applied sciences

MDPI

Article

AWEncoder: Adversarial Watermarking Pre-Trained Encoders in Contrastive Learning

Tianxing Zhang [1], Hanzhou Wu [1], Xiaofeng Lu [1,2], Gengle Han [1] and Guangling Sun [1,*]

[1] School of Communication and Information Engineering, Shanghai University, Shanghai 200444, China; zhang_tx@shu.edu.cn (T.Z.); hanzhou@shu.edu.cn (H.W.); luxiaofeng@shu.edu.cn (X.L.); hangenglehgl@shu.edu.cn (G.H.)
[2] Wenzhou Institute, Shanghai University, Wenzhou 325088, China
* Correspondence: sunguangling@shu.edu.cn

Abstract: As a self-supervised learning paradigm, contrastive learning has been widely used to pre-train a powerful encoder as an effective feature extractor for various downstream tasks. This process requires numerous unlabeled training data and computational resources, which makes the pre-trained encoder become the valuable intellectual property of the owner. However, the lack of a priori knowledge of downstream tasks makes it non-trivial to protect the intellectual property of the pre-trained encoder by applying conventional watermarking methods. To deal with this problem, in this paper, we introduce AWEncoder, an adversarial method for watermarking the pre-trained encoder in contrastive learning. First, as an adversarial perturbation, the watermark is generated by enforcing the training samples to be marked to deviate respective location and surround a randomly selected key image in the embedding space. Then, the watermark is embedded into the pre-trained encoder by further optimizing a joint loss function. As a result, the watermarked encoder not only performs very well for downstream tasks, but also enables us to verify its ownership by analyzing the discrepancy of output provided using the encoder as the backbone under both white-box and black-box conditions. Extensive experiments demonstrate that the proposed work enjoys quite good effectiveness and robustness on different contrastive learning algorithms and downstream tasks, which has verified the superiority and applicability of the proposed work.

Keywords: model watermarking; contrastive learning; security; deep neural networks

check for updates

Citation: Zhang, T.; Wu, H.; Lu, X.; Han, G.; Sun, G. AWEncoder: Adversarial Watermarking Pre-Trained Encoders in Contrastive Learning. *Appl. Sci.* **2023**, *13*, 3531. https://doi.org/10.3390/app13063531

Academic Editors: Konstantinos Rantos, Konstantinos Demertzis and George Drosatos

Received: 10 February 2023
Revised: 2 March 2023
Accepted: 7 March 2023
Published: 9 March 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The rapid development of deep learning (DL) has benefited a lot from a large number of diverse labeled datasets, which, however, simultaneously has become a factor hindering the further development of DL. It is due to the reason that the cost of collecting sufficient high-quality datasets with correct labels is prohibitively expensive. Fortunately, the emergence of self-supervised learning (SSL) [1] to train a powerful encoder with the unlabeled samples can effectively overcome the above obstacle. Contrastive learning [2,3], as one mainstream self-supervised learning techniques, has achieved great success in various tasks. For example, SimCLR [4] and MoCo [5] have already enabled the SSL encoders to outperform traditional supervised learning-based encoders in several downstream tasks. However, training the encoders still requires lots of unlabeled data and consumes a great deal of computing resources. Many malicious users are likely to steal the pre-trained encoders to obtain illegal income and even bring serious security risks [6,7]. Therefore, we need to find reliable solutions to protect the intellectual property of the pre-trained encoders.

As a major technique to protect the intellectual property of DL models, model watermarking [8–11] has recently been widely studied. It can be roughly divided into two categories: *white-box watermarking* and *black-box watermarking*. The former [12,13] embeds a watermark into the internal parameters [14], feature maps [15] or structures [16] of a DL

model, which needs white-box access to the watermarked model. The latter [17] usually uses backdooring [18,19] or similar techniques, such as adversarial attack [20], to mark a model. The ownership can be verified by analyzing the classification results of the marked model in correspondence to a set of carefully crafted samples. Compared with white-box watermarking, black-box watermarking is more desirable for applications, since in most cases, the model owner has no access to the internal details of the target model. However, for protecting the aforementioned encoders, the lack of a priori knowledge of downstream tasks makes it quite difficult to craft the special samples for black-box watermarking. It indicates that we cannot simply extend the existing black-box watermarking schemes to the encoders. As a result, we urgently need to develop novel watermarking schemes specifically for the pre-trained encoders.

In this paper, we introduce AWEncoder, a copyright protection method for the pre-trained encoders in contrastive learning via an adversarial watermark. In the proposed method, through optimizing an adversarial perturbation [21], we determine such a perturbation according to the pre-trained encoder that it can cluster the perturbed training samples around a randomly selected key image in the embedding space and can also be used as the secret watermark. The watermark is then embedded into the pre-trained encoder via contrastive learning by further optimizing a joint loss involving watermark embedding. In this way, even if the downstream task of the watermarked encoder is unknown to us, we can still verify its ownership under the black-box condition. Experimental results demonstrate that the ownership can be reliably verified under both white-box and black-box conditions, which is quite helpful for applications.

