Copyright-Certified Distillation Dataset: Distilling One Million Coins into One Bitcoin with Your Private Key

Tengjun Liu[1*], **Ying Chen**[1], **Wanxuan Gu**[2]
[1] School of Computer Science,Fudan University
[2] NVIDIA
{tjliu17, yingchen20}@fudan.edu.com, guwanxuan17@163.com

Abstract

The rapid development of neural network dataset distillation in recent years has provided new ideas in many areas such as continuous learning, neural network architecture search and privacy preservation. Dataset distillation is a very effective method to distill large training datasets into small data, thus ensuring that the test accuracy of models trained on their synthesized small datasets matches that of models trained on the full dataset. Thus, dataset distillation itself is commercially valuable, not only for reducing training costs, but also for compressing storage costs and significantly reducing the training costs of deep learning. However, copyright protection for dataset distillation has not been proposed yet, so we propose the first method to protect intellectual property by embedding watermarks in the dataset distillation process. Our approach not only popularizes the dataset distillation technique, but also authenticates the ownership of the distilled dataset by the models trained on that distilled dataset.

Introduction

Deep neural networks have recently made a number of breakthroughs and have had great success with well-known applications and tasks ranging from computer vision (Krizhevsky, Sutskever, and Hinton 2012; He et al. 2016) and natural language processing (Devlin et al. 2018; Doan and Reddy 2020) to other popular areas such as games (Berner et al. 2019; Silver et al. 2017), computational advertising (Zhao et al. 2020b; Xu et al. 2021), and structural biology (AlQuraishi 2019; Doan et al. 2021). However, the training of neural networks usually requires large datasets, and at the same time, the whole training process is labor-intensive and resource-intensive. Though IP protection methods (Uchida et al. 2017; Darvish Rouhani, Chen, and Koushanfar 2019; Adi et al. 2018; Quan et al. 2020) based on embedding watermarks on the models can achieve certain effectiveness on protecting the copyright of neural networks. But as the network models become larger and larger, the corresponding model weights take up more and more space. Therefore, model owners want to know if they can provide services by providing a small amount of weight information instead of sharing the total model weights.

To alleviate the pressure from large models, back in 2015, (Hinton et al. 2015) proposed a method to distill the knowledge of complex models into simple models, i.e., model distillation. However, the performance of small models obtained from model distillation can have a significant upper limit and gradually fail to meet the increasingly complex needs. And more importantly training a good deep learning model usually requires a large training dataset, so finding a way to avoid not leaking data or needing to obtain or share a large dataset first are particularly important to drive the commercialization of neural networks. To address this problem, (Wang et al. 2018) proposed a task related but orthogonal to model distillation, namely dataset distillation. They were able to distill the knowledge of large training datasets into very small sets of synthetic training images (down to one image per class), while ensuring that model trained on the distilled data would yield similar test performance to that of the model trained on the original dataset. Since the distilled dataset takes up only a small amount of space, this significantly reduces the training cost and training time for consumers. Since then, dataset distillation has become an active research topic in machine learning (Bohdal, Yang, and Hospedales 2020; Nguyen, Chen, and Lee 2020; Nguyen et al. 2021; Sucholutsky and Schonlau 2021; Zhao and Bilen 2021a,b; Zhao, Mopuri, and Bilen 2020) with various applications such as continuous learning, neural architecture search, and privacy-preserving ML. Although there are many approaches to dataset distillation, no one has investigated the copyright protection of distilled datasets.

Therefore, we propose a copyright-protectable authentication method for distilled datasets to address the above challenges. Our approach makes the model of the distilled dataset generated by training not only learn the normal functions but also embed the watermark into the training model itself for the purpose of authenticating and protecting the copyright of the distilled dataset. We improve the performance of our dataset distillation method on normal functions by imitating a multi-expert system trained on real datasets on the one hand, and make our dataset distillation method certifiable by learning watermarked training datasets to embed model watermarks based on backdoor attacks on the other hand. It is worth mentioning that we update the generated distillation dataset by learning and imitating the knowledge of expert networks trained under diversity constraints during

*Corresponding authors

synthetic dataset distillation and continuously adding back propagation information of the model loss function. Ultimately, we achieve the task of learning normal functions and embedding model watermark information using distillation datasets through a process similar to multi-task learning. Our approach provides a feasible solution for copyright protection of distillation datasets. An illustration of our approach is shown in Figure 1.

Our contributions are summarized below:

1) We present a novel problem: embedding watermark to distillation dataset for intellectual copyright protection.

2) We propose a copyright-certified distillation dataset method, which enable models trained on these dataset to simultaneously learn normal functions and embed watermarks which later used for copyright authentication of the distilled dataset.

3) We specifically design several robustness enhancements by a comprehensive examining the potential threats and perform comprehensive experiments in different settings to show the performance of our method.

Related Works

Watermarking of Deep Neural Networks. Current research on neural network model watermarking has focused on embedding watermarks into the model weights themselves or embedding specific watermarks into the model outputs. The former concept of neural network model watermarking by embedding watermarks on the model weights themselves was first proposed by Uchida et al. (Uchida et al. 2017), who embedded watermark information on the model weight feature layer to achieve copyright identity authentication. The latter concept such as backdoor watermarking, which was first proposed by (Zhang et al. 2018). They backdoor attacked a model by training dataset containing a specific trigger pattern, such that the trained model misidentifies the images containing the particular trigger pattern in the prediction phase. Since then, some researchers have found that it is possible to construct watermarking protocols with the help of cryptography (Adi et al. 2018; Zhu et al. 2020) to implement this watermarking method based on backdoor to accomplish the watermarking authentication task.

