ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3503-4485-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICASSP48485.2024.10447550

POISONING-FREE DEFENSE AGAINST BLACK-BOX MODEL EXTRACTION

Haitian Zhang[1], *Guang Hua*[2], *Wen Yang*[1]

[1]School of Electronic Information, Wuhan University, China
[2]ICT Cluster, Singapore Institute of Technology, Singapore
{haitian.zhang, yangwen}@whu.edu.cn, ghua@ieee.org

ABSTRACT

Recent research has shown that an adversary can use a surrogate model to steal the functionality of a target deep learning model even under the black-box condition and without data curation, while the existing defense mainly relies on API poisoning to disturb the surrogate training. Unfortunately, due to poisoning, the defense is achieved at the price of fidelity loss, sacrificing the interests of honest users. To solve this problem, we propose an Adversarial Fine-Tuning (AdvFT) framework, incorporating the generative adversarial network (GAN) structure that disturbs the feature representations of out-of-distribution (OOD) queries while preserving those of in-distribution (ID) ones, circumventing the need for OOD sample collection and API poisoning. Extensive experiments verify the effectiveness of the proposed framework. Code is available at `github.com/Hatins/AdvFT`.

Index Terms— Model stealing, model extraction, adversarial fine-tuning (AdvFT), generative adversarial network (GAN), data-free model extraction (DFME), KnockoffNets.

1. INTRODUCTION

Model extraction attack [1], also known as model stealing [2] and surrogate attack [3], aims to replicate the functionality of a target deep neural network (DNN) model in a surrogate model. It can be launched under the black-box condition, where the adversary only has access to the model's API [4]. It can even be launched with the absence of surrogate query data [5], known as data-free model extraction.

Existing model extraction attacks follow a shared 3-step pipeline: i) collection of (usually unlabeled) surrogate data, ii) collection of the target model outputs from surrogate data queries, and iii) training of a surrogate model using the dataset in i) supervised by the outputs in ii). However, it is difficult to collect in-distribution (ID) data because the data distribution is confidential. Alternatively, adversaries resort to the out-of-distribution (OOD) data, composed of either natural images [4, 6, 7] or synthetic/abstract ones [5, 8, 9, 10, 11].

The corresponding defense methods can be classified into ownership-verification-based [3, 12, 13, 14] and performance-degradation-based [2, 15, 16, 17, 18, 19], respectively. The former embeds watermarks into the protected model, whereby the hidden information can transfer to surrogate models for ownership verification. Alternatively, the defense can aim at incurring performance degradation of surrogate models, constrained by the fidelity requirement, i.e., the original functionality of the protected model should be preserved for honest users. To date, most existing methods of this type are based on API poisoning, an add-on step behind the model output.

Table 1: Summary of existing model extraction attack strategies.

Type	Reference	General Idea
Nature Image	Orekondy [4]	Use the ImageNet as surrogate data
	Pal [6]	Adaptively select effective samples
	Jagielski [7]	Incorporate semi-supervised technique
Synthetic Image	Truong [5]	Steal with the data generated by GAN
	Papernot [9]	Enrich data with data augmentation
	Wang [10]	Incorporate adversarial examples
	Sanyal [11]	Train surrogate model with hard labels

We briefly describe the state-of-the-art API poisoning methods here. i) Deceptive perturbation (DP) [15]. It poisons the soft labels while keeping the top1 unaltered. It sacrifices output confidence levels and is only effective against soft-label-based model extraction. ii) Prediction poisoning (PP) [16]. It additionally poisons the top1 labels with an ID-sample-guided threshold, which further trades fidelity off for protection. iii) Adaptive misinformation (AM) [17]. It assumes that honest users always use ID queries, while the adversary does not have enough ID data. It thus first detects ID and OOD samples, followed by poisoning the outputs of the latter. iv) categorical inference poisoning (CIP) [20]. Via a fine-grained analysis, it poisons confident OOD queries strongly and uncertain OOD and ID queries mildly, jointly improving performance degradation, backdooring, and fidelity.