In summary, the main contributions of this paper include:

- We propose a novel adversarial watermarking strategy for the pre-trained encoders in the embedding space, which is more effective for watermark verification compared with previous black-box watermarking methods.
- Unlike conventional watermarking methods assuming that the downstream task of the watermarked model is the same as the original one, the proposed method does not require a priori knowledge of the downstream task. As a result, the proposed method enables us to verify the ownership under white-box and stricter black-box settings, which is more applicable to practice compared with previous ones.
- Extensive experimental results demonstrate that the proposed method has the satisfactory ability to verify the ownership of the target model and resist common watermark removal attacks, such as model fine-tuning [22] and model pruning [23], which has good application prospects.

The rest of the structure is organized as follows. We first provide preliminaries in Section 2, followed by the proposed method in Section 3. Experimental results and analysis are provided in Section 4. Finally, we conclude this paper in Section 5.

2. Preliminaries

2.1. Contrastive Learning

Contrastive learning is one of the most popular SSL techniques. SSL learns from unlabeled data, which can be regarded as an intermediate form between supervised and unsupervised learning. In this paper, we watermark a pre-trained encoder by contrastive learning. To this end, we consider two representative contrastive learning algorithms, SimCLR [4] and MoCo v2 [3]. We briefly describe them in the following. It is pointed out that the proposed method is not subject to these two algorithms.

2.1.1. SimCLR [4]

SimCLR aims to learn representations by maximizing agreement between differently augmented views of the same sample via a contrastive loss function in the latent space. Briefly, SimCLR consists of three modules that are *data augmentation*, *feature encoder* f and *projection head* g. Data augmentation transforms a sample $\mathbf{x} \in \mathcal{X}$ into two augmented views $\mathbf{x}_i$ and $\mathbf{x}_j$ treated as a positive pair. The feature encoder f extracts representation

vectors from augmented examples, for example, $\mathbf{h}_i = f(\mathbf{x}_i)$ for the augmented sample $\mathbf{x}_i$. The projection head g maps representations to the space where the contrastive loss is applied, for example, $\mathbf{z}_i = g(\mathbf{h}_i) = g(f(\mathbf{x}_i))$. With $N > 0$ samples in a mini-batch, we are able to collect $2N$ augmented samples. Two augmented samples determined from the same sample constitute a positive pair. Otherwise, it is treated as a negative pair. Let $\text{sim}(\mathbf{u}, \mathbf{v}) = \mathbf{u}^{\mathrm{T}}\mathbf{v}/||\mathbf{u}||||\mathbf{v}||$ represent the dot product between ℓ_2 normalized $\mathbf{u}$ and $\mathbf{v}$ (that is, cosine similarity). The loss function for a positive pair $(\mathbf{z}_i, \mathbf{z}_j)$ and a negative pair $(\mathbf{z}_i, \mathbf{z}_k)$ is then defined as [4]:

$$\mathcal{L}_{i,j} = -\log\left(\frac{\exp\left(\text{sim}\left(\mathbf{z}_i, \mathbf{z}_j\right)/\tau\right)}{\sum_{k=1}^{2N} \mathbb{I}[k \neq i] \cdot \exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_k)/\tau)}\right), \tag{1}$$

where τ denotes a temperature parameter and $\mathbb{I}[k \neq i] \in \{0, 1\}$ is an indicator function equal to 1 if, and only if $k \neq i$.

2.1.2. MoCo v2 [3]

As an improved version of MoCo, MoCo v2 consists of a query encoder $f_q(\mathbf{x}; \theta_q)$ and a momentum key encoder $f_k(\mathbf{x}; \theta_k)$. The better performance of MoCo v2 is attributed to a large dictionary $\mathcal{K} = \{\mathbf{k}_1, \mathbf{k}_2, ..., \mathbf{k}_{|\mathcal{K}|}\}$, which are the feature vectors of previous batches as the negative samples and are continuously updating. A batch of N samples will be encoded by f_q as feature vectors and simultaneously encoded by f_k as different augmentations. Suppose that there is a single key $\mathbf{k}_+$ in the dictionary that an encoded query $\mathbf{q}$ matches, where the contrastive loss function is defined as [3]:

$$\mathcal{L} = -\log\left(\frac{\exp\left(\mathbf{q}^{\mathrm{T}}\mathbf{k}_+/\tau\right)}{\sum_{i=1}^{|\mathcal{K}|} \exp(\mathbf{q}^{\mathrm{T}}\mathbf{k}_i/\tau)}\right). \tag{2}$$

By minimizing the above contrastive loss function, the parameters of f_q, that is, denoted by θ_q, are updated by back-propagation, but the parameters of f_k, that is, θ_k, are updated by:

$$\theta_k \leftarrow \lambda\theta_k + (1 - \lambda)\theta_q, \tag{3}$$

where $\lambda \in [0, 1)$ is a momentum coefficient.