Data Distillation. The dataset distillation was first introduced by (Wang et al. 2018), who distilled datasets based on hyperparametric optimization methods (Maclaurin, Duvenaud, and Adams 2015). Subsequently, soft labeling-assisted distillation learning (Bohdal, Yang, and Hospedales 2020; Sucholutsky and Schonlau 2021), gradient matching to improve learning efficiency (Zhao, Mopuri, and Bilen 2020), data augmentation to improve learning (Zhao and Bilen 2021a), infinitely wide kernel-based optimization (Nguyen, Chen, and Lee 2020; Nguyen et al. 2021), and currently the best performance based on imitating long-path expert learning (Cazenavette et al. 2022) have been used to improve dataset distillation. At present, dataset distillation has been applied to many domains, including continuous learning (Zhao and Bilen 2021a; Wang et al. 2018; Zhao, Mopuri, and Bilen 2020), neural network architecture search cloud services (Zhao and Bilen 2021a; Zhao, Mopuri, and Bilen 2020), federated learning (Goetz and Tewari 2020; Sucholutsky and Schonlau 2020; Zhu et al. 2020), and privacy preservation of image, text, and medical imaging data (Li et al. 2020; Sucholutsky and Schonlau 2020).

Imitation Learning Imitation learning attempts to learn a good strategy by observing a collection of expert demonstrations (Osa et al. 2018; Peng et al. 2018, 2021). Behavioral cloning trains the learned policy to behave in the same way as the expert demonstrations. Some complex formulations involve policy learning with specific labels (Ross, Gordon, and Bagnell 2011), while other methods avoid any labels at all, such as distribution matching (Ho and Ermon 2016). This type of approach (behavioral cloning) has been shown to work well in offline situations (Fu et al. 2020).

Watermarking Requirements

Problem Formulation

We assume that the clean training dataset is $\mathcal{D}_{\mathrm{C}}$ and clean validation dataset is $\mathcal{D}_{\mathrm{T}}$. Let the neural network model f trained on the clean training dataset as $\mathcal{D}_{\mathrm{C}}$ be $f_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}_{\mathrm{C}}}(\boldsymbol{x}) = \boldsymbol{y}$ and parameter of the model as $\boldsymbol{\theta}$. We set the average accuracy of clean validation dataset is Acc. We let the dataset for embedding watermarking as $\mathcal{D}_{\mathrm{W}}$ containing pairs uniformly named as $(\boldsymbol{x'}, \boldsymbol{y'}) \in \mathcal{D}_{\mathrm{W}}$ and its secret key as Key. We assume the watermarked model as $\hat{f}_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}_{\mathrm{C}}\bigcup\mathcal{D}_{\mathrm{W}}}(\boldsymbol{x}) = \boldsymbol{y}$.

Functionality-Preserving Requirement. The model after embedding the watermark should perform about the same as the model trained on a clean training set. Then we can define Functionality-preserving requirement as follows.

$$\Pr(d(f(\boldsymbol{x}), \hat{f}(\boldsymbol{x})) \leq \delta)_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}_{\mathrm{C}}} \geq 1-\varepsilon \tag{1}$$

Where ε and δ represent the security level and its threshold. It can be proved by a posterior probability. In order to evaluate the functionality-preserving performance quantitatively in watermarking scheme, we transform it into :

$$\forall(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{D}_{\mathrm{C}}, d(f(\boldsymbol{x}), \hat{f}(\boldsymbol{x})) \leq \delta \tag{2}$$

Watermarking Protocol Requirement.We use $\mathrm{Verify}(\hat{f}(\boldsymbol{x'}), Key)$ as watermark validation function and $\Pr(\mathrm{Verify}(\hat{f}(\boldsymbol{x'}), Key) = 1) \geq 1-\varepsilon$ to check correctness of watermark validation.

Our problem is to generate synthetic distilled dataset $\mathcal{D}_{\mathrm{syn}}$ and make the accuracy $\hat{Acc}$ of the model $\hat{f}_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}_{\mathrm{syn}}}(\boldsymbol{x})$ have the same accuracy with Acc of the model $f(\boldsymbol{x})$ trained on clean dataset $\mathcal{D}_{\mathrm{C}}$. At the same time, we have to ensure the model trained on $\mathcal{D}_{\mathrm{syn}}$ can pass the watermark verification.

Threat Model

We assume that the user usually comes with some prudent operations after training a refined dataset containing copyright information, so we treat the operations presented below as our threats.

Model modification. Fine-tune (FT) : Many users usually will finetune the model trained by distilled dataset on running backpropagation on a smaller data set to fit their tasks. Neuron pruning (NP) : Users with restricted computational

