing a training set, and actual training of the model, an adversary may attempt to copy the functionality of a remote model into his/her local model. Assuming such an adversary, researchers have proposed model extraction attacks (*i.e.* model stealing attacks) that extract the victim model's ability to predict images and transfer it into the attacker's model (Tramér et al., 2016; Orekondy et al., 2019; Chandrasekaran et al., 2020; Papernot et al., 2017). Specifically, an adversary in this attack scenario collects an arbitrary set of public data, exploiting the output confidence vector of a victim model as a labeling oracle, and utilizing the collected pairs to train a counterfeit model.

Model extraction attacks could be used to erase watermarks, as it would not introduce the watermark of a victim model into a new model. Since the public images used for model extraction are randomly collected, the query samples from the trigger set is highly unlikely to be included in the surrogate dataset; the surrogate model thus has little chance to copy the exact trigger set during extraction. In this regard, several researchers have designed watermarking schemes that are robust against model extraction (Jia et al., 2021; Maini et al., 2021; Li et al., 2022). Jia et al. (2021) tightly coupled the model's original task with the query image prediction task using the soft nearest neighbor loss so that watermarks always remain if the attacker copies the victim model's capability of predicting regular images. For ownership verification, Maini et al. (2021) suggested measuring the distance between the data points in the owner's training set and the decision boundary of a suspicious model. The key observation behind their approach is that knowledge learned from the owner's training set propagates to all counterfeit models. Unlike the previous methods, our method does not require the query samples for the trigger set to be out-of-distribution samples, and thus we can use in-distribution samples for watermarking. This is beneficial since the out-of-distribution (OOD) samples could be easily filtered out using OOD detectors, to hinder ownership verification. While the previous works couple or associate the trigger set with the original objective to transfer the watermarking into the surrogate models, our method does not and leads to learn strong correlation by giving the margin to queries.

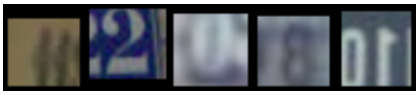
3 Method

In this section, we first introduce a general trigger set-based watermarking scheme, and then describe our method.

3.1 WATERMARKING BY USING TRIGGER SET

Let the original objective be learning the mapping h_θ from x to y in $\mathcal{D} = \{(x, y)\}$, parameterized by θ. To watermark the model, the model owner constructs the trigger set $\mathcal{D}_q = \{(x_q, y_q)\}$ for the ownership verification. Suppose the owner discovers the suspicious model which is believed to be a counterfeit model extracted from the target model h_θ. To verify the ownership, the owner can query the samples in the trigger set $\mathcal{D}_q$ to the suspicious model and compare the statistics of the response from h_θ and that of the suspicious model.

If the ordinary mapping of $\mathcal{D}$ has the similar statistics to $\mathcal{D}_q$, the ownership verification is difficult. Thus, we should ensure that the mapping of $\mathcal{D}_q$ is clearly distinguishable from the ordinary mapping. If the sample of the trigger set x_q is sampled from the original objective $\mathcal{D}$, then there exists a corresponding mapping y'_q in $\mathcal{D}$. Thus, the corresponding label y_q should be differ from y'_q, and the mapping of x_q to y_q can give a strong evidence for verifying the ownership. The example of the mapping is illustrated in Figure 2.



Label of $\mathcal{D}$	1	2	0	8	1
Label of $\mathcal{D}_q$	0	3	2	7	6

Figure 2: An example of the trigger set $\mathcal{D}_q$. The standard model maps the samples to the corresponding labels from $\mathcal{D}$, but the watermarked model maps the samples to the labels from $\mathcal{D}_q$.

The basic way to learn the mapping of both $\mathcal{D}$ and $\mathcal{D}_q$ is to use the training objective given by,

$$\min_\theta \mathbb{E}_{(x,y)\sim\mathcal{D}\setminus\mathcal{D}'_q}\Big[\ell\big(h_\theta(x), y\big)\Big] + \mathbb{E}_{(x_q,y_q)\sim\mathcal{D}_q}\Big[\ell\big(h_\theta(x_q), y_q\big)\Big], \tag{1}$$

where the mapping of $\mathcal{D}'_q$ should be rejected for watermarking. Training with the objective in Equation 1 can easily watermark the given model, achieving perfect accuracy on $\mathcal{D}_q$. However, the functionality stealing methods diminishes the correlation of the trigger set (see Table 1). This is because the mapping of the trigger set $\mathcal{D}_q$ is the false mapping for the original objective $\mathcal{D}$, and thus the samples from $\mathcal{D}_q$ is considered as the outliers.

3.2 MARGIN-BASED WATERMARKING

To overcome such weakness, the training objective should make the model learn strong correlation of the trigger set $\mathcal{D}_q$ into the model h_θ, and thus the watermarking is not removed even with the functionality stealing attacks. To this end, we propose a margin-based watermarking method for deep neural networks. The idea is inspired from the adversarial attack and robust training (Madry et al., 2018).

Table 1: The results for the method in Equation 1. Since the opposite label of $\mathcal{D}_q$ is considered as the outliers for the functionality stealing attacks, the surrogate model obtained by extraction does not maintain the correlation of the trigger set.

Model	Watermarked	Extraction
Obj. Acc.	0.9310	0.9060
Wat. Acc.	1.0000	0.0000

The objective of our margin-based watermarking is given as follows:

$$\min_\theta \mathbb{E}_{(x,y)\sim\mathcal{D}\setminus\mathcal{D}'_q}\Big[\ell\big(h_\theta(x), y\big)\Big] + \mathbb{E}_{(x_q,y_q)\sim\mathcal{D}_q}\Big[\max_{\mathtt{tr}}\ell\big(h_\theta(\mathtt{tr}(x_q)), y_q\big)\Big], \tag{2}$$

where $\mathtt{tr}$ indicates a semantic-preserving transformation which cannot change the mapping of x_q to y_q.