2.2. Problem and Threats

Mainstream black-box watermarking methods mainly focus on classification-based models such as [24,25]. The *zero-bit* watermark is often embedded into the model by enforcing the model to learn the mapping relationship between the carefully crafted samples and the pre-determined labels. However, with the rise of contrastive learning, the pre-trained encoders are treated as feature extractors for various downstream tasks, and the pre-training process is relying on the SSL strategy rather than label-based supervised learning [26,27]. It indicates that traditional black-box watermarking techniques are not suitable for the pre-trained encoders in contrastive learning.

On the other hand, from the viewpoint of the adversary, he steals the encoder through an unauthorized method and adds new layers to build the specific downstream task. The adversary may not train the encoder from scratch, but he is able to modify the encoder, such as fine-tuning and pruning. In this case, even the owner of the encoder does not know what datasets and tasks will be used downstream. As a result, when watermarking an encoder, it is very necessary for the embedded watermark to be transferable and robust. Moreover, it is quite desirable that the ownership of the encoder can be verified under both the white-box condition and black-box condition. This has motivated the authors to propose a novel adversarial watermarking method.

3. Proposed Method

Figure 1 shows the general framework of the proposed method. In the following, we provide the technical details.

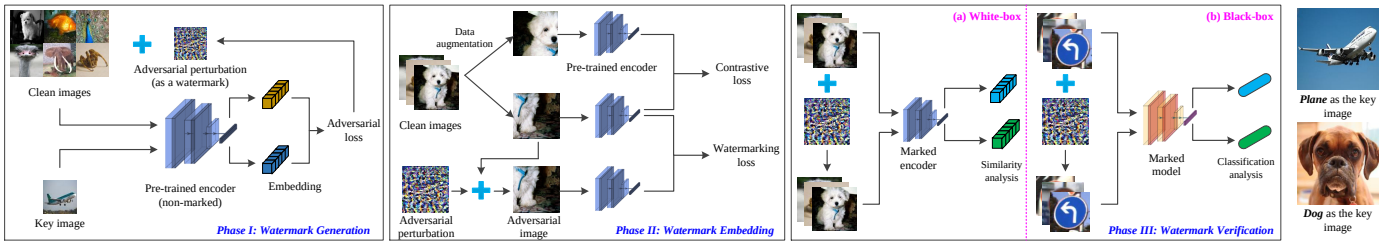


Figure 1. General framework for the proposed method. We use the same dataset for Phase I, Phase II and Phase III (**a**) so that the watermark can be successfully embedded into E_θ while avoiding degrading the encoding performance of the encoder. In Phase II, each clean image is augmented into two images, from which only one randomly selected image was used for generating the corresponding adversarial image that will be used for watermark embedding.

3.1. Watermark Generation

The first step is to generate an adversarial perturbation $\mathbf{w}_{\text{adv}}$ as the watermark using the pre-trained encoder E_θ. Since our task is encoder-oriented, the traditional perturbation optimization strategy based on classification labels is inapplicable. To deal with this problem, we use the embedding of a randomly selected image $\mathbf{x}_{\text{tar}}$ (called *key image*) extracted by E_θ, that is, $E_\theta(\mathbf{x}_{\text{tar}})$, to generate $\mathbf{w}_{\text{adv}}$ with a clean dataset $\mathcal{D}$. This enables the embedding of a perturbed image denoted by $E_\theta(\mathbf{x}_i + \mathbf{w}_{\text{adv}})$ (where $\mathbf{x}_i \in \mathcal{D}, i \in \{1, 2, ..., |\mathcal{D}|\}$) cluster around $E_\theta(\mathbf{x}_{\text{tar}})$. In other words, the distance between $E_\theta(\mathbf{x}_{\text{tar}})$ and $E_\theta(\mathbf{x}_i + \mathbf{w}_{\text{adv}})$ is expected to be as low as possible. To achieve this goal, we minimize the following loss during perturbation optimization:

$$\mathcal{L}_{\text{adv}} = \mathbb{E}_{\mathbf{x}_i \sim \mathcal{D}}[1 - \text{sim}(E_\theta(\mathbf{x}_i + \mathbf{w}_{\text{adv}}), E_\theta(\mathbf{x}_{\text{tar}}))]. \tag{4}$$

By back-propagation, $\mathbf{w}_{\text{adv}}$ can be generated without changing the parameters of E_θ. It is obvious that different $\mathbf{x}_{\text{tar}}$ results in different $\mathbf{w}_{\text{adv}}$. By using $\mathbf{x}_{\text{tar}}$ as a key, it is difficult for the adversary to forge the watermark. Some watermark generation methods by linearly superimposed fixed patterns [28] may also perturb the embedding of an image, but their effect of changing the behavior of the encoder is weaker than our method.