In this paper, we propose a novel approach to performance-degradation-based defense against model extraction. Instead of API poisoning, the defense is built in the training phase via the proposed Adversarial Fine-Tuning (AdvFT) process. Specifically, a generator is trained to synthesize OOD samples that are hard to distinguish by the original and protected model, then the protected model is fine-tuned to enlarge the feature distance between the ID and GAN-generated OOD samples. In

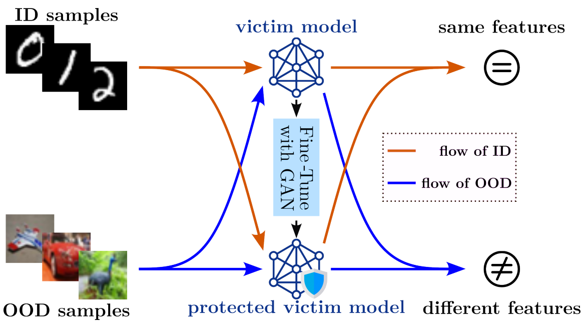


Fig. 1: Illustration of the general idea of AdvFT.

this way, AdvFT can incur surrogate performance degradation without API poisoning and achieve the highest model fidelity.

2. THE PROPOSED METHOD

2.1. Threat Model

We consider a black-box threat model in which the adversary only observes the model output. Meanwhile, we consider the more practical OOD-based model extraction attacks represented by [5, 8, 9, 10, 11]. Specifically, the adversary attempts to collect enough OOD samples for query, and the corresponding inference probabilities (or top1 labels) from the target model API are collected to supervise the surrogate training.

2.2. The Overall AdvFT Framework

The motivation of the proposed AdvFT is to enable the target model to respond differently to OOD samples while maintaining the features of ID samples, as illustrated in Fig. 1. In the red ID flow, the features extracted by the original and protected models are supposed to be same, while for the blue OOD flow, the two models are supposed to extract features differently. To achieve so, a GAN-inspired framework is utilized, as shown in Fig. 2. To train the generator, the two models are fixed, and their outputs are expected to be consistent. In the training process of the fine-tuned model, the generator is fixed and the model is fine-tuned to distinguish ID and generated samples.

2.3. Training Strategy

Training of $\mathcal{G}$. Consider the to-be-protected model as the concatenation of a feature extractor and a linear classifier, then the feature can be represented by the penultimate layer output [21]. Denote the original and protected (fine-tuned) model by $\mathcal{M}_{\mathrm{R}}$ and $\mathcal{M}_{\mathrm{F}}$, respectively, then the training of the generator $\mathcal{G}$ aims to generate x_{fake} with indistinguishable feature representations of the two models, via the minimization of

$$\mathcal{L}_{\mathcal{G}} = \|\mathcal{F}_{\mathrm{F}}(x_{\text{fake}}) - \mathcal{F}_{\mathrm{R}}(x_{\text{fake}})\|_2^2 , \qquad (1)$$

where $\mathcal{F}_{\mathrm{F}}(x_{\text{fake}})$ and $\mathcal{F}_{\mathrm{R}}(x_{\text{fake}})$ denote the penultimate layer features, respectively. $\mathcal{G}$ will be updated $\mathcal{I}_{\mathcal{G}}$ times per epoch.

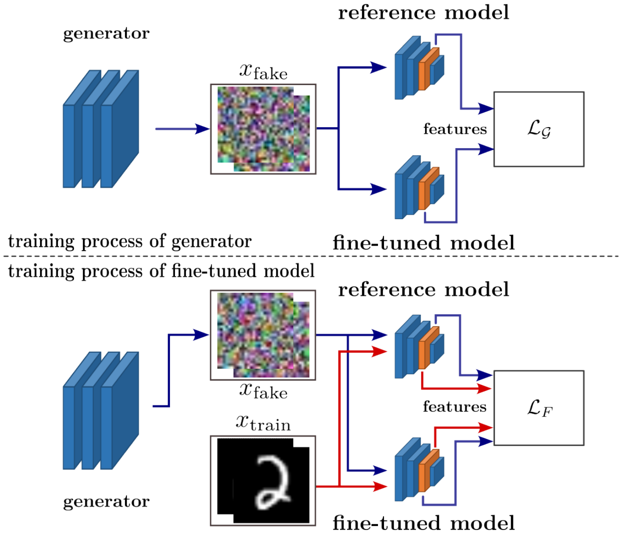


Fig. 2: Illustration of the AdvFT process.

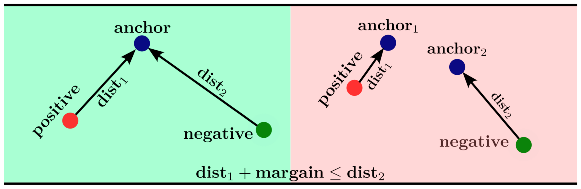


Fig. 3: Comparison of the triplet loss (green) and $\mathcal{L}_{\mathrm{F}_1}$ (pink).