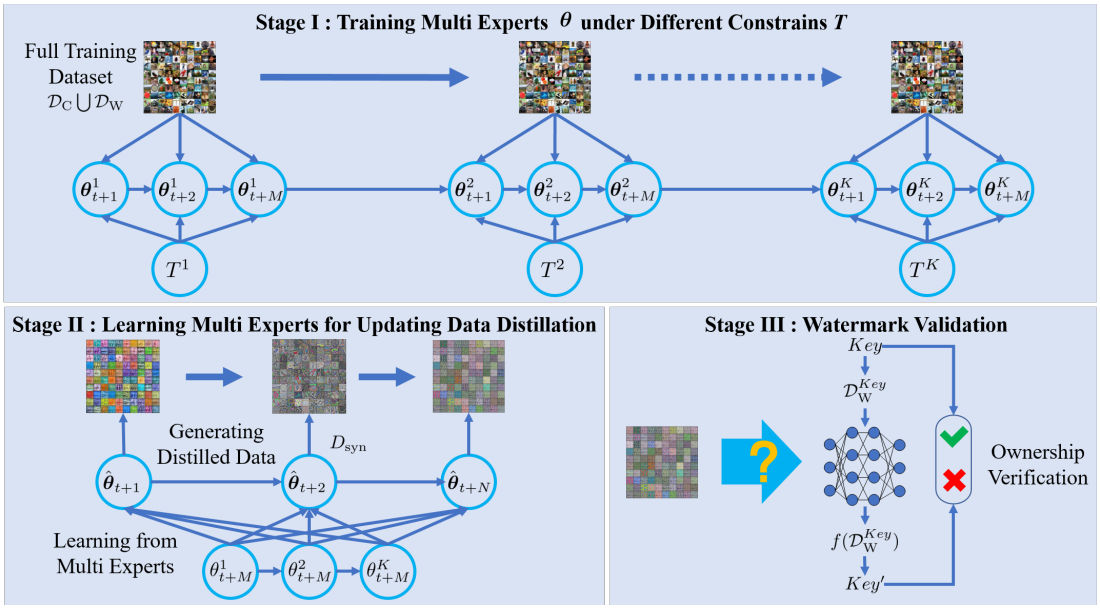

Figure 1: Training: in Stage I, we train multi expert models $\boldsymbol{\theta}$ under different constrains T for M steps; Distilling: in Stage II, we perform data distillation by sample parameters from multi experts at random timestep to update parameters of our distilled model $\hat{\boldsymbol{\theta}}$ and synthetic data $\mathcal{D}_{\text{syn}}$. Validation: in Stage III, we use our Key to generated watermarked pairs $\mathcal{D}_{\text{W}}^{Key}$ for querying the suspicious model whether has trained on our distilled dataset with no copyright permission. Then we compare our Key with the extracted Key' calculating from the inferences $f(\mathcal{D}_{\text{W}}^{Key})$ of the model to validate our copyright.

resource will pruning some less important neuron to save energy. Fine-pruning (FP) (Liu, Dolan-Gavitt, and Garg 2018): Prudent users will conduct fine-pruning to eliminate potential backdoors inserted in the pre-trained model by distilled dataset. They prune less important neurons and fine-tuning on their local dataset which are usually smaller than the original training dataset.

Watermarking Detection. The embedded watermark has to be such that a potential attacker cannot distinguish between a watermarked model and a clean model, otherwise the attacker will use a clean model to evade copyright regulation. Current watermark detection methods for neural networks are based on backdoor screening methods or reverse engineering to highlight backdoor-based watermarks.

Watermarking Rewriting. After the attacker obtains the model and knows the watermarking algorithm, they will embed their watermark into the model and declare ownership thereafter. Since the parameters of the model are hard to prove not redundant, the embedding of the new watermark can always be satisfied. Therefore, the performance requirement for watermarking is that the embedding of the new watermark does not erase the previous watermark.

Privacy Exposure. Privacy exposure means that an attacker has the ability to identify the model host without permission. That is, a watermarked DNN exposes relevant information about its author that may allow the model to be identified directly by its author information and not by its performance.

Ownership Piracy. Ownership piracy is not aimed at removing the original legal watermark, but construct anther illegal watermark without tuning its parameters or training extra learning modules. Due to cryptological protocols are often used in secure watermarking schemes, it is almost impossible to black-box forging attack to achieve ownership piracy within time complexity polynomial.

Method

Since watermarking for distilled datasets has not been studied, we propose a backdoor attack-based neural network watermarking approach with a dataset distillation method to reverse embed the model watermarking knowledge into the generated distilled dataset. As a result, our distilled dataset has not only the knowledge allowing the model to learn the normal functions but also the watermarking information satisfying the copyright authentication requirements. The workflow of our watermarking scheme is shown in Figure 1. Our watermarking scheme is composed of three stages:

Stage I : Generating Watermarked Images for Training. In this stage we mainly introduce an algorithm to generate watermarked pairs $\mathcal{D}_{\text{W}} = \{(\boldsymbol{x'_1}, \boldsymbol{y'_1}), \cdots, (\boldsymbol{x'_n}, \boldsymbol{y'_n})\}$ based on backdoor attacking to embed watermark into multi expert models. We use Key to generate a cluster of n samples with specific watermarked and labeled samples $\mathcal{D}_{\text{W}}$ as below:

$$\mathcal{D}_{\text{W}} \leftarrow \text{KenGen}(Key, (\boldsymbol{x}, \boldsymbol{y}))_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}_{\text{C}}}$$

For function KeyGen$(\cdot)$, we also achieve this by two one-way hash functions H_x and H_y taking the Key as input to generate the $\boldsymbol{x'}, \boldsymbol{y'}$ sequences. We use SHA256 as hash function, whose output can be divided into exactly 32 pixel values. So we can use these pixels to form the desired watermark pattern. And we continue to use it as input to the

hash equation to generate a new watermark pattern. We repeat the above operation until n watermarked pairs are generated. The above process can be formulated as:

$$\begin{cases} \boldsymbol{W_{i+1}} = H_x(Key, \boldsymbol{W_i}), i = 1, \cdots, n \\ \boldsymbol{x'_{i+1}} = \boldsymbol{x'_i} + \boldsymbol{W_{i+1}} \\ \boldsymbol{y'}_{i+1} = \text{mod}(H_y(Key, \boldsymbol{W_{i+1}}), Cls) \end{cases} \tag{3}$$

Where W_i is the pixel block of the i_{th} watermark pattern. We set the size of the watermark pattern to 16×16 on CIFAR-10 and CIFAR-100 datasets and to 32×32 on Tiny ImageNet just patched the watermark pattern on left-up corner of the images.

Stage II : Dataset Distillation from Multi Experts. We mainly introduce a dataset distillation algorithm which can be able to distill the knowledge of normal function and watermark needed from multi experts training on watermarked training dataset $\mathcal{D} = \mathcal{D}_{\text{C}} \bigcup \mathcal{D}_{\text{W}}$. Below is our specific design of the dataset distillation process for different requirements.