3.2. Watermark Embedding

After generating $\mathbf{w}_{\text{adv}}$, the next step is to embed $\mathbf{w}_{\text{adv}}$ into the pre-trained E_θ. It is realized by further training E_θ according to a combined loss $\mathcal{L}_{\text{comb}}$, which consists of two components, that is, the contrastive loss $\mathcal{L}_{\text{con}}$ and the watermarking loss $\mathcal{L}_{\text{wat}}$. In other words, we have $\mathcal{L}_{\text{comb}} = \mathcal{L}_{\text{con}} + \alpha\mathcal{L}_{\text{wat}}$, where α is a parameter balancing the loss and we use $\alpha = 40$ by default. On one hand, $\mathcal{L}_{\text{con}}$ has already been defined in Section 2, referring to Equations (1) and (2). Notice that $\mathcal{L}_{\text{con}}$ needs to be adjusted when other contrastive learning algorithms are applied. On the other hand, $\mathcal{L}_{\text{wat}}$ is defined as the Kullback–Leibler (KL) divergence [29] between the adversarial embedding and the non-adversarial embedding processed with the softmax function σ, that is,

$$\mathcal{L}_{\text{wat}} = \mathbb{E}_{\mathbf{x}'_i \sim \mathcal{D}'}\left[\text{KL}(\sigma(E_\theta(\mathbf{x}'_i)), \sigma(E_\theta(\mathbf{x}'_i + \mathbf{w}_{\text{adv}})))\right], \tag{5}$$

where $\mathbf{x}'_i$ is sampled from the augmented dataset $\mathcal{D}'$. Thus, E_θ can be watermarked by updating its parameters during training. We will use E^+_θ to represent the marked version of E_θ.

3.3. Watermark Verification

When the defender verifies whether the suspicious encoder infringes the intellectual property of the watermarked encoder, white-box scenario and black-box scenario can be considered. In the white-box scenario, the defender has the access to the target encoder $E_\theta^- \approx E_\theta^+$. Namely, the defender can directly obtain the output of E_θ^- for watermark verification. We use the average Jensen–Shannon (JS) divergence between a set of clean images and the corresponding adversarial images for similarity analysis:

$$T_{\text{sim}} = 1 - \frac{1}{|\mathcal{D}''|} \sum_{i=1}^{|\mathcal{D}''|} \text{JS}(\sigma(E_\theta^-(\mathbf{x}_i'')), \sigma(E_\theta^-(\mathbf{x}_i'' + \mathbf{w}_{\text{adv}}))), \tag{6}$$

where $\mathcal{D}''$ is the clean dataset used for watermark verification. The watermark will be successfully verified if T_{sim} is less than a threshold t_s, namely, we hope to keep T_{sim} as high as possible. In our experiments, $|\mathcal{D}''|$ is set to 1000 by default.

In a black-box scenario, given a suspicious downstream model $\mathcal{M}$, the defender wants to verify whether $\mathcal{M}$ is developed from E_θ^+. The defender should build a clean dataset $\mathcal{D}^*$ related to the downstream task. Then, the classification performance for the downstream task is analyzed by:

$$T_{\text{cls}} = 1 - \frac{1}{|\mathcal{D}^*|} \sum_{i=1}^{|\mathcal{D}^*|} \mathbb{I}[\mathcal{M}(\mathbf{x}_i^*) \neq \mathcal{M}(\mathbf{x}_i^* + \mathbf{w}_{\text{adv}})], \tag{7}$$

where function $\mathbb{I}$ returns 1 when the inside condition is true, or 0 otherwise. The watermark will be successfully verified if T_{cls} is less than a threshold t_c, namely, we hope to keep T_{cls} as high as possible. In our experiments, $|\mathcal{D}^*|$ is set to 1000 by default.

Remark 1. *As with many existing adversarial attacking methods, $\boldsymbol{w}_{adv}$ is generated by using back-propagation. The strength of $\boldsymbol{w}_{adv}$ is constrained by a threshold ϵ. To ensure that $\|\boldsymbol{w}_{adv}\|_\infty \leq \epsilon$, $\boldsymbol{w}_{adv}$ is projected on the ℓ_∞ norm-ball around $\boldsymbol{x}_i$ with radius ϵ, we set ℓ_∞-norm and ϵ = 15 by default. With this perturbation strength, it can effectively cluster the perturbed images around the key image $\boldsymbol{x}_{tar}$ in the embedding space. The implementation can be found in the released source code.*

4. Experimental Results and Analysis

In our experiments, two benchmark datasets CIFAR-10 [30] and ImageNet [31] are used to pre-train the encoder. Additionally, three benchmark datasets STL-10 [32], GTSRB [33] and ImageNet are used for the downstream task of the encoder. For ImageNet, we randomly select 30 semantic categories out for training and another 10 categories different from training out for testing. For the watermark generation phase, the pre-trained encoders are ResNet-18 and ResNet-50 [34] as the base model for SimCLR and MoCo v2 and the strength of the adversarial watermark is set as ϵ = 15. For the watermark embedding phase, we set the batch size to 50 for SimCLR and 32 for MoCo v2. Meanwhile, the total number of epochs is set to 50 and the learning rate is set to 0.003. Additionally, the key image is randomly selected from ImageNet, which belongs to the semantic category "Plane" (see Figure 1) and this image does not appear in the training set. We conduct a simulation with PyTorch, accelerated by a single RTX 3080 GPU. To validate and reproduce our experiments, a code is released via https://github.com/fc88zhang/AWEncoder (accessed on 18 July 2022).