Fine-Tuning of $\mathcal{M}_{\mathrm{F}}$. In the AdvFT framework, $\mathcal{M}_{\mathrm{F}}$ is not only the protected model but also one discriminator in the adversarial framework. It is supposed to generate deviated features of x_{fake} compared to those generated by the reference model. To achieve so, we incorporate the triplet loss [22] and make a modification to suit our scenario, given by

$$\mathcal{L}_{\mathrm{F}_1} = \max\Big\{ \|\mathcal{F}_{\mathrm{F}}(x_{\text{train}}) - \mathcal{F}_{\mathrm{R}}(x_{\text{train}})\|_2^2 \\ - \|\mathcal{F}_{\mathrm{F}}(x_{\text{fake}}) - \mathcal{F}_{\mathrm{R}}(x_{\text{fake}})\|_2^2 + m, 0 \Big\} , \qquad (2)$$

where x_{train} denotes the ID training sample, and $m > 0$ is a margin. Compared to the classic triplet loss which only has one anchor, the positive samples and negative samples in $\mathcal{L}_{\mathrm{F}_1}$ have their own anchor respectively, which is illustrated in Fig. 3. On the other hand, to preserve model fidelity, the original training data are used here to preserve the responses to ID samples, and we use $\mathcal{L}_{\mathrm{F}_2} = \|\mathcal{F}_{\mathrm{F}}(x_{\text{train}}) - \mathcal{F}_{\mathrm{R}}(x_{\text{train}})\|_2^2$ to ensure fidelity. Since $\mathcal{L}_{\mathrm{F}_1}$ and $\mathcal{L}_{\mathrm{F}_2}$ are on the same scale, the final loss function is simply their summation, i.e,

$$\mathcal{L}_{\mathrm{F}} = \mathcal{L}_{\mathrm{F}_1} + \mathcal{L}_{\mathrm{F}_2} . \qquad (3)$$

$\mathcal{M}_{\mathrm{F}}$ will be updated $\mathcal{I}_{\mathrm{F}}$ times per epoch. Note that in the fine-tuning process, the last layer of $\mathcal{M}_{\mathrm{F}}$ is frozen [21].

The AdvFT framework is summarized in **Algorithm 1**. Compared to the existing performance-degradation-based defense methods, AdvFT has the following unique properties.

Algorithm 1: The AdvFT Process.

```
Input: $\mathcal{M}_\mathrm{R}$, $\mathcal{G}$, $x_\mathrm{train}$, $m$, $N$, $\mathcal{I}_\mathcal{G}$, $\mathcal{I}_\mathrm{F}$;
Output: $\mathcal{M}_\mathrm{F}$;
Initialization: $\mathcal{M}_\mathrm{F} \leftarrow \mathcal{M}_\mathrm{R}$;
Freeze the last layer of $\mathcal{M}_\mathrm{F}$;
for $i = 1$ to $N$ do
    for $j = 1$ to $\mathcal{I}_\mathcal{G}$ do
        $x_\mathrm{fake} \leftarrow \mathcal{G}(\mathrm{noise})$;
        Compute $\mathcal{L}_\mathcal{G}$ using (1);
        Update $\mathcal{G}$ via $\mathcal{L}_\mathcal{G}$ and back-propagate;
    end
    for $k = 1$ to $\mathcal{I}_F$ do
        $x_\mathrm{fake} \leftarrow \mathcal{G}(\mathrm{noise})$;
        Compute $\mathcal{L}_\mathrm{F}$ using (3);
        Update $\mathcal{M}_\mathrm{F}$ via $\mathcal{L}_\mathrm{F}$ and back-propagate;
    end
end
```

First, it achieves effective defense by adjusting the internal mechanism of the model instead of relying on the extra external API poisoning step. Second, it can well preserve the original training data feature distribution via the use of $\mathcal{L}_{\mathrm{F}_2}$, achieving improved model fidelity. Third, compared to the soft-label based defense [15], AdvFT can work against both soft- and hard-label based model extraction attacks.

3. EXPERIMENTS

We consider two state-of-the-art OOD-based model extraction methods, i.e., synthetic-image-based data-free model extraction (DFME) [5] and nature-image-based KnockoffNets [4]. We assume the surrogate model structure is identical to the target model, following the setting in [17]. Among the existing performance-degradation-based defense methods, we implement the state-of-the-art DP [15] and CIP [20] for comparison.