For adversarial training, the objective is to defend against the adversarial attacks which try to lead the model into making incorrect predictions by adding a small amount of ℓ_p perturbation. Thus, the adversarial training utilizes the valid transformation as the ℓ_p perturbations. We also utilize the transformation by adding limited ℓ_p perturbation, which is given by

$$\min_\theta \mathbb{E}_{(x,y)\sim\mathcal{D}\setminus\mathcal{D}'_q}\Big[\ell\big(h_\theta(x), y\big)\Big] + \mathbb{E}_{(x_q,y_q)\sim\mathcal{D}_q}\Big[\max_{\|\delta\|_\infty\leq\epsilon}\ell\big(h_\theta(x_q+\delta), y_q\big)\Big], \tag{3}$$

where ϵ is the hyperparameter for choosing the proper margin. To solve the inner maximization problem of Equation 3, we can use the projected gradient ascent that is commonly used for adversarial

Algorithm 1 Margin-based watermarking

Input: The original objective $\mathcal{D}$, trigger set (watermarking set) $\mathcal{D}_q$, model h_θ, learning rate for parameter η_θ, loss function ℓ, step size for projected gradient ascent α, maximum bound for projected gradient ascent ϵ, # of iteration K.
Output: Watermarked model h_θ.

while not converge **do**
  $x_{batch}, y_{batch} \sim \mathcal{D}$
  $\ell_{obj} \leftarrow \ell(h_\theta(x_{batch}), y_{batch})$
  $x_{wat}, y_{wat} \sim \mathcal{D}_q$
  $\delta \leftarrow \mathbf{0}_{x_{wat}}$
  for iter **in** range(K) **do**
    $\hat{\ell}_{wat} \leftarrow \ell(h_\theta(x_{wat} + \delta), y_{wat})$
    $\delta \leftarrow \delta + \alpha \cdot \texttt{sign}(\nabla_{x_{wat}} \hat{\ell}_{wat})$
    $\delta \leftarrow \texttt{Proj}(\delta, \epsilon)$
  end for
  $\ell_{wat} \leftarrow \ell(h_\theta(x_{wat} + \delta), y_{wat})$
  $\ell \leftarrow \ell_{obj} + \ell_{wat}$
  $\theta \leftarrow \theta - \eta_\theta \cdot \nabla_\theta \ell$
end while

training (Madry et al., 2018). The solution of the inner maximization term in Equation 3 depends on the current parameters of the model. Thus, the training objective will ensure that the model's decision boundary has a large margin to each query. The given margin of the trigger set induces the twisted decision boundary of the model when compared with that of the standard model. Since the prediction for any sample internally represents the clue of the distorted decision boundary of h_θ, the margin can be transferred by the functionality stealing methods. Also, we can guarantee the model learns the mapping of x_q to y_q when the loss of $x_q + \delta$ with y_q converges to the sufficiently small value, since it is the upper bound for the loss of x_q to y_q. The entire watermarking procedure of our method is described in Algorithm 1.

4 Experiments

4.1 Functionality Stealing: Distillation and Extraction

To begin, we briefly introduce the distillation and extraction which are used for the functionality stealing methods.

Distillation is originally proposed for distilling the knowledge from the large model to the small model. The training objective of the distillation is written as,

$$\min_{\hat{\theta}} \mathbb{E}_{(x,y)\sim\mathcal{D}} \Big[\alpha \cdot \ell\big(\hat{h}_{\hat{\theta}}(x), y\big) + \beta \cdot \tilde{\ell}\big(\hat{h}_{\hat{\theta}}(x), h_\theta(x)\big) \Big], \tag{4}$$

where $\hat{h}_{\hat{\theta}}$ is the surrogate model (student model), α and β are hyperparameters. Note that the dataset $\mathcal{D}$ is same as the training dataset used for learning teacher model h_θ. When considering the scenario of the functionality stealing by the adversary, the distillation is not applicable in practice since the distillation requires the source dataset which used for training the watermarked model. Also, the distillation objective is composed of two measures, the loss with the true label and the loss with the target model. Thus, for watermarked model, the distillation can be seen as the untargeted adversary since the original objective has no correlation or negative correlation with the trigger set. From the viewpoint, the distillation can be considered as the upper bound for functionality stealing method, which means that the distillation is the most powerful adversary. We provide the additional discussion about the distillation in the appendix B.2.

On the other hand, model extraction is another functionality stealing method. The training objective for model extraction is given by,

$$\min_{\hat{\theta}} \mathbb{E}_{(\tilde{x},\tilde{y})\sim\tilde{\mathcal{D}}} \Big[\tilde{\ell}\big(\hat{h}_{\hat{\theta}}(\tilde{x}), h_\theta(\tilde{x})\big) \Big], \tag{5}$$

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 model without watermarking, the lower is better since the accuracy of the watermarking can have more statistical significance. The result shows that our method strictly outperforms the baseline for ownership verification using DNN watermarking.