4.1. Effectiveness

We first provide examples for the adversarial image in Figure 1 Phase III. As shown in Figure 2, due to the diversity, different images result in different degradation of the adversarial image, but the visual quality is all satisfactory, verifying the applicability of adversarial perturbation. It is admitted that one may use other perturbation strategies, which is not our main focus.

Then, we evaluate the effectiveness of the watermarked encoder in both black-box and white-box scenarios. The encoders are pre-trained on two kinds of training datasets and contrastive learning algorithms. We compare WPE [26] and the proposed AWEncoder in three downstream tasks with black-box access. The results are shown in Table 1, where "CE" is short for clean encoder and "WE" is short for watermarked encoder. The fifth column shows the classification accuracy on the downstream task before and after watermarking. The last column gives T_{cls} before/after watermarking. It is inferred that although there is a slight decrease in classification accuracy for the downstream task after watermarking, by assessing the difference between T_{cls} for CE and WE, AWEncoder is more discriminative.



(a) (b) (c) (d) (e) (f) (g) (h) (i)

Figure 2. Some examples for the adversarial image: (**a**,**d**,**g**) are clean images randomly selected from GTSRB, ImageNet, and STL-10, respectively; (**b**,**e**,**h**) are adversarial with SimCLR; (**c**,**f**,**i**) are adversarial with MoCo v2.

Table 1. Effectiveness evaluation under the black-box condition.

Pre-Trained Dataset	Encoder	Downstream Dataset	Method	Accuracy CE/WE		T_{cls} CE/WE/|CE-WE|		
ImageNet	SimCLR (ResNet-18)	ImageNet	WPE	74.5%	73.4%	25.5%	86.4%	60.9%
			AWEncoder	74.5%	71.1%	30.5%	96.1%	**65.6%**
		GTSRB	WPE	77.8%	77.5%	31.7%	90.4%	58.7%
			AWEncoder	77.8%	75.5%	16.8%	88.9%	**72.1%**
		STL-10	WPE	64.7%	64.1%	27.3%	91.6%	64.3%
			AWEncoder	64.7%	61.2%	13.7%	93.9%	**80.2%**
CIFAR-10	MoCo v2 (ResNet-50)	ImageNet	WPE	73.0%	72.9%	19.3%	80.1%	60.8%
			AWEncoder	73.0%	70.5%	30.9%	94.9%	**64.0%**
		GTSRB	WPE	84.5%	83.7%	14.9%	84.2%	69.3%
			AWEncoder	84.5%	84.0%	12.8%	90.2%	**77.4%**
		STL-10	WPE	70.5%	68.9%	11.5%	80.9%	69.4%
			AWEncoder	70.5%	68.2%	17.6%	92.7%	**75.1%**

In the white-box scenario, we verify the ownership by contrasting the embedding similarity of clean images and watermarked images. Table 2 presents how the similarity score of WE is much higher than that of CE, which indicates that the similarity score can be used for reliable verification. When the watermark is generated with an incorrect key image ("dog" in Figure 1 is used in our experiment), T_{sim} is much lower than the correct one. It indicates that AWEncoder has high security.

Table 2. Effectiveness evaluation under the white-box condition.

Model		T_{sim}	
		Correct Watermark	Incorrect Watermark
SimCLR	CE	0.14	0.11
	WE	**0.89**	0.27
MoCo v2	CE	0.26	0.25
	WE	**0.98**	0.33

In addition, the selection of key images is also an important factor for the effectiveness of watermarking. In the experiment, we randomly select an image from the semantic category of "Plane" as the key image. In order to avoid the influence caused by the randomness of the key image, we randomly select several images from the "Plane", and further select images from other semantic categories (such as "Cat" and "Dog") as the key images to test the effectiveness of the watermark. The experimental results are shown in Table 3. Selecting different key images has little influence on the effectiveness of watermarking, so in our experiment, we randomly select **Plane 1** in the category of "Plane" as the key image. We provide examples of different key images used in our experiments, as shown in Figure 3.

Table 3. Effectiveness valuation of different images as the key image.

Downstream Dataset	Setting	SimCLR WE		MoCo v2 WE	
		T_{cls}	T_{sim}	T_{cls}	T_{sim}
GTSRB	***Plane 1***	**88.9%**	**0.89**	**90.2%**	**0.98**
	Plane 2	87.8%	0.89	89.6%	0.97
	Plane 3	87.7%	0.88	90.0%	0.98
	Dog	87.4%	0.86	89.5%	0.97
	Cat	88.0%	0.88	90.1%	0.98



(a) (b) (c) (d) (e)

Figure 3. Some examples of the key images: (**a**,**c**,**d**) are different key images randomly selected in "Plane" and (**b**,**e**) are the key images selected in "Cat" and "Dog".