3.1. AdvFT and DP against DFME

We first use MNIST, Fashion-MNIST (FMNIST), and CIFAR10 for evaluation against DFME, and the results are shown in the Table 2, where $\mathrm{Acc}_{\mathcal{M}_\mathrm{R}}$ and $\mathrm{Acc}_{\mathcal{M}_\mathrm{F}}$ denote the accuracies of target (victim) model before and after defense, respectively, $\mathrm{Acc}_{\mathcal{M}_\mathrm{S}}$ the accuracy of the surrogate model, Q the query times, $\mathrm{KL}_\mathrm{train}$ and $\mathrm{KL}_\mathrm{test}$ the KL divergence of the softmax probabilities of before and after defense, for training and testing sets, respectively. We observe that AdvFT reduces at least 70% accuracies for all tasks, competitive to CIP, while DP achieves about 60%. Note that DFME is a soft-label-based attack, and DP can be ineffective against hard-label-based attack, e.g., [11], but AdvFT can still have effective performance. In addition, the advantage of AdvFT over DP and CIP for fidelity control can be observed from the KL divergence differences, with 2 to 3 orders of magnitude of improvements.

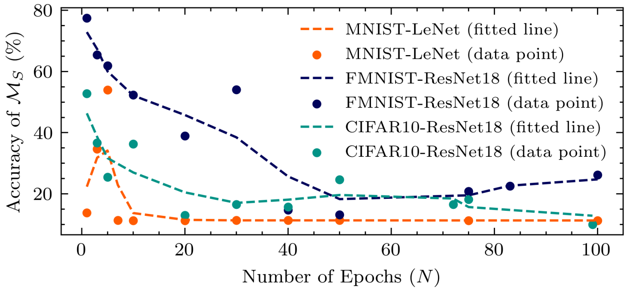


Fig. 4: DFME surrogate model accuracies vs. AdvFT epochs.

To verify the effectiveness of AdvFT in disturbing the OOD features, we test the DFME surrogate model accuracy $\mathrm{Acc}_{\mathcal{M}_\mathrm{S}}$ versus the number of epochs N used in AdvFT across different tasks, and the results are shown in Fig. 4. It can be seen that $\mathrm{Acc}_{\mathcal{M}_\mathrm{S}}$ is generally inversely related to N, indicating that the fine-tuning process can eventually make the features of OOD samples deviated from their original regions.

3.2. AdvFT and DP against KnockoffNets

KnockoffNets [4] constructs the surrogate data by adopting large public datasets, e.g, ImageNet [23] and OpenImages [24]. Compared to DFME, the distributions of its surrogate data can be more similar to those of the original training data, and thus KnockoffNets require less query times Q. The corresponding experimental results are shown in Table 3, which are consistent with those shown in Table 2.

3.3. Limitation and Potential

It can be seen from Table 3 that the proposed AdvFT yields inferior defense performance to DP and CIP against KnockoffNets for the CIFAR10 classification. The reason lies in that x_fake generated by $\mathcal{G}$ here is unable to enlarge the feature distances between the samples in ImageNet and CIFAR10, which are naturally close to each other. In other words, the current version of AdvFT is suitable for the attacking situation in which the original training data have a different distribution than the surrogate data, e.g., all the existing synthetic-image-based attacks [8, 9, 5, 10, 11], and part of natural-image-based attacks where the distributions of the training and surrogate data differs. However, we believe via the adjustment of $\mathcal{G}$, AdvFT can be tuned to solve the above problem, and that is considered as a future work.

3.4. Visualization of Feature Distribution

To further illustrate the mechanism of AdvFT and inspired by the treatment in [21], we set the number of neurons in the penultimate layer of the models to 2 for direct feature visualization. We set MNIST data as ID samples and CIFAR10