Dataset		SVHN		CIFAR10		CIFAR100	
Method		EWE	Ours	EWE	Ours	EWE	Ours
Watermarked Model	Obj. Acc.	0.9015 ± 0.0110	0.9222 ± 0.0230	0.8610 ± 0.0054	0.8947 ± 0.0066	0.5511 ± 0.0167	0.5966 ± 0.0662
	Wat. Acc.	0.4532 ± 0.1517	$\mathbf{1.0000 \pm 0.0000}$	0.2668 ± 0.0822	$\mathbf{1.0000 \pm 0.0000}$	0.6814 ± 0.1016	$\mathbf{1.0000 \pm 0.0000}$
Distillation	Obj. Acc.	0.9325 ± 0.0030	0.9307 ± 0.0211	0.8888 ± 0.0035	0.9255 ± 0.0043	0.6373 ± 0.0040	0.5731 ± 0.0616
	Wat. Acc.	0.0495 ± 0.0165	$\mathbf{0.6600 \pm 0.1054}$	0.0164 ± 0.0105	$\mathbf{0.6667 \pm 0.1930}$	0.0573 ± 0.0342	$\mathbf{0.2733 \pm 0.2031}$
Extraction (same source)	Obj. Acc.	0.8844 ± 0.0092	0.9321 ± 0.0223	0.8397 ± 0.0102	0.9063 ± 0.0328	0.5300 ± 0.0157	0.6363 ± 0.0812
	Wat. Acc.	0.7623 ± 0.0802	$\mathbf{0.8200 \pm 0.1453}$	0.5101 ± 0.0558	$\mathbf{0.7300 \pm 0.0300}$	0.3090 ± 0.1134	$\mathbf{0.7500 \pm 0.3477}$
Extraction (different source)	Obj. Acc.	0.6499 ± 0.0455	0.9354 ± 0.0179	0.7076 ± 0.0040	0.8699 ± 0.0008	0.3210 ± 0.0182	0.4998 ± 0.0780
	Wat. Acc.	$\mathbf{0.9540 \pm 0.0385}$	0.7467 ± 0.0321	0.6624 ± 0.1422	$\mathbf{0.8033 \pm 0.0611}$	0.3576 ± 0.0487	$\mathbf{0.8967 \pm 0.1447}$
Model w/o Watermark	Obj. Acc.	0.9629 ± 0.0017		0.9356 ± 0.0031		0.7422 ± 0.0019	
	Wat. Acc.	0.0326 ± 0.0125	$\mathbf{0.0067 \pm 0.0058}$	0.0002 ± 0.0002	$\mathbf{0.0000 \pm 0.0000}$	0.0122 ± 0.0123	$\mathbf{0.0067 \pm 0.0115}$

where $\tilde{\mathcal{D}}$ is the surrogate dataset for extraction and $\tilde{\ell}$ is the loss function which can be differ from the original loss function ℓ. As the Equation 5 shows that the surrogate dataset should not be same with the dataset used for training the target model h_θ. Even using the different dataset, the previous work shows that the model extraction can achieve the reasonable performance of the original objective (Orekondy et al., 2019). Also, the loss function $\tilde{\ell}$ for model extraction can have various types, including cross entropy, KL divergence and ℓ_1-loss. To validate the watermarking methods, we perform two types of model extraction with two different surrogate datasets for each watermarked model. In all the cases, we used KL divergence for the loss function $\tilde{\ell}$.

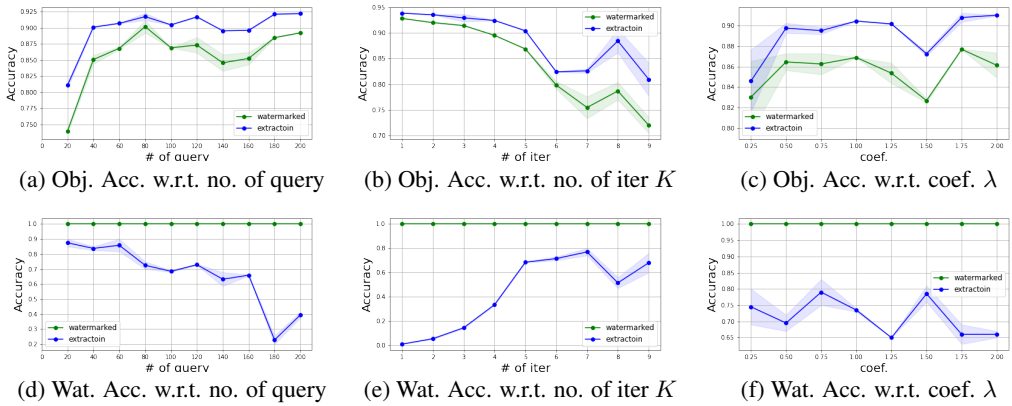
4.2 Experimental Setup

We verify the efficacy of our method, by validating it on benchmark datasets, namely CIFAR10, CIFAR100 and SVHN. For all experiments, we firstly train the watermarked model with the objective described in Section 3. Next, we perform functionality stealing attacks for each watermarked model. As for functionality stealing attacks to defend against, we first consider distillation, which is the strongest attack that is only possible in the white-box attack scenario. Also, we consider model extraction with both the same dataset to the training dataset, and a different dataset. For CIFAR10 dataset experiments, we perform model extraction with CIFAR10 and CIFAR100 surrogate dataset, for SVHN, we use SVHN and CIFAR10, and for CIFAR100, we use CIFAR100 and CIFAR10. We expect that the model extraction with the dataset used for learning the target model is the best adversary for the model extraction (Jia et al., 2021), but for fair comparison, we assume the realistic setting for model watermarking and attacks. For all the experiments, we used the ResNet-34 (He et al., 2016) for the watermarked model and the ResNet-34 for extraction and ResNet-18 for distillation of the surrogate model.

We randomly select the trigger set from the validation set of the original objective, *i.e.*, our method does not have any constraints on the choice of trigger set. The corresponding label of each query is randomly selected from the false labels with respect to its ground-truth label, which ensures the statistics of the trigger set is clearly distinguishable with the original objective. We follow the hyperparameters of He et al. (2016) for training all the ResNets and Hinton et al. (2015) for distillation. For our method, we set the margin as $\|\delta\|_\infty \leq 5/255$ and number of steps $K = 5$ in Equation 3 and Algorithm 1. All the experiments are done by one Titan Xp or RTX 2080Ti GPU. The detailed description of our experiments are in Section B.1 of the appendix.