4.2. Uniqueness

To evaluate uniqueness, we generate forged watermarks by different methods including replacing *Plane* with *Dog* as the key image (see Figure 1), changing the value of ϵ, and replacing the pre-trained encoder based on ResNet-18 with a surrogate pre-trained encoder based on ResNet-50 [34]. Table 4 shows the results, in which the similarity score and the classification score of the correct watermark are much higher than that of the incorrect one. It indicates that the proposed method provides superior performance on watermarking uniqueness.

Table 4. Uniqueness evaluation due to different settings.

Downstream Dataset	Encoder	Setting	SimCLR (ImageNet)		MoCo v2 (CIFAR-10)	
			T_{cls}	T_{sim}	T_{cls}	T_{sim}
GTSRB	Pre-trained encoder	*Plane*, $\epsilon = 15$	**88.9%**	**0.89**	**90.2%**	**0.98**
		Plane, $\epsilon = 20$	19.4%	0.17	24.6%	0.25
		Dog, $\epsilon = 15$	29.4%	0.22	30.6%	0.27
	Surrogate encoder	*Plane*, $\epsilon = 15$	33.3%	0.27	37.1%	0.31

4.3. Robustness

In practice, adversaries may perform removal attacks, such as fine-tuning and pruning, to erase the watermark. To quantify the robustness of AWEncoder, we consider two common removal methods, that is, fine-tuning all layers (FTAL) and retraining all layers (RTAL). FTAL fine-tunes the entire encoder with the training dataset and RTAL retrains the entire encoder with the downstream training dataset. In addition, we remove the parameters with the smallest L1-norms to prune the encoder. We also verify the robustness of AWEncoder

in both white-box and black-box settings. The results in Tables 5 and 6 demonstrate that in the white-box setting, pruning has a small effect on the watermarked encoder. Although fine-tuning increases T_{sim} to a certain extent, AWEncoder can still effectively verify the ownership by applying a suitable threshold. In brief summary, AWEncoder is capable of resisting common attacks.

Table 5. Robustness against pruning (under white-box condition).

Pruning Ratio	T_{sim}			
	SimCLR CE/WE		MoCo v2 CE/WE	
-	0.14	0.89	0.26	0.98
0.2	0.16	0.88	0.28	0.96
0.4	0.20	0.84	0.29	0.92
0.6	0.25	0.76	0.33	0.85
0.8	0.28	0.70	0.35	0.80

Table 6. Robustness against fine-tuning (under white-box condition).

Fine-Tuning	T_{sim}			
	SimCLR CE/WE		MoCo v2 CE/WE	
-	0.14	0.89	0.26	0.98
FTAL	0.17	0.74	0.30	0.84
RTAL	0.24	0.68	0.34	0.76

For the black-box scenario, we compare AWEncoder with WPE by applying different attacks. The results are shown in Tables 7 and 8. It can be easily inferred that both pruning and fine-tuning will compromise the watermarking performance; however, AWEncoder is more robust than WPE against the removal attacks, which verifies the superiority of AWEncoder.

Table 7. Robustness in black-box verification against pruning.

Downstream Dataset	Pruning Ratio	Method	SimCLR (ImageNet)				
			Accuracy CE/WE		T_{cls} CE/WE/|CE-WE|		
GTSRB	0.2	WPE	79.8%	80.3%	30.8%	84.5%	53.7%
		AWEncoder	79.8%	78.0%	19.9%	85.4%	**65.5%**
	0.4	WPE	77.9%	77.2%	29.6%	79.2%	49.6%
		AWEncoder	77.9%	76.3%	19.5%	80.8%	**61.3%**
	0.6	WPE	71.2%	70.5%	27.9%	68.0%	40.1%
		AWEncoder	71.2%	68.5%	20.5%	79.6%	**59.1%**
	0.8	WPE	64.9%	64.8%	27.5%	65.7%	38.2%
		AWEncoder	64.9%	63.8%	22.1%	78.4%	**56.3%**
Downstream dataset	Pruning ratio	Method	MoCo v2 (CIFAR-10)				
GTSRB	0.2	WPE	84.0%	83.6%	15.6%	81.3%	65.7%
		AWEncoder	84.0%	82.9%	14.8%	85.4%	**70.6%**
	0.4	WPE	76.9%	78.8%	19.4%	74.2%	54.8%
		AWEncoder	76.9%	77.5%	16.3%	80.1%	**63.8%**
	0.6	WPE	71.0%	72.1%	17.5%	61.9%	44.4%
		AWEncoder	71.0%	70.0%	23.4%	76.3%	**52.9%**
	0.8	WPE	64.6%	65.7%	18.4%	53.7%	35.3%
		AWEncoder	64.6%	62.6%	22.1%	73.3%	**51.2%**

Table 8. Robustness in black-box verification against fine-tuning.