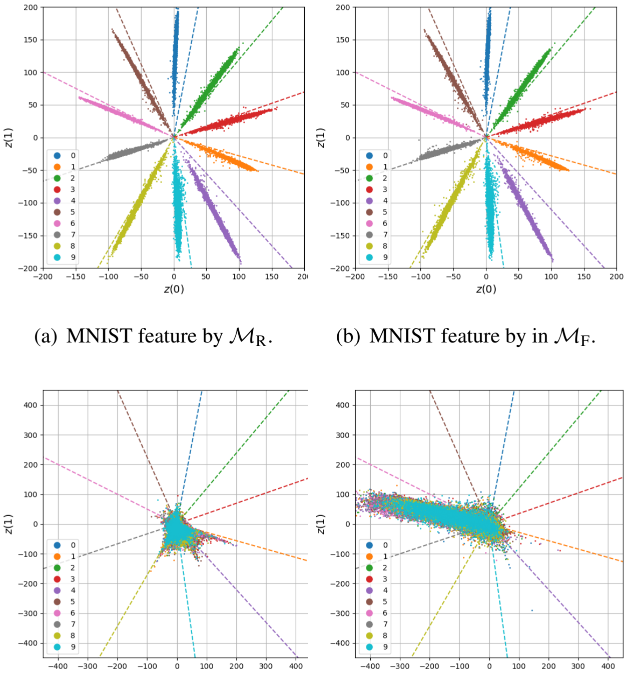
Table 2: Experiment results of AdvFT, DP [15], and CIP [20] against DFME [5], respectively.

Model	Method	Before Defense			After Defense					
		$\text{Acc}_{\mathcal{M}_\text{R}}$	$\text{Acc}_{\mathcal{M}_\text{S}}$	Q	m	N	$\text{Acc}_{\mathcal{M}_\text{F}}$	$\text{Acc}_{\mathcal{M}_\text{S}}$	KL_train	KL_test
MNIST-LeNet	AdvFT	99.41%	99.21%	$6M$	100	6	**99.43%**	**10.35%**	$\mathbf{1.2 \times 10^{-4}}$	$\mathbf{1.6 \times 10^{-3}}$
	DP				-	-	99.41%	38.43%	1.77	1.76
	CIP				-	-	99.38%	12.43%	0.15	0.23
FMNIST-ResNet18	AdvFT	92.59%	87.10%	$8M$	100	83	**92.61%**	22.53%	$\mathbf{1.9 \times 10^{-3}}$	**0.15**
	DP				-	-	92.59%	23.66%	1.75	1.72
	CIP				-	-	92.47%	**19.43%**	0.22	0.34
CIFAR10-ResNet18	AdvFT	90.33%	82.65%	$15M$	100	30	89.90%	**16.57%**	$\mathbf{6.9 \times 10^{-3}}$	**0.11**
	DP				-	-	**90.33%**	24.75%	1.73	1.67
	CIP				-	-	89.95%	18.22%	0.41	0.44

Table 3: Experimental results of AdvFT, DP [15], and CIP [20] against KnockoffNets [4], respectively.

Model	Method	Before Defense		After Defense			
		$\text{Acc}_{\mathcal{M}_\text{S}}$	Q	m	N	$\text{Acc}_{\mathcal{M}_\text{F}}$	$\text{Acc}_{\mathcal{M}_\text{S}}$
MNIST	AdvFT	98.90%	$50K$	100	75	**99.42%**	**21.33%**
	DP			-	-	99.41%	80.83%
	CIP			-	-	99.35%	25.83%
FMNIST	AdvFT	82.49%	$50K$	150	59	92.59%	**22.54%**
	DP			-	-	92.59%	41.56%
	CIP			-	-	92.52%	30.47%
CIFAR10	AdvFT	84.99%	$100K$	100	30	90.30%	85.55%
	DP			-	-	**90.33%**	77.93%
	CIP			-	-	90.15%	**49.85%**

as the OOD samples for the original model for illustration. The visualization of features are presented in Fig. 5, where the left figures were drawn based on the model before fine-tuning, while the right figures are based on the model after fine-tuning. We can see that the features of the training set remain almost the same while the features of the OOD samples have a huge offset. This verifies that AdvFT can achieve the performance-degradation defense while preserving high fidelity.



(a) MNIST feature by $\mathcal{M}_\text{R}$. (b) MNIST feature by in $\mathcal{M}_\text{F}$.

(c) CIFAR10 feature by $\mathcal{M}_\text{R}$. (d) CIFAR10 feature by $\mathcal{M}_\text{F}$.

Fig. 5: Visualization of ID (MNIST) and OOD (CIFAR10) feature representations before and after AdvFT, where $\mathcal{M}_\text{R}$ is trained by the MNIST using ResNet18 with an accuracy of 99.46%, and $\mathcal{M}_\text{F}$ is obtained from $\mathcal{M}_\text{R}$ undergone AdvFT with an accuracy of 99.47%.

