Mixer: DNN Watermarking using Image Mixup

Kassem Kallas [1] Teddy Furon [2]

Centre Inria de l'Université de Rennes

ICASSP 2023
4 - 10 JUNE, RHODES ISLAND, GREECE

Abstract

It is crucial to protect the intellectual property rights of DNN models prior to their deployment. The DNN should perform two main tasks: its primary task and watermarking task. This paper proposes a lightweight, reliable, and secure DNN watermarking that attempts to establish strong ties between these two tasks. The samples triggering the watermarking task are generated using image Mixup either from training or testing samples. This means that there is an infinity of triggers not limited to the samples used to embed the watermark in the model at training. The extensive experiments on image classification models for different datasets as well as exposing them to a variety of attacks, show that the proposed watermarking provides protection with an adequate level of security and robustness.

Literature review

There have been various attempts to watermark Deep Neural Networks (DNNs) for intellectual property protection. In the first attempt in 2017, Uchida et al. directly embedded the watermark signal into the network weights of the host DNN [1]. Subsequently, in 2018, a black-box setting was introduced where the DNN is trained to retain a memory of specific input-label pairs known as backdoors or watermarked inputs [2]. These backdoors can be synthetic, adversarial, or benign inputs with visible or invisible overlays. It has been demonstrated that watermarking does not impact the model's accuracy and the watermark remains robust against attacks.

The authors of [3] draw an analogy with classic media watermarking, where security is defined by the attacker's inability to estimate the secret key. A popular approach to watermarking classification models is when the **Owner** injects a few inputs with unrelated labels into the training set, allowing the model to retain a memory of them [4]. Once deployed, the **Verifier** submits these samples (also known as triggers) and checks whether their outputs correspond to the unrelated labels. However, this approach suffers from several drawbacks:

1. Secret key/Watermark is a small finite set of image-label pairs.
2. Watermark can be partially erased if the **Attacker/Usurper** transforms the model.
3. Once the triggers are disclosed, the attacker can force the model to forget them.

Objectives

We propose a blind, robust, and secure black-box watermarking which has the following properties:

- **Property 1:** Secret key is an infinite trigger manifold.
- **Property 2:** Triggers used at the verification are different than the ones used at the embedding.
- **Property 3:** Equivalent to asymmetric media watermarking.

Proposed Method: Mixer

Image Mixup

$$\tilde{x} = \lambda x_i + (1-\lambda)x_j, \quad 0 < \lambda < 1. \tag{1}$$

Mixup based watermark embedding

$$\widetilde{X} = \text{clip}\left(\sum_{i=1}^{C} \lambda_i X_i + x_o\right) \tag{2}$$

where, X_i is a random image from class i, $\lambda_i > 0$ is the weight assigned to the i-th class, and x_o is a visible constant additive overlay.

For labels, we use weighting vector $\boldsymbol{\mu}$:

$$\widetilde{\mathbf{y}} = \sum_{i=1}^{C} \mu_i \mathbf{y}_i \tag{3}$$

where $\mathbf{y}_i$ is the one-hot vector of class i.

Secret keys: $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$, and x_o

Note that $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ are generated randomly from a compound of Dirichlet distributions.

Mixer: Training and Verification

Training

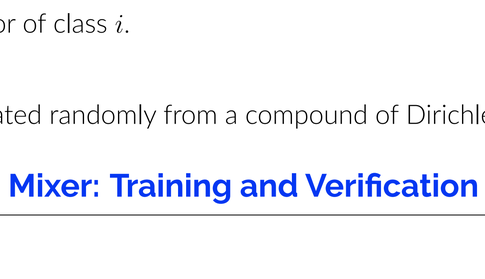
1. Generates a set S_e of n_e mixup images
2. Generate mixup labels $\widetilde{\mathbf{y}}$
3. Concatenate $(S_e, \widetilde{\mathbf{y}})$ with the training set
4. Perform training to learn the secret manifold
5. Generates a different set S_d of n_d mixup images for testing

Verification

1. Owner: gives the keys $(\boldsymbol{\lambda}, \boldsymbol{\mu}, x_o)$ to the Verifier
2. Verifier: crafts a set S_d of n_d mixup images
3. Verifier: queries the black box model using S_d not seen at training
4. Verifier: computes the ratio of images that matches using:

$$\rho_{n_d} = |\{x \in S_d | \arg\max \mu_i m(x)_i = \arg\max \mu_i\}|/n_d, \tag{4}$$