Fine-Tuning	Downstream Dataset	Method	SimCLR (ImageNet)				
			Accuracy CE/WE		T_{cls} CE/WE/|CE-WE|		
FTAL	ImageNet	WPE	74.8%	74.3%	22.3%	74.0%	51.7%
		AWEncoder	74.8%	72.1%	30.6%	92.4%	**61.8%**
	GTSRB	WPE	65.7%	66.2%	32.8%	61.7%	28.9%
		AWEncoder	65.7%	63.2%	23.9%	82.4%	**58.5%**
	STL-10	WPE	63.4%	62.7%	31.0%	86.7%	55.7%
		AWEncoder	63.4%	60.8%	14.6%	89.3%	**74.7%**
RTAL	ImageNet	WPE	94.5%	94.3%	21.6%	60.5%	38.9%
		AWEncoder	94.5%	92.7%	39.8%	81.4%	**41.6%**
	GTSRB	WPE	98.5%	98.9%	29.7%	55.0%	25.3%
		AWEncoder	98.5%	97.6%	37.6%	78.4%	**40.8%**
	STL-10	WPE	83.1%	82.2%	24.6%	50.3%	25.7%
		AWEncoder	83.1%	82.7%	34.1%	80.0%	**45.9%**
Fine-tuning	Downstream dataset	Method	MoCo v2 (CIFAR-10)				
FTAL	ImageNet	WPE	78.3%	77.2%	18.2%	77.6%	59.4%
		AWEncoder	78.3%	76.9%	30.1%	90.9%	**60.8%**
	GTSRB	WPE	89.1%	87.5%	15.9%	55.8%	39.9%
		AWEncoder	89.1%	87.7%	19.3%	85.8%	**66.5%**
	STL-10	WPE	71.3%	70.5%	13.2%	67.9%	54.7%
		AWEncoder	71.3%	70.1%	18.8%	81.1%	**62.3%**
RTAL	ImageNet	WPE	97.1%	96.6%	22.3%	61.8%	39.5%
		AWEncoder	97.1%	96.9%	35.2%	75.8%	**40.6%**
	GTSRB	WPE	98.0%	97.3%	16.0%	50.7%	34.7%
		AWEncoder	98.0%	97.1%	33.8%	82.1%	**48.3%**
	STL-10	WPE	89.2%	87.5%	18.7%	43.5%	24.8%
		AWEncoder	89.2%	85.2%	34.1%	75.2%	**41.1%**

5. Conclusions

In this paper, we proposed AWEncoder, an effective copyright protection technique for the pre-trained encoder in contrastive learning, which cannot only be applied in both white-box and black-box scenarios, but also transferred to several downstream tasks. Compared to the existing encoder watermarking, AWEncoder significantly improves the effectiveness and robustness. In the future, we will extend the proposed method to federated learning and ensemble learning, which are another two popular learning strategies widely applied in deep learning.