Functionality-Preserving Enhancing. Our model intends to learn two tasks, one for normal function classification training, another for watermarking tasks. Considering two tasks may not converge simultaneously, we add an extra loss by comparing with a normal function well pre-trained model $\boldsymbol{\theta}^0$, which used to guide the training process to achieve the tasks in order.

$$\mathcal{L}_{\boldsymbol{\theta}} = ||\boldsymbol{\theta} - \boldsymbol{\theta}^0||_2^2 \tag{4}$$

For functionality-preserving of each task, we use the cross entropy loss for achieving $\mathcal{L}_{\mathcal{D}_{\text{C}}}$ and $\mathcal{L}_{\mathcal{D}_{\text{W}}}$ as shown below:

$$\begin{cases} \mathcal{L}_{\mathcal{D}_{\text{C}}} = \underset{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}_{\text{C}}}{\mathcal{L}_{\mathcal{D}_{\text{C}}}} (f(\boldsymbol{x}), \boldsymbol{y}) \\ \mathcal{L}_{\mathcal{D}_{\text{W}}} = \underset{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}_{\text{W}}}{\mathcal{L}_{\mathcal{D}_{\text{W}}}} (f(\boldsymbol{x}), \boldsymbol{y}) \end{cases} \tag{5}$$

Therefore, the total loss function is:

$$\mathcal{L}_{\mathcal{D}_{\text{C}} \bigcup \mathcal{D}_{\text{W}}} = \mathcal{L}_{\mathcal{D}_{\text{C}}} + \lambda_1 \cdot \mathcal{L}_{\mathcal{D}_{\text{W}}} + \lambda_2 \cdot \mathcal{L}_{\boldsymbol{\theta}} \tag{6}$$

Resistance for Model Fine-tuning. Inspired by (Cazenavette et al. 2022), we propose a multi-task dataset distillation method learning from multi-domain expert knowledge. That is, we differ from them in that we do not only train expert networks on clean training datasets, but also on datasets that contain backdoor triggers to add watermarks to the model. In addition, we also select some of these two types of expert networks above to do additional robustness enhancement during the training process. Taking i_{th} expert for fine-tuning enhancing, we use the fine-pruning defense (Liu, Dolan-Gavitt, and Garg 2018) approach here to strengthen neurons having a relatively large impact on the model performance and get an fine tuning model $\bar{f}(\boldsymbol{x})$ from $f(\boldsymbol{x})$. Thus the loss to be optimized under fine-tuning enhancing mode of the expert model can be expressed as:

$$\bar{\mathcal{L}}_{\mathcal{D}_{\text{W}}} = \underset{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}_{\text{W}}}{\mathcal{L}_{\mathcal{D}_{\text{W}}}} (\bar{f}(\boldsymbol{x}), \boldsymbol{y}) \tag{7}$$

Then the total loss is updated to:

$$\mathcal{L}_{\mathcal{D}_{\text{C}} \bigcup \mathcal{D}_{\text{W}}} = \mathcal{L}_{\mathcal{D}_{\text{C}}} + \lambda_1 \cdot \mathcal{L}_{\mathcal{D}_{\text{W}}} + \lambda_3 \cdot \bar{\mathcal{L}}_{\mathcal{D}_{\text{W}}} + \lambda_2 \cdot \mathcal{L}_{\boldsymbol{\theta}} \tag{8}$$

Later we train the last 10 epochs of the robustness-enhanced expert training using the same procedure as the normal model training.

Resistance for Watermark Detection. Due to lack of a universal watermark detector for neural network, the evaluation of resistance for watermark detection is impractical. An intuitive measure is to observe the difference in the parameters between a watermarked model and a clean model. We empirically use λ_2 to control for the difference between them.

Resistance for Watermark Rewriting. Usually the operation of watermark overwriting is a relatively expensive operation for an attacker. For the process of watermark rewriting requires balancing both normal function learning and watermark embedding tasks, which makes it relatively difficult for an attacker to achieve without having the complete dataset. Considering the cost of watermark rewriting, we assume that the attacker does not disturb the backbone layer of the model too much, but only fine-tunes the last layer of the model. In this case, our watermarking approach can still remain effective, as proven by (Adi et al. 2018).

Resistance for privacy exposure. Constraining privacy exposure means attackers can not distinguish between different watermarked models by different Key, even if the number of candidates has been reduced to two. In order to constraining privacy exposure, Key in our watermarking protocol is generated by a probabilistic algorithm and two one-way hash function are used in KenGen$(\cdot)$. Above processes is sufficient and crucial to make adversary can only conduct a random guess, that is proved by (Li and Wang 2021).

Resistance for Ownership Piracy. We have designed a hash chain based watermarking authentication method. First we embed a watermark to the model through a backdoor attack method. Then, the images and labels of our authentication dataset need to be correctly τ steps (here τ is the threshold for ownership validation, and $\tau < n$) on the model in succession that match the hash inference we designed before the watermark authentication is considered successful. From Equation (11), when n=15, $Cls = 10$ and $\tau = 5$, the collision probability is smaller than 10^{-15}. So our one-way hash-based authentication watermarking mechanism has the ability to resist ownership pirate.

Multi Experts Guided Dataset Distillation. For all experts networks, we used the same network structure and training parameters for training but under different loss function constrains. During the training process, we save snapshot parameters of each expert at every epoch. Actually these sequences of parameters of each expert represent the theoretical upper bound for the dataset distillation task. Here we define $\boldsymbol{\theta}_t^i$ as the parameters of expert i at the training step t. Accordingly, we define $\hat{\boldsymbol{\theta}}_t$ as the parameters of watermarked model trained on synthetic distillation dataset at training step t. Inspired by (Cazenavette et al. 2022), we also encourage $\hat{\boldsymbol{\theta}}_t$ to mimic expert following long-range parameter matching strategy. At the first step of dataset distillation, we randomly pick several images from clean training dataset $\mathcal{D}_{\text{C}}$ to initial distilled data $\mathcal{D}_{\text{syn}}$ which will be generated later. During each distillation step, we randomly select one expert and sample a random timestep $\boldsymbol{\theta}_t^*$ as the initial state

of performing dataset distillation model. Then we perform N gradient descent updates on the dataset distillation model with respect to the classification loss calculating on generated synthetic dataset $\mathcal{D}_{\text{syn}}$:

$$\hat{\boldsymbol{\theta}}_{t+i+1} = \hat{\boldsymbol{\theta}}_{t+i} - \alpha \nabla \ell(\Phi(\mathcal{D}_{\text{sub}}; \hat{\boldsymbol{\theta}}_{t+i})) \tag{9}$$

Where D_{sub} are randomly sampled from $\mathcal{D}_{\text{syn}}$, and $\Phi(\cdot)$ is the differentiable augmentation technique (Karras et al. 2020; Tran et al. 2020; Zhao et al. 2020a,c) used in previous work (Cazenavette et al. 2022), and α is a trainable learning rate used to update the dataset distillation model. The differentiable augmentation technique make us be able to perform back-propagate through the augmentation layer to our synthetic dataset.