4.3 Results on Functionality Stealing

Table 2 shows the original and watermarking accuracies of different models against model extraction and distillation attacks for different two methods, margin-based watermarking and EWE (Jia et al., 2021) which we found by measuring the watermark accuracy. All the experiments are done for 3 times with the trigger set size of 100 and we denote the mean and standard deviation for each experiment. The result show that our method achieves perfect accuracy on the trigger set, while EWE fail to do so. This allows to clearly distinguish the standard model and watermarked model of ours. Additionally, our method achieves better objective performance on all the cases. For the ownership



(a) Obj. Acc. w.r.t. no. of query (b) Obj. Acc. w.r.t. no. of iter K (c) Obj. Acc. w.r.t. coef. λ

(d) Wat. Acc. w.r.t. no. of query (e) Wat. Acc. w.r.t. no. of iter K (f) Wat. Acc. w.r.t. coef. λ

Figure 3: Comparison of hyperparameters. We measure the objective accuracy and watermark accuracy for both watermarked model and stolen model while varying the size of the trigger set, number of iterations (margin), and coefficient λ in Equation 6. The shaded region shows the standard deviation of the estimated values.

verification of the suspicious models, our method outperforms the watermark accuracy for all the cases except the case of SVHN, extraction with different source. Since the EWE models on SVHN utilizes the CIFAR10 samples for the trigger set, the watermark accuracy of this surrogate model can achieve the high watermarking accuracy. Even the watermark accuracy is high in the surrogate model, the watermark accuracy of the watermarked model shows lower than of the surrogate model, which can confuse the ownership verification. Moreover, although the model extraction partially erase the watermark, the models trained with our method still achieve sufficiently high watermark accuracy for verifying the ownership. The models with distillation or extraction show sufficiently high watermarking accuracy, we can assure the model which steal the functionality or not by measuring the watermarking accuracy since the standard model achieves near 0 accuracy on the trigger set.

4.4 The choice of the margin and query

The size of the trigger set We measure the performance of our method while varying the size of the trigger set. Since training the model to predict the assigned labels for the query samples in the trigger set provides false signals that conflict with conventional training, we conjecture that the size of the trigger set may be effective for the watermarking. Also, for ownership verification, being able to watermark a large trigger set is beneficial for ownership verification since the statistical significance can be measured more precisely. We vary the size of the trigger set as 20 to 200, and the results are shown in Figure 3a and 3d. The stolen model from the watermarked model with bigger size of the trigger set shows lower watermark accuracy. This tendency arise from the false mapping characteristics of the trigger set, that the method can being unstable with the large size of the trigger set. This shows that the margin-based approach can inject the strong correlation of the trigger set, but can only inject limited size of the false mapping. However, for the smaller size of the trigger set, the stolen model captures the most of the watermarking. Thus, we recommend to use the limited size of the trigger set, such as 100 which used for our experiments in Table 2.

Varying the margin Although our method achieves good watermarking performance on DNNs, however, the performance of the model on the original test set is decreased. This is somehow similar to the result of adversarial training, which trades-off the accuracy with robustness to adversarial perturbations. We investigate how to achieve good watermarking performance without sacrificing the model accuracy on conventional examples. To this end, we consider two settings. First, we can control the margin by increasing/decreasing the penalty (*i.e.*, ϵ in Equation 3) and the number of iterations (K in Equation 3). We vary the number of iterations and maximum perturbation size ϵ for margin, and the results are shown in Figure 3b and 3e. The results show that weaker margin by smaller K improves the performance of the original objective. However, since the margin decreases, the signal for the watermarked query samples decreases as well, with the model extraction attacks. Further, when the margin increases, the training becomes unstable. We conjecture that the mapping of the trigger set is opposite to the true mapping, thus the training becomes unstable where the maximum margin size increases. This result also implies that there exists trade-off between watermarking performance and the performance on the original samples.

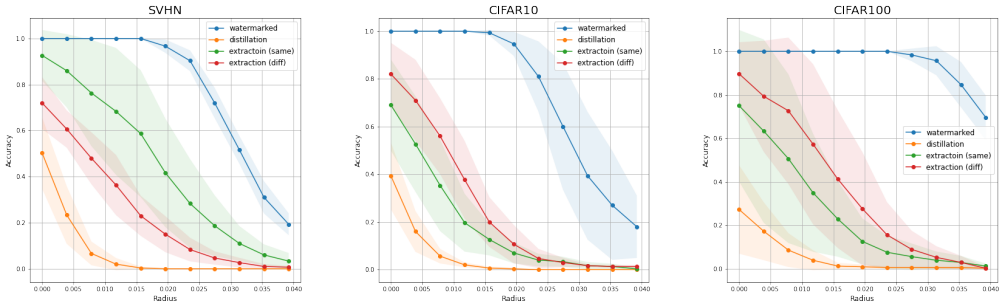


Figure 4: Degradation of accuracy according to change of the maximum radius in watermarked and surrogate models. The shaded region shows the standard deviation of the estimated values. The result proves that our intuition is reflected to the watermarking.

Another way is to control the effect of the watermarking loss, by controlling the coefficient λ in the below objective:

$$\min_{\theta} \mathbb{E}_{(x,y)\sim\mathcal{D}\setminus\mathcal{D}'_q}\Big[\ell\big(h_\theta(x),y\big)\Big] + \lambda\mathbb{E}_{(x_q,y_q)\sim\mathcal{D}_q}\Big[\max_{\|\delta\|_\infty\leq\epsilon}\ell\big(h_\theta(x_q+\delta),y_q\big)\Big]. \tag{6}$$

We set the coefficient λ as 0.25 to 2 respectively. The results with varying λ are provided in Figure 3c and 3f. The results show that varying λ in range 0.25 to 2 does not show difference over both the original objective and watermark. This suggests that the margin size is more important than the ratio between the original and watermarking objective. We additionally perform experiments of the choice of the query by selecting each query as a Mixup (Zhang et al., 2017) query, which is in Section B.3 of the appendix.