Author Contributions: Conceptualization, T.Z.; methodology, T.Z., G.S. and H.W.; software, T.Z. and G.S.; validation, G.S. and X.L.; supervision, H.W.; project administration, G.H. and H.W.; funding acquisition, H.W., X.L. and G.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (NSFC) under Grant No. 61902235, the Scientific and Technological Innovation Plan of Shanghai STC, No. 21511102605 Wuxi Municipal Health Commission Translational Medicine Research Project, No. ZH202102.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jing, L.; Tian, Y. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 4037–4058. [CrossRef] [PubMed]
2. Grill, J.-B.; Strub, F.; Altch, F.; Tallec, C.; Richemond, P.; Buchatskaya, E.; Doersch, C.; Avila Pires, B.; Guo, Z.; Gheshlaghi Azar, M.; et al. Bootstrap your own latent: A new approach to self-supervised learning. *Proc. Neural Inf. Process. Syst.* **2020**, *33*, 21271–21284.
3. Chen, X.; Fan, H.; Girshick, R.; He, K. Improved baselines with momentum contrastive learning. *arXiv* **2020**, arXiv:2003.04297.
4. Chen, T.; Kornblith, S.; Norouzi, M.; Hinton, G. A simple framework for contrastive learning of visual representations. In Proceedings of the 37th International Conference on Machine Learning, Virtual, 13–18 July 2020; pp. 1597–1607.
5. He, K.; Fan, H.; Wu, Y.; Xie, S.; Girshick, R. Momentum contrast for unsupervised visual representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 9726–9735.
6. Orekondy, T.; Schiele, B.; Fritz, M. Knockoff Nets: Stealing functionality of black-box models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4949–4958.
7. Chandrasekaran, V.; Chaudhuri, K.; Giacomelli, I.; Jha, S.; Yan, S. Exploring connections between active learning and model extraction. In Proceedings of the 29th USENIX Conference on Security Symposium, Boston, MA, USA, 12–14 August 2020; pp. 1309–1326.
8. Fan, L.; Ng, K.W.; Chan, C.S. Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks. *Proc. Neural Inf. Process. Syst.* **2019**, *32*, 4714–4723.
9. Zhang, J.; Chen, D.; Liao, J.; Fang, H.; Zhang, W.; Zhou, W.; Cui, H.; Yu, N. Model watermarking for image processing networks. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 12805–12812.
10. Chen, J.; Wang, J.; Peng, T.; Sun, Y.; Cheng, P.; Ji, S.; Ma, X.; Li, B.; Song, D. Copy, Right? A testing framework for copyright protection of deep learning models. *arXiv* **2021**, arXiv:2112.05588.
11. Wu, H.; Liu, G.; Yao, Y.; Zhang, X. Watermarking neural networks with watermarked images. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *31*, 2591–2601. [CrossRef]
12. Uchida, Y.; Nagai, Y.; Sakazawa, S.; Satoh, S. Embedding watermarks into deep neural networks. In Proceedings of the ACM International Conference on Multimedia Retrieval, Bucharest, Romania, 6–9 June 2017; pp. 269–277.
13. Namba, R.; Sakuma, J. Robust watermarking of neural network with exponential weighting. In Proceedings of the ACM Asia Conference on Computer and Communications Security, Auckland, New Zealand, 9–12 July 2019; pp. 228–240.
14. Wang, J.; Wu, H.; Zhang, X.; Yao, Y. Watermarking in deep neural networks via error back-propagation. In Proceedings of the IS&T Electronic Imaging, Media Watermarking, Security and Forensics, Burlingame, CA, USA, 26–30 January2020; pp. 1–8.
15. Rouhani, B.D.; Chen, H.; Koushanfar, F. Deepsigns: An end-to-end watermarking framework for ownership protection of deep neural networks. In Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems, Providence, RI, USA, 13–17 April 2019; pp. 485–497.
16. Zhao, X.; Yao, Y.; Wu, H.; Zhang, X. Structural watermarking to deep neural networks via network channel pruning. In Proceedings of the IEEE Workshop on Information Forensics and Security, Montpellier, France, 7–10 December 2021; pp. 1–6.
17. Adi, Y.; Baum, C.; Cisse, M.; Pinkas, B.; Keshet, J. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In Proceedings of the 27th USENIX Security Symposium, Baltimore, MD, USA, 15–17 August 2018; pp. 1615–1631.
18. Gu, T.; Liu, K.; Dolan-Gavitt, B.; Garg, S. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access* **2019**, *7*, 47230–47244. [CrossRef]
19. Jia, J.; Liu, Y.; Gong, N.Z. Badencoder: Backdoor attacks to pre-trained encoders in self-supervised learning. *arXiv* **2021**, arXiv:2108.00352.
20. Merrer, E.L.; Perez, P.; Trédan, G. Adversarial frontier stitching for remote neural network watermarking. *Neural Comput. Appl.* **2020**, *32*, 9233–9244. [CrossRef]
21. Moosavi-Dezfooli, S.-M.; Fawzi, A.; Fawzi, O.; Frossard, P. Universal adversarial perturbations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1765–1773.
22. Liu, K.; Dolan-Gavitt, B.; Garg, S. Fine-pruning: Defending against backdooring attacks on deep neural networks. In Proceedings of the International Symposium on Research in Attacks, Intrusions, and Defenses, Crete, Greece, 10–12 September 2018; pp. 273–294.
23. Li, H.; Kadav, A.; Durdanovic, I.; Samet, H.; Graf, H.P. Pruning filters for efficient convnets. *arXiv* **2016**, arXiv:1608.08710.
24. Jia, H.; Choquette-Choo, C.A.; Chandrasekaran, V.; Papernot, N. Entangled watermarks as a defense against model extraction. In Proceedings of the 30th USENIX Security Symposium, Virtual, 11–13 August 2021; pp. 1937–1954.
25. Zhang, J.; Gu, Z.; Jang, J.; Wu, H.; Stoecklin, M.P.; Huang, H.; Molloy, I. Protecting intellectual property of deep neural networks with watermarking. In Proceedings of the ACM Asia Conference on Computer and Communications Security, Melbourne, VI, Australia, 10–14 July 2018; pp. 159–172.
26. Wu, Y.; Qiu, H.; Zhang, T.; Li, J.; Qiu, M. Watermarking pre-trained encoders in contrastive learning. *arXiv* **2022**, arXiv:2201.08217.
27. Cong, T.; He, X.; Zhang, Y. SSLGuard: A watermarking scheme for self-supervised learning pre-trained encoders. *arXiv* **2022**, arXiv:2201.11692.
28. Chen, X.; Liu, C.; Li, B.; Lu, K.; Song, D. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv* **2017**, arXiv:1712.05526.

29. Zhang, H.; Yu, Y.; Jiao, J.; Xing, E.; Ghaoui, L.E.; Jordan, M. Theoretically principled trade-off between robustness and accuracy. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 7472–7482.
30. Krizhevsky, A.; Hinton, G. *Learning Multiple Layers of Features from Tiny Images*; Personal communication; University of Toronto: Toronto, ON, Canada, 2009.
31. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
32. Coates, A.; Ng, A.; Lee, H. An analysis of single-layer networks in unsupervised feature learning. In Proceedings of the 14th International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 11–13 April 2011; pp. 215–223.
33. Stallkamp, J.; Schlipsing, M.; Salmen, J.; Igel, C. The German traffic sign recognition benchmark: A multi-class classification competition. In Proceedings of the International Joint Conference Neural Networks, San Jose, CA, USA, 31 July–5 August 2011; pp. 1453–1460.
34. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.