After N steps updating dataset distillation, we return to update the expert parameters for M training updates which willed used in the next round for updating the dataset distillation model. Then we use a weight matching loss to update our synthetic distilled images. Specifically, in order to bring an extra weight matching loss to enhance robustness, we train the Idx expert model under finepruning setting. Thus, the total loss is shown below:

$$\mathcal{L} = \frac{\left\| \hat{\boldsymbol{\theta}}_{t+N} - \boldsymbol{\theta}^*_{t+M} \right\|_2^2}{\left\| \boldsymbol{\theta}^*_t - \boldsymbol{\theta}^*_{t+M} \right\|_2^2} + \beta \frac{\left\| \hat{\boldsymbol{\theta}}_{t+N} - \boldsymbol{\theta}^{Idx}_{t+M} \right\|_2^2}{\left\| \boldsymbol{\theta}^{Idx}_t - \boldsymbol{\theta}^{Idx}_{t+M} \right\|_2^2}, Idx \notin * \tag{10}$$

The L_2 error can get a strong signal from experts and normalization brings some benefits to self-calibrate the magnitude difference across neurons and layers, which are also proved in (Cazenavette et al. 2022). Where β is a hyperparameter used to adjust the proportion of knowledge learned from Idx expert. Finally, we minimize this object to update the synthetic distilled dataset and the trainable learning rate α. Since our experts models can be pre-computed before distillation, which allowing for rapid distillation and experimentation.

Stage III : Performing Watermark Verfication. We propose an watermark verification algorithm Verify$(\cdot)$ to prove if a watermark Key is present in a model $\hat{\boldsymbol{\theta}}$ or not. We generate $\mathcal{D}_{\text{W}}^{Key}$ from Key and we use $\Pr\{\text{Verify}(\mathcal{D}_{\text{W}}^{Key}, Key) = 1\} \geq 1 - \varepsilon$ at least τ steps. As for an arbitrary key composing n randomly selected intergers, it almost computationally impossible to generate a Key' to collide with the true Key. Supposing we randomly forge a Key' and the probability of i_{th} interger verification probability is $1/Cls$, where Cls is the number of classifications. Therefore, the upper boundary probability of this event can be expressed by the following equation.

$$\Pr\{\text{Verify}(\mathcal{D}_{\text{W}}^{Key}, Key')\} \leq \sum_{i=\tau}^{n} \frac{i!(Cls-1)^{n-1}}{n!(n-i)!(Cls)^n} \tag{11}$$

Where τ is the threshold of ownership validation. Therefore the possibility of using an arbitrary Key' to make an effective collision is almost negligibility.

Experiments and Results

Experiment Setup

Dataset. We choose three datasets to test our method, they are CIFAR-10, CIFAR-100 and ImageNet-Tiny (Zhao and Bilen 2021b).

Evaluation and Benchmarking: Since we are the first watermarking study for dataset distillation, we compare our approach with other dataset distillation methods for comparison on one hand, and evaluate traditional watermarking performance metrics on the other hand. For the comparison performance of dataset distillation, we compare to several recent methods, such as Differentiable Siamese Augmentation (DSA) (Zhao and Bilen 2021a), concurrent works Distribution Matching (DM) (Zhao and Bilen 2021b), Aligning Features (CAFE) (Wang et al. 2022) and Matching training trajectories (MTT) (Cazenavette et al. 2022).

Network Architectures. We followed the same network architecture ConvNet used in (Cazenavette et al. 2022) for all our watermarking distillation task. The ConvNet consists of several convolutional blocks with 128 filters, Instance normalization (Ulyanov, Vedaldi, and Lempitsky 2016), RELU, and 2×2 average pooling with stride 2. And a single layer following these convolutional blocks produces the logits.

Further Details. For the evaluation of watermarking verification, we set the length n of watermarked sequences generated by KeyGen$(\cdot)$ to 10 and the threshold for ownership validation τ to one in CIFAR-10. Due to the state-of-art performances of normal function accuracy of distillation dataset methods are not great than 50%, so we just set the length n to 5 in CIFAR-100 and Tiny ImageNet to simply analyze the effectiveness of our method. As a result, their lower performance makes the distilled dataset itself less valuable at this point. Therefore, we assume that an attacker is not willing to spend too much effort to implement the ownership pirate attack. So we consider sacrificing the performance of resistance to ownership pirate on the CIFAR-100 and Tiny Image dataset by reducing the threshold τ to one to perform copyright authentication. As for the ratio of the $\frac{\mathcal{D}_{\text{W}}}{\mathcal{D}_{\text{C}}}$, we experimentally found that this ratio, if set too high, would affect the performance of the distillation dataset in normal functions, so we finally set it to 0.1 for all experiments Based on several studies (Cazenavette et al. 2022) have proved that long-range matching performs better, here we empirically set M to 500 and N to 20 for all experiments. The optimal λ_1, λ_2 and λ_3 are chosen by grid search.