5. Verfier: grants model m ownership iff $\rho_{n_d} > \tau$



Experimental Setup

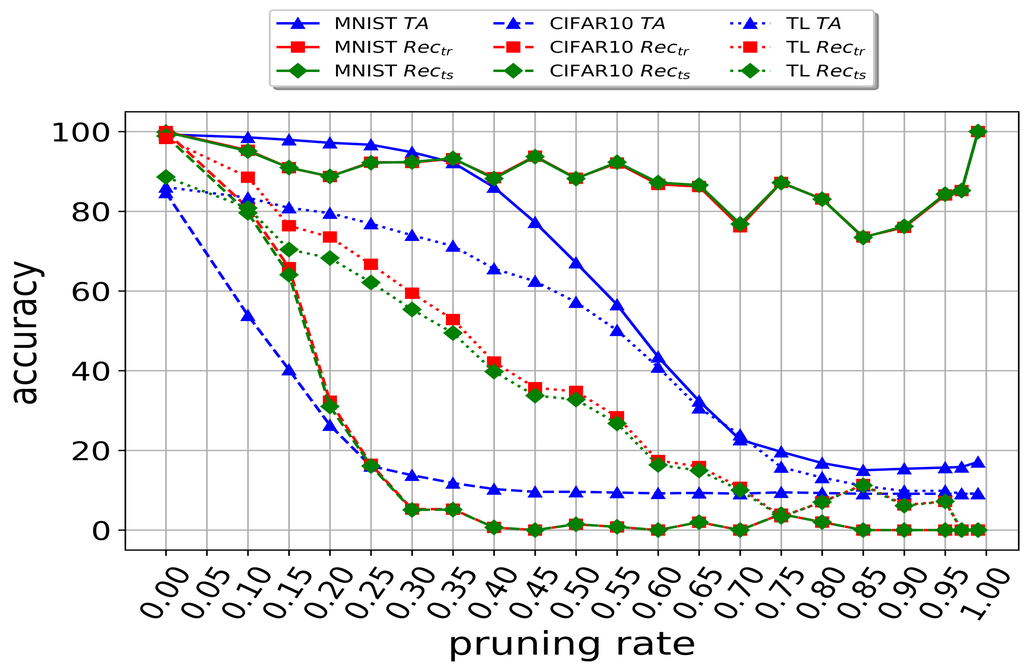
- Datasets: MNIST, CIFAR10, ImageNet →CIFAR10 for transfer learning
- DNN architectures: off-the-shelf
- Dataset split: 80% for training, 10% for validation, and 10% for fine-tuning
- Usurper attacks: fine-tuning, weights pruning ($k \in (0, 1)$), weights quantization (Dynamic, Full Uint8, Full Int8, Float16), JPEG55
- Performance evaluation criterion: TA, Rec_{tr}, Rec_{ts}, and USR
- Rec_{tr}, Rec_{ts}, and USR are computed over 1000 samples of S_e, S_d, and random keys, respectively.

Experimental Results

Results

TABLE I
PERFORMANCE RESULTS OF MIXER

Metric	Host DNN	Watermarked DNN	Fine-Tune	Dyn. Quant.	Full Uint8. Quant.	Full Int8. Quant.	Float16 Quant.	JPEG55
MNIST								
TA	99.34	99.29	99.32	99.3	99.29	8.9	99.29	99.12
Rec_{tr}	-	100	100	100	100	0.0	100	100
Rec_{ts}	10.0	99.9	100	99.9	99.9	0.0	99.9	99.9
CIFAR10								
TA	83.99	84.69	84.59	84.57	84.51	9	84.6	77.01
Rec_{tr}	-	100	100	100	100	0.0	100	90.8
Rec_{ts}	10.0	98.9	99.19	98.9	98.8	0.0	98.9	89.1
Transfer Learning								
TA	86.54	86.07	85.5	86.0	85.9	9.1	86.07	82.89
Rec_{tr}	-	88.29	95.9	98.1	98.2	0.0	98.3	93.7
Rec_{ts}	10.0	88.59	84.6	88.6	88.6	0.0	88.6	82.7



Facing a Usurper

Usurper it granted access to: model m, statistical distribution for generating $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$, and the overlay x_o. With these unrealistic assumptions: reached: $\mathbf{51.1}\%$ for MNIST, $\mathbf{38.4}\%$ for CIFAR10, and $\mathbf{39.5}\%$ for transfer learning.

Conclusions

Mixer as a DNN Watermarking method:

- Robust, Secure and does not impair the performance on the original task
- Creates a crucial inter-connection between the DNN main task and the watermarking task

Acknowledgment

We would like to thank the ANR and AID french agencies for funding Chaire SAIDA ANR-20-CHIA-0011.

References

[1] Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *USENIX Conference on Security Symposium*, 2018.

[2] Mauro Barni, Fernando Pérez-González, and Benedetta Tondi. DNN watermarking: Four challenges and a funeral. In *ACM Workshop on Information Hiding and Multimedia Security*, 2021.

[3] Kallas, Kassem, and Teddy Furon. "RoSe: A RObust and SEcure Black-Box DNN Watermarking." In *IEEE Workshop on Information Forensics and Security*. 2022.

[4] Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh. Embedding watermarks into deep neural networks. *In ACM Int. Conf. on Multimedia Retrieval*, 2017.

Author[1] Portfolio Website

www.kassemkallas.com