4.5 Labeling strategy for trigger sets

Our method is designed to distort the model's decision boundaries by assigning a false label for each one of the trigger set. For each trigger image, we randomly selected one of the false labels with respect to its ground-truth label. Considering that the choice of a trigger set and its corresponding false labels may vary the distortion of a decision boundary, we compare the watermark and original task accuracies over different labeling strategies.

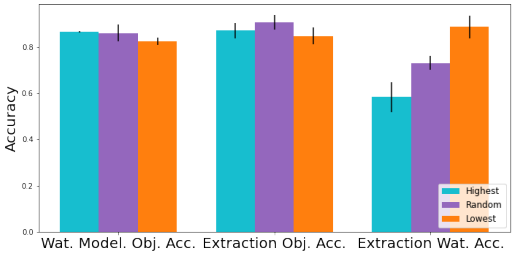


Figure 5: Objective and watermark accuracy for the watermarked model and its surrogate (by extraction) across the different labeling strategies.

We leverage the confidence scores that the model without watermarking emits for each query to choose a label. Note that a strategy of selecting a false label having the highest confidence score would distort the decision boundary less compared to that of using the lowest confidence score. Therefore, we compare these two cases (*i.e.*, selecting the highest and lowest confidence score) to the random labeling strategy. The results are shown in Figure 5.

In all the cases, the watermarked model showed the perfect watermark accuracy so we omitted it in the figure. When using the highest confidence score strategy, the watermarked model achieved the highest objective accuracy; however, the model showed the lowest watermark accuracy after conducting model extraction. By contrast, the lowest confidence score strategy degrades the objective accuracy, as it would distort the decision boundary of the watermarked model more aggressively. Recall from Figure 1 that this extracted model imitates the more distorted decision boundary of the watermarked model and thus has a higher chance of copying the watermark. Therefore, this strategy helps the extracted model achieve higher watermark accuracy, which enables claiming a stronger ownership. The results also demonstrate that there exists a trade-off between the watermark and the performance of the original task. Above all, our method shows sufficient watermark accuracy for ownership verification compared to the other labeling strategies, which enables that the model owners can adopt their own labeling strategy without suffering from the performance.

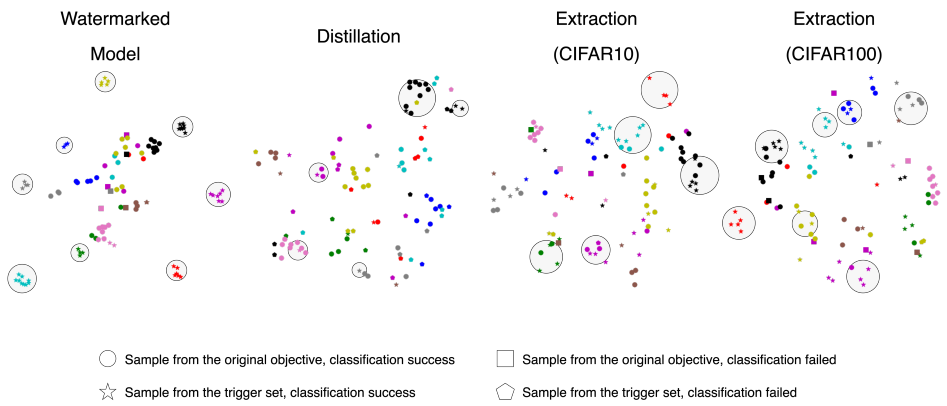


Figure 6: t-SNE visualization of the watermarked model and the surrogate models for CIFAR10. Same colored markers indicates that the markers have the same class label. The gray shaded circle represents the margin area of the queries.

4.6 Comparison of the margin between the watermarked and surrogate models

We investigate how well the margin of the queries from the trigger set transfers to the surrogate models. We perform the steepest gradient ascent for the trigger set to characterize the margin of each queries while varying the maximum radius (ϵ in Equation 3). The results are shown in Figure 4. Note that the surrogate models built by distillation and extraction have not seen the queries from the trigger set. We observe that although not all of the margins transfer to the surrogate models, some of them do. This gives the credence for our intuition that the margin of the decision boundary transfers to the surrogate models, even though the functionality stealing methods aim to indirectly copy the functionality using a surrogate dataset. We conjecture that for any watermarking methods that aim to impose strong correlations on the trigger set, our margin-based methods will also ensure that they are transferred to the surrogate models, trained to copy the functionalities of the original models.

4.7 Visualization of the trained models

We additionally perform the visualization of the embedding space obtained with our proposed margin-based watermarking method. We use t-SNE (Van der Maaten & Hinton, 2008) to visualize the confidence score for each sample from the original objective and queries from the trigger set. The results are shown in Figure 6. For clear interpretation, we shaded the margin area for queries which achieves the successful watermarking. Since the watermarked model is trained by Equation 3, the queries are strongly clustered together. Moreover, despite the surrogate models do not experience the queries during the functionality stealing, the survived queries also have some margins. This result additionally supports our claim that the margin is transferred during functionality stealing. For more visualization, please see Section C of the appendix.