Functionality-Preserving Performance

For the functionality-preserving property we require that the model trained on generated copyright-certified distillation dataset should be as accurate as a model training on clean training dataset. We perform our experiments on all three datasets with the 32 × 32 resolution for CIFAR-10 and CIFAR-100 dataset and 64 × 64 resolution for Tiny ImageNet. And we evaluate the performance of the dataset distillation on different number of distilled images per class from 1 to 50. For experiments on CIFAR-10 and CIFAR-100 we use a depth-3 ConvNet and for Tiny ImageNet we use a depth-4 ConvNet, both of them are taken from the

Dataset	Img /cls	Ratio %	DSA	DM	CAFE	MTT	Ours	Ours*	Clean
CIFAR-10	1	0.02	28.8 ±0.7	26.0 ±0.8	30.3 ±1.1	46.3 ±0.8	46.1 ±0.7	71.4 ± 0.3	
CIFAR-10	10	0.2	52.1 ±0.5	48.9 ±0.6	46.3 ±0.6	65.3 ±0.7	63.8 ± 0.5	93.3 ± 0.2	84.8 ± 0.1
CIFAR-10	50	1	60.6 ±0.5	63.0 ±0.4	55.5 ±0.6	71.6 ±0.2	70.8 ± 0.3	97.6 ± 0.2	
CIFAR-100	1	0.02	13.9 ±0.3	11.4 ±0.3	12.9 ±0.3	24.3 ±0.3	24.2 ± 0.5	31.5 ± 0.2	
CIFAR-100	10	0.2	32.3 ±0.3	29.7 ±0.3	27.8 ±0.3	40.1 ±0.4	38.4 ± 0.3	52.4 ± 0.3	56.2 ± 0.3
CIFAR-100	50	1	42.8 ±0.4	43.6 ±0.4	37.9 ±0.3	47.7 ±0.2	46.8 ± 0.3	56.3 ± 0.2	
Tiny ImageNet	1	0.02	-	3.9 ±0.2	-	8.8 ±0.3	8.5 ± 0.3	11.4 ± 0.5	
Tiny ImageNet	10	0.2	-	12.9 ±0.4	-	23.2 ±0.2	22.8 ± 0.3	29.5 ± 0.7	37.6 ± 0.4
Tiny ImageNet	50	1	-	24.1 ±0.3	-	28.8 ±0.3	27.9 ± 0.2	30.6 ± 0.6	

Table 1: The comparision between our method with other dataset distillation with respect to functionality-preserving performances. All experiments are done with a 128-width ConvNet. All other dataset distillation focus on classification task only and do not include watermarking task. Column of "Ours*" marked by $(*)$ is the watermark validation accuracy only existed in our experiments.

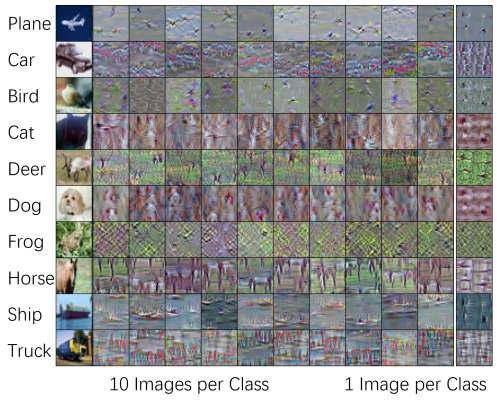

Figure 2: Generated distilled image for CIFAR-10.

Attack Method	Acc_W			
	λ_3, λ_2=0	λ_3=0	λ_2=0	$\lambda_3, \lambda_2 \neq 0$
Finetuning	83.6	86.6	87.5	90.9
Finepruing	86.5	85.3	88.5	89.0
Neuron pruning	85.6	88.0	88.7	89.5

Table 2: Ablation study on CIFAR-10 with 10 image per class dataset distillation setting. "Acc_W" means training with sythentic distilled dataset with $\lambda_1 \neq 0$ and shows the average performance of the the watermark validation accuracy. As for the third column to last column represent the combinations of our mentioned two enhancements: resistance for model fine-tuning and functionality-preserving.

open-source code (Cazenavette et al. 2022; Zhao and Bilen 2021a).

As seen in Table 1, our method outperforms all baselines and except MTT in every setting. When increasing the number of distilled images per class from 1 to 50, the classification accuracy and watermark validation accuracy both show an increase from 46.1% to 70.8% and 71.4% to 97.6%. Here we provide a comparison between one image per class and 10 images per class through visualizations shown in Figure 2. When more images are allowed to be ditilled, we can see that more expressive and diverse features of the relative class are distilled.

Although we have about 1$\sim$2% loss in accuracy compared to the best method (MTT (Cazenavette et al. 2022)), the best performance of MTT is based on zca whitening method which usually processes and stores large changes to the input image. Therefore their method does not very suitable for backdoor-based watermark embedding methods. Worth to note, on the image class distillation setting, we achieve double test accuracy comparing with the next best method (DSA (Zhao and Bilen 2021a)) on CIFAR-10 and CIFAR-100 dataset shown in the sixth the last column of the Table 1. Though the total performances on CIFAR-100 are worse than on CIFAR-10, but watermark validation accuracy of the former still large than normal function accuracy. For CIFAR-100, these distilled images are shown in Figure 3.

In Tiny ImageNet, we improve the next best method (DM (Zhao and Bilen 2021b)) to almost twice test accuracy in 10 Img/Cls and 20 Img/Cls settings shown in the seventh column of the Table 1. In addtion, we can see that the dataset distillation task on Tiny ImageNet becomes harder than CIFAR-10 and CIFAR-100, which need to handle 200 classes and resolution up to 64×64. Even the clean model trained on full clean dataset can just achieves an accuracy of 37.6±0.4%. Thus it is not weird that the accuracies of the dataset distillation methods are generally no more than 31%.

Notably, our approach not only outperforms all baselines in terms of normal functionality-preserving performances, but also achieves good performance in the task of watermarking authentication. The watermark validation accuracy on CIFAR-10, we get with an average score of 87.4±0.3%. Limited by the difficulty of training CIFAR-100 and the task of Tiny ImageNet itself, our watermark authentication accuracy is only able to reach 52.4% and 29.5% in 10 images per class setting as shown in the eleventh row of Table 1. Although the accuracy on Tiny ImageNet is low, it at least shows that our method still has some watermark embedded into the model. Because the blind guess probability is 0.01 (CIFAR-100) and 0.005 (Tiny ImageNet), respectively.

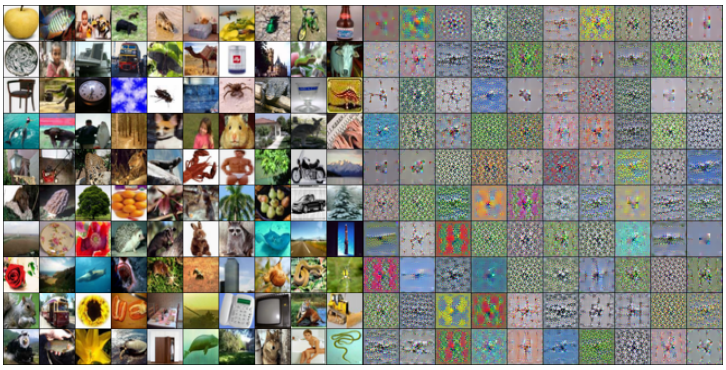

Figure 3: Generating 1 distilled image per class for CIFAR-100.

Watermark Detection

We examine our watermarked model trained on distilled model against watermark detection. Here we use the property inference attack (Ganju et al. 2018) to perform watermark detection, which also used in model watermark detection (Li and Wang 2021). Our method did not casue a significant difference between the distributions of parameters of the two models. Thus an attacker can hardly identify a model is a watermarked model or a clean model.

Ablation and Robustness

We conducted three tuning attacks : finetuning, finepruning, neuron pruning, and the overwriting attack to our proposed copyright-certified distillation dataset method. We compared the performance of the λ_2,λ_3 under different combinations to control the final loss function to adapt to different threats. λ_2 control the influence on funactionality-preserving performance guided by a pre-trained model $\boldsymbol{\theta}^0$. λ_3 control the resistance of model to fine-tuning. Our results is shown in Table 2. We observe that utilizing either one of enhancement, we improve the performance almost about 3%$\sim$4%, for example, facing finetuning we improve the performance from 83.6% to 86.6% and 87.5% shown in the second row of Table 2. From the last column of Table 2, when we adopt both enhancements, we can achieve better performance than either using alternative one. Therefore, our enhanced design in functionality-preserving and resistance to tuning make sense.

Considering overwriting attack is a relatively expensive operation. So we here simulate this attack by using another key to generate a different set of watermarked images and finetuning the last layer of the model on them for extra epochs from 10 to 100. The results are shown in Table 3. From the first and last column of Table 3, we observe that the primary watermark validation accuracy almost maintain 88% within a fluctuation about 4% and only 0.3% fluctuation on normal function accuracy. Meanwhile overwriting by finetuning the last layer show several limitations on performance, it seems model overfits after 50 epochs with no performance improvement as shown in the second column of Table 3.

We also evaluate our copyright-certified distillation dataset performs on different architectures on the CIFAR-10 with 10 distilled images per class. Here we compare our baseline ConvNet performance with ResNet, VGG, and AlexNet. Results are shown in Table 4

Scheme	Acc_W	Acc_R	Acc
Primary watermark	93.3	-	63.8
Overwriting 10 epochs	90.9	20.5	63.6
Overwriting 50 epochs	88.5	68.7	64.3
Overwriting 100 epochs	87.5	71.8	64.1

Table 3: Robustness facing overwriting tested on CIFAR-10 with 10 image per class. "Acc_W" is the watermarking validation accuracy of primary watermark. "Acc_R" represents the watermarking validation accuracy of the overwrote watermark. "Acc" is accuracy of normal function.

Dataset	Model	Acc_W	Acc
CIFAR-10 10 img/Cls	ConvNet	93.3	63.8
	ResNet	74.4	46.2
	VGG	40.9	49.5
	AlexNet	21.4	33.9

Table 4: Performances of our copyright-certified distillation dataset method tested on ConvNet, ResNet, VGG, AlexNet, under CIFAR-10 with 10 image per class setting. "Acc_W" is the watermarking validation accuracy of primary watermark. "Acc" is accuracy of normal function.

Conclusion

In this paper, we present a new problem: embedding watermarking into distilled dataset for intellectual copyright protection. And We first propose a copyright-certified distillation dataset method, which enable models trained on these dataset to simultaneously learn normal functions and embed watermarks which later used for copyright authentication of the distilled dataset. According to the requirements of watermarking, we comprehensively design several robustness enhancements and perform various experiments. In the experiments, we demonstrate that our copyright-certified distillation dataset method is secure against all possible treats.

References

Adi, Y.; Baum, C.; Cisse, M.; Pinkas, B.; and Keshet, J. 2018. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *27th USENIX Security Symposium (USENIX Security 18)*, 1615–1631.

AlQuraishi, M. 2019. AlphaFold at CASP13. *Bioinformatics*, 35(22): 4862–4865.

Berner, C.; Brockman, G.; Chan, B.; Cheung, V.; Debiak, P.; Dennison, C.; Farhi, D.; Fischer, Q.; Hashme, S.; Hesse, C.; et al. 2019. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*.

Bohdal, O.; Yang, Y.; and Hospedales, T. 2020. Flexible dataset distillation: Learn labels instead of images. *arXiv preprint arXiv:2006.08572*.

Cazenavette, G.; Wang, T.; Torralba, A.; Efros, A. A.; and Zhu, J.-Y. 2022. Dataset Distillation by Matching Training Trajectories. *arXiv preprint arXiv:2203.11932*.

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

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Doan, K. D.; Manchanda, S.; Mahapatra, S.; and Reddy, C. K. 2021. Interpretable graph similarity computation via differentiable optimal alignment of node embeddings. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 665–674.

Doan, K. D.; and Reddy, C. K. 2020. Efficient implicit unsupervised text hashing using adversarial autoencoder. In *Proceedings of The Web Conference 2020*, 684–694.