5 Conclusion

In this work, we proposed margin-based watermarking of deep neural networks. The margin-based watermarking train the trigger set with a sufficient margin and achieves the robust decision boundary of the queries. We validate the margin-based watermarking for two functionality stealing methods, distillation and extraction. For all the cases, the margin-based watermarking outperforms the baselines on the watermarking accuracy. Additionally, we compare the size of both trigger set and margin, where the latter one largely affects to watermark the model. Through the quantitative and qualitative analysis, we prove that the margin is transferred to the surrogate models by the functionality stealing methods due to the learned strong correlation of the trigger set. Lastly, we show that our method can use any kind of queries where the only requirement is that the set is distinguishable from the original task. Our results suggest that there exists trade-offs between watermarking and objective accuracy, and we aim to seek for a better method for improving both of them.

REPRODUCIBILITY STATEMENT

We provide the source codes for our experiment of Table 2, Figure 3 and 4. We describe the experimental settings in Section 4.2 and B.1. We would publicly release the codes after the acceptance.

REFERENCES

Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *Proceedings of the USENIX Security Symposium*, pp. 1615–1631, 2018.

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

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020.

Varun Chandrasekaran, Kamalika Chaudhuri, Irene Giacomelli, Somesh Jha, and Songbai Yan. Exploring connections between active learning and model extraction. In *Proceedings of the USENIX Security Symposium*, pp. 1309–1326, 2020.

Huili Chen, Bita Darvish Rouhani, Cheng Fu, Jishen Zhao, and Farinaz Koushanfar. 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.

Xinyun Chen, Wenxiao Wang, Chris Bender, Yiming Ding, Ruoxi Jia, Bo Li, and Dawn Song. REFIT: A unified watermark removal framework for deep learning systems with limited data. In *Proceedings of the ACM Asia Conference on Computer and Communications Security*, pp. 321–335, 2021a.

Xuxi Chen, Tianlong Chen, Zhenyu Zhang, and Zhangyang Wang. 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*, 2021b.

Lixin Fan, Kam Woh Ng, and Chee Seng Chan. Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks. In *Proceedings of the Advances in Neural Information Processing Systems*, pp. 4716–4725, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Hengrui Jia, Christopher A. Choquette-Choo, Varun Chandrasekaran, and Nicolas Papernot. Entangled watermarks as a defense against model extraction. In *Proceedings of the USENIX Security Symposium*, pp. 1937–1954, 2021.

Erwan Le Merrer, Patrick Perez, and Gilles Trédan. Adversarial frontier stitching for remote neural network watermarking. *Neural Computing and Applications*, 32(13):9233–9244, 2020.

Yiming Li, Linghui Zhu, Xiaojun Jia, Yong Jiang, Shu-Tao Xia, and Xiaochun Cao. Defending against model stealing via verifying embedded external features. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

Zheng Li, Chengyu Hu, Yang Zhang, and Shanqing Guo. 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.

Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 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.

Weitang Liu, Xiaoyun Wang, John D Owens, and Yixuan Li. Energy-based out-of-distribution detection. *arXiv preprint arXiv:2010.03759*, 2020.

Raphael Gontijo Lopes, Stefano Fenu, and Thad Starner. Data-free knowledge distillation for deep neural networks. *arXiv preprint arXiv:1710.07535*, 2017.

Nils Lukas, Yuxuan Zhang, and Florian Kerschbaum. Deep neural network fingerprinting by conferrable adversarial examples. In *International Conference on Learning Representations*, 2020.

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

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

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

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

Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. 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.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the ACM Asia Conference on Computer and Communications Security*, pp. 506–519, 2017.

Bita Darvish Rouhani, Huili Chen, and Farinaz Koushanfar. 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.

Florian Tramér, Fan Zhang, Ari Juels, Michael K. Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction APIs. In *Proceedings of the USENIX Security Symposium*, pp. 601–618, 2016.

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

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

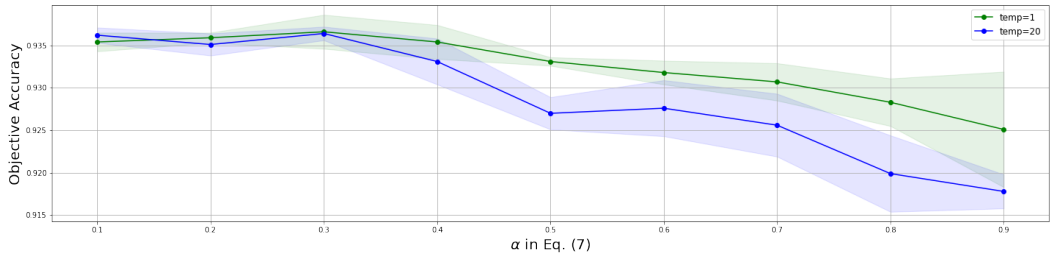
Peng Yang, Yingjie Lao, and Ping Li. Robust watermarking for deep neural networks via bi-level optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14841–14850, 2021.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

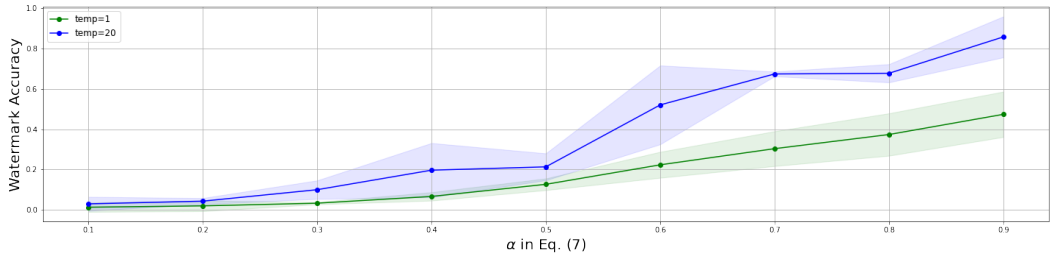
Jialong Zhang, Zhongshu Gu, Jiyong Jang, Hui Wu, Marc Ph. Stoecklin, Heqing Huang, and Ian Molloy. Protecting intellectual property of deep neural networks with watermarking. In *Proceedings of the ACM Asia Conference on Computer and Communications Security*, pp. 159–172, 2018.