Fu, J.; Kumar, A.; Nachum, O.; Tucker, G.; and Levine, S. 2020. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*.

Ganju, K.; Wang, Q.; Yang, W.; Gunter, C. A.; and Borisov, N. 2018. Property inference attacks on fully connected neural networks using permutation invariant representations. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, 619–633.

Goetz, J.; and Tewari, A. 2020. Federated learning via synthetic data. *arXiv preprint arXiv:2008.04489*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hinton, G.; Vinyals, O.; Dean, J.; et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Ho, J.; and Ermon, S. 2016. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29.

Karras, T.; Aittala, M.; Hellsten, J.; Laine, S.; Lehtinen, J.; and Aila, T. 2020. Training generative adversarial networks with limited data. *Advances in Neural Information Processing Systems*, 33: 12104–12114.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.

Li, F.; and Wang, S. 2021. Secure watermark for deep neural networks with multi-task learning. *arXiv preprint arXiv:2103.10021*.

Li, G.; Togo, R.; Ogawa, T.; and Haseyama, M. 2020. Soft-label anonymous gastric x-ray image distillation. In *2020 IEEE International Conference on Image Processing (ICIP)*, 305–309. IEEE.

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

Maclaurin, D.; Duvenaud, D.; and Adams, R. 2015. Gradient-based hyperparameter optimization through reversible learning. In *International conference on machine learning*, 2113–2122. PMLR.

Nguyen, T.; Chen, Z.; and Lee, J. 2020. Dataset meta-learning from kernel ridge-regression. *arXiv preprint arXiv:2011.00050*.

Nguyen, T.; Novak, R.; Xiao, L.; and Lee, J. 2021. Dataset distillation with infinitely wide convolutional networks. *Advances in Neural Information Processing Systems*, 34.

Osa, T.; Pajarinen, J.; Neumann, G.; Bagnell, J. A.; Abbeel, P.; and Peters, J. 2018. An algorithmic perspective on imitation learning. *arXiv preprint arXiv:1811.06711*.

Peng, X. B.; Abbeel, P.; Levine, S.; and van de Panne, M. 2018. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions on Graphics (TOG)*, 37(4): 1–14.

Peng, X. B.; Ma, Z.; Abbeel, P.; Levine, S.; and Kanazawa, A. 2021. Amp: Adversarial motion priors for stylized physics-based character control. *ACM Transactions on Graphics (TOG)*, 40(4): 1–20.

Quan, Y.; Teng, H.; Chen, Y.; and Ji, H. 2020. Watermarking deep neural networks in image processing. *IEEE transactions on neural networks and learning systems*, 32(5): 1852–1865.

Ross, S.; Gordon, G.; and Bagnell, D. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 627–635. JMLR Workshop and Conference Proceedings.

Silver, D.; Schrittwieser, J.; Simonyan, K.; Antonoglou, I.; Huang, A.; Guez, A.; Hubert, T.; Baker, L.; Lai, M.; Bolton, A.; et al. 2017. Mastering the game of go without human knowledge. *nature*, 550(7676): 354–359.

Sucholutsky, I.; and Schonlau, M. 2020. Secdd: Efficient and secure method for remotely training neural networks. *arXiv preprint arXiv:2009.09155*.

Sucholutsky, I.; and Schonlau, M. 2021. Soft-label dataset distillation and text dataset distillation. In *2021 International Joint Conference on Neural Networks (IJCNN)*, 1–8. IEEE.

Tran, N.-T.; Tran, V.-H.; Nguyen, N.-B.; Nguyen, T.-K.; and Cheung, N.-M. 2020. Towards good practices for data augmentation in gan training. *arXiv preprint arXiv:2006.05338*, 2: 3.

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

Ulyanov, D.; Vedaldi, A.; and Lempitsky, V. 2016. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*.

Wang, K.; Zhao, B.; Peng, X.; Zhu, Z.; Yang, S.; Wang, S.; Huang, G.; Bilen, H.; Wang, X.; and You, Y. 2022. Cafe: Learning to condense dataset by aligning features. *arXiv preprint arXiv:2203.01531*.

Wang, T.; Zhu, J.-Y.; Torralba, A.; and Efros, A. A. 2018. Dataset distillation. *arXiv preprint arXiv:1811.10959*.

Xu, Z.; Li, D.; Zhao, W.; Shen, X.; Huang, T.; Li, X.; and Li, P. 2021. Agile and accurate ctr prediction model training for massive-scale online advertising systems. In *Proceedings of the 2021 International Conference on Management of Data*, 2404–2409.

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

Zhao, B.; and Bilen, H. 2021a. Dataset condensation with differentiable siamese augmentation. In *International Conference on Machine Learning*, 12674–12685. PMLR.

Zhao, B.; and Bilen, H. 2021b. Dataset Condensation with Distribution Matching. *arXiv preprint arXiv:2110.04181*.

Zhao, B.; Mopuri, K. R.; and Bilen, H. 2020. Dataset condensation with gradient matching. *arXiv preprint arXiv:2006.05929*.

Zhao, S.; Liu, Z.; Lin, J.; Zhu, J.-Y.; and Han, S. 2020a. Differentiable augmentation for data-efficient gan training. *Advances in Neural Information Processing Systems*, 33: 7559–7570.

Zhao, W.; Xie, D.; Jia, R.; Qian, Y.; Ding, R.; Sun, M.; and Li, P. 2020b. Distributed hierarchical gpu parameter server for massive scale deep learning ads systems. *Proceedings of Machine Learning and Systems*, 2: 412–428.

Zhao, Z.; Zhang, Z.; Chen, T.; Singh, S.; and Zhang, H. 2020c. Image augmentations for gan training. *arXiv preprint arXiv:2006.02595*.

Zhu, R.; Zhang, X.; Shi, M.; and Tang, Z. 2020. Secure neural network watermarking protocol against forging attack. *EURASIP Journal on Image and Video Processing*, 2020(1): 1–12.