Jie Zhang, Dongdong Chen, Jing Liao, Han Fang, Weiming Zhang, Wenbo Zhou, Hao Cui, and Nenghai Yu. Model watermarking for image processing networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020a.

Jie Zhang, Dongdong Chen, Jing Liao, Weiming Zhang, Gang Hua, and Nenghai Yu. Passport-aware normalization for deep model protection. In *Proceedings of the Advances in Neural Information Processing Systems*, pp. 22619–22628, 2020b.



(a) Objective accuracy



(b) Watermark accuracy

Figure 7: Objective and watermark accuracy against distillation while varying α and T in Equation 7.

A SUMMARY

This appendix is constructed as follows: we provide experimental settings, the meaning of the distillation for evaluating the watermarking method, 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 watermarked model. We use $\tilde{\ell}$ in Equation 4 and 5 as KL divergence. The hyerparameters of distillation in Equation 4 is same as Hinton et al. (2015) which is discussed at Section B.2 in detail. 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 EWE (Jia et al., 2021) experiments, we used the authors' source code from Jia et al. (2021).

B.2 DISTILLATION FOR EVALUATING THE WATERMARKING METHOD

The purpose of distillation is to distill the knowledge from high-performing teacher model to the small student model, which may underperform because of its low capacity. Thus the distillation can be seen as the functionality copying mechanism, which can selectively transfer the knowledge from the teacher model to the student model. Various distillation method can be exist such as data-free distillation (Lopes et al., 2017), but we used the knowledge distillation which fully utilizes the source dataset, the sample and its corresponding label. The distillation used in our experiment cannot be performed by the adversary since there needs an assumption that the adversary should hold the source dataset for the watermarked model, and under this assumption, the adversary need not to steal the functionality of the watermarked model. Nevertheless, evaluating the watermarking method against distillation is still meaningful. We introduce the reason and additional experiments of distillation in this section.

The watermarking methods by using trigger set mainly focus on the response or predictoin of the model, where the response should be distinguishable with the common models so that the victim can verify whether the suspicious model steals the watermarked model or not. Thus the sample-label assignment rule of the trigger set should be orthogonal to that of the original task. 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 the distillation.

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

$$\min_{\hat{\theta}} \mathbb{E}_{(x,y)\sim\mathcal{D}}\Big[(1-\alpha)\cdot \mathbf{CE}\big(\sigma(\mathbf{z}_s), y\big) + 2\alpha T^2 \cdot \mathbf{KL}\big(\sigma(\mathbf{z}_s/T), \sigma(\mathbf{z}_t/T)\big)\Big], \tag{7}$$

where σ is the softmax function, $\mathbf{z}_s, \mathbf{z}_t$ indicates the logit of the student and teacher model and α and T are hyperparameters. We used $\alpha = 0.7$ and $T = 20$ to evaluate the watermarking in Table 2 which also used in Hinton et al. (2015). Intuitively, increasing the ratio of the cross-entropy term in Equation 7 can produce the model which may more similar with the model without any watermraking. To verify this, we perform the distillation for our watermarking method while varying α and T. The results are shown in Figure 7.

Since the temperature T controls both the logit and the ratio between cross-entropy and KL divergence, using the lower temperature $T = 1$ diminishes more watermark than the case of $T = 20$. Also, the low α suppress the knowledge transfer from the watermarked model to the surrogate model, the watermark accuracy drops while decreasing α. Even though is no targeted elimination method for watermark, however, the distillation is still effective for erasing the watermark. In conclusion, the distillation used in our experiment can be used for 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 8: 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 average of the Mixup components and the corresponding label is randomly selected except the true label of the Mixup components, see Figure 8. We set the number of Mixup query of 100 and the number of Mixup components of 2 and 5. Since one Mixup query contains multiple correlation of the Mixup components, the false correlation of the trigger set might be stronger than of our previous settings.

Table 3: Comparison of the number of Mixup queries.

# of Mixup		2	5
Watermarked Model	Obj. Acc.	0.7928	0.4480
	Wat. Acc.	**1.0000**	**1.0000**
Distillation	Obj. Acc.	0.9030	0.8976
	Wat. Acc.	**0.4200**	0.1000
Extraction (CIFAR10)	Obj. Acc.	0.8006	0.4440
	Wat. Acc.	0.6700	**0.8300**
Extraction (CIFAR100)	Obj. Acc.	0.7524	0.3966
	Wat. Acc.	**0.8900**	0.8500

Even with the stronger false correlation with the original objective, the result in Table 3 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 the 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 4: p-value for watermarked model and its surrogate models. The results show that our method can show superior statistical significance than of the previous methods.

Method		EWE (Jia et al., 2021)	Maini et al. (2021)	Li et al. (2022)	**Ours**
p-value (CIFAR10)	Wat. Model	10^{-8}	10^{-4}	10^{-7}	10^{-13}
	Distill.	10^{-8}	10^{-4}	10^{-7}	10^{-8}
	Extract.	10^{-8}	10^{-3}	10^{-3}	10^{-8}

Table 5: 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	CIFAR10 Acc.	Energy of $\mathcal{D}$	Success queries in $\mathcal{D}_q$	Failed queries in $\mathcal{D}_q$
Wat. Model	0.8414	-7.7472	**-15.1316**	N/A
Distill.	0.9200	**-7.8103**	-7.2842	-6.9181
Extract. (same)	0.8662	-7.2169	**-7.3657**	-6.2308
Extract. (diff)	0.8010	-6.7147	**-8.0391**	-6.1190

B.4 Statistical significance comparison

We provide the comparison of the statistical significance by measuring the p-value of the watermarked model against the model without watermarking. The results are shown in Table 4. As shown in Table 4, our method builds the model which is clearly differentiable with the model without watermarking.

B.5 In-distribution Analysis

To prevent one scenario that the adversary reject the queries to reject the ownership verification process, we analyze the additional quantity: in-distribution analysis. Suppose one watermarking method achieves the 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 confirms the query as the in-distribution sample or not.

To check the query which is 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 5. The results shows that the energy of the success queries in $\mathcal{D}_q$ have the lower energy than of the samples from 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.

C t-SNE visualization

We provide the additional t-SNE visualization in Figure 9 with perplexity of 10 where the figure does not represent the margin area. In Figure 9, 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.

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

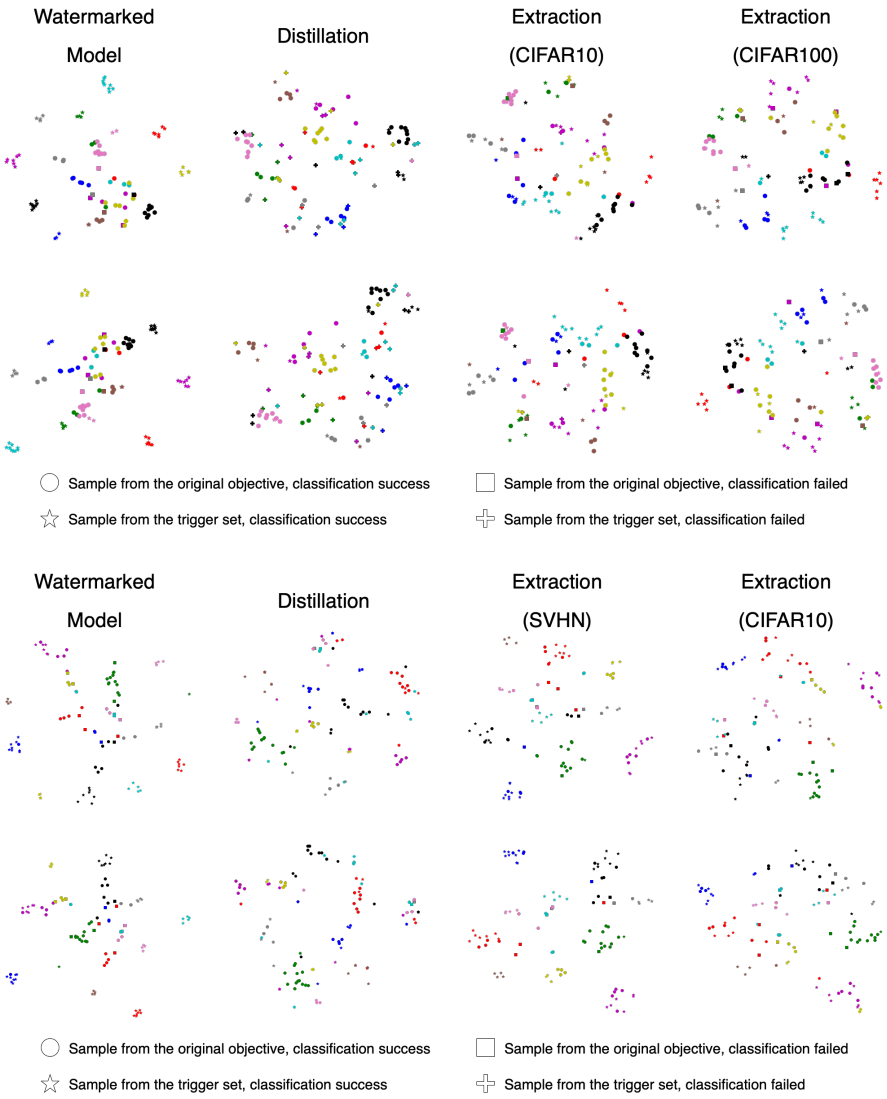


Figure 9: t-SNE visualization of the watermarked model and the surrogate models for CIFAR10 (top) and SVHN (bottom).

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 10 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 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,

$$\min_{\hat{\theta}} \mathbb{E}_{(\tilde{x},\tilde{y})\sim\tilde{\mathcal{D}}}\Big[\tilde{\ell}\big(\hat{h}_{\hat{\theta}}(\tilde{x}+\tilde{\delta}), h_\theta(\tilde{x}+\tilde{\delta})\big)\Big], \quad \tilde{\delta} = \arg\max_{\|\delta\|\leq\epsilon} \tilde{\ell}(h_\theta(\tilde{x}+\delta), h_\theta(\tilde{x})). \tag{8}$$

The stolen model with the extraction of Equation 8 can achieve more similar decision boundary because the existence of δ makes extraction to experience more various input images and its cor-

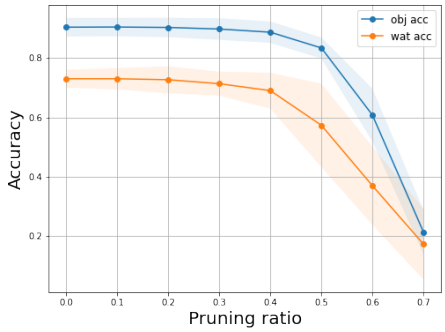


Figure 10: Results against pruning attack. Even with pruning, the stolen model still maintain the watermark accuracy. When the watermark accuracy largely drops, the objective accuracy also largely drops.

Table 6: Results of Equation 8. 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
Watermarked Model	Obj. Acc.	0.8947 ± 0.0066
	Wat. Acc.	$\mathbf{1.0000 \pm 0.0000}$
Extraction (margin)	Obj. Acc.	0.8494 ± 0.0048
	Wat. Acc.	$\mathbf{0.9833 \pm 0.0058}$

responding predictions. Table 6 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.