4. CONCLUSION

In this paper, we have proposed an adversarial fine-tuning framework, termed AdvFT, to protect deep classification models from black-box model stealing attacks. It incorporates the GAN structure and features three components including a generator, the unprotected victim model (reference), and the protected (fine-tuned) model initialized by the reference. The generator creates surrogate data indistinguishable by the two models, while the protected model is fine-tuned in contrast to distinguish the original ID training data and the generated ones. The proposed AdvFT differs from most of the existing performance-degradation-based defense methods by avoiding the extra API poisoning process and is the first method that achieves the state-of-the-art defense via the modification of the model's internal mechanism. More importantly, this is achieved with simultaneously improved model fidelity, thanks to the proposed fidelity loss. The advantages of the proposed framework have been verified by extensive experiments using different models under multiple classification tasks. In future, it is worth investigating the modification of AdvFT for the more challenging situation with closely distributed ID training and OOD surrogate data.

5. REFERENCES

[1] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, “Stealing machine learning models via prediction APIs,” in *Proc. USENIX Security*, 2016, pp. 601–618.

[2] M. Juuti, S. Szyller, S. Marchal, and N. Asokan, “PRADA: Protecting against DNN model stealing attacks,” in *Proc. IEEE European Symposium on Security and Privacy*, 2019, pp. 512–527.

[3] H. Jia, C. A. Choquette-Choo, V. Chandrasekaran, and N. Papernot, “Entangled watermarks as a defense against model extraction,” in *Proc. USENIX Security*, Aug. 2021, pp. 1937–1954.

[4] T. Orekondy, B. Schiele, and M. Fritz, “Knockoff nets: Stealing functionality of black-box models,” in *Proc. IEEE/CVF CVPR*, 2019, pp. 4954–4963.

[5] J.-B. Truong, P. Maini, R. J. Walls, and N. Papernot, “Data-free model extraction,” in *Proc. IEEE/CVF CVPR*, 2021, pp. 4771–4780.

[6] S. Pal, Y. Gupta, A. Shukla, A. Kanade, S. Shevade, and V. Ganapathy, “Activethief: Model extraction using active learning and unannotated public data,” in *Proc. AAAI*, vol. 34, 2020, pp. 865–872.

[7] M. Jagielski, N. Carlini, D. Berthelot, A. Kurakin, and N. Papernot, “High accuracy and high fidelity extraction of neural networks,” in *Proc. USENIX Security*, 2020, pp. 1345–1362.

[8] S. Kariyappa, A. Prakash, and M. K. Qureshi, “MAZE: Data-free model stealing attack using zeroth-order gradient estimation,” in *Proc. IEEE/CVF CVPR*, 2021, pp. 13 814–13 823.

[9] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, “Practical black-box attacks against machine learning,” in *Proc. ACM Asia CCS*, Abu Dhabi, United Arab Emirates, 2017, pp. 506–519.

[10] W. Wang, B. Yin, T. Yao, L. Zhang, Y. Fu, S. Ding, J. Li, F. Huang, and X. Xue, “Delving into data: Effectively substitute training for black-box attack,” in *Proc. IEEE/CVF CVPR*, 2021, pp. 4761–4770.

[11] S. Sanyal, S. Addepalli, and R. V. Babu, “Towards data-free model stealing in a hard label setting,” in *Proc. IEEE/CVF CVPR*, 2022, pp. 15 284–15 293.

[12] S. Szyller, B. G. Atli, S. Marchal, and N. Asokan, “DAWN: Dynamic adversarial watermarking of neural networks,” in *Proc. ACM Int. Conf. Multimedia*, 2021, pp. 4417–4425.

[13] Z. Peng, S. Li, G. Chen, C. Zhang, H. Zhu, and M. Xue, “Fingerprinting deep neural networks globally via universal adversarial perturbations,” in *Proc. IEEE/CVF CVPR*, 2022, pp. 13 430–13 439.

[14] J. Chen, J. Wang, T. Peng, Y. Sun, P. Cheng, S. Ji, X. Ma, B. Li, and D. Song, “Copy, right? a testing framework for copyright protection of deep learning models,” in *2022 IEEE Symposium on Security and Privacy*, 2022, pp. 824–841.

[15] T. Lee, B. Edwards, I. Molloy, and D. Su, “Defending against neural network model stealing attacks using deceptive perturbations,” in *Proc. IEEE Security and Privacy Workshops*, 2019, pp. 43–49.

[16] T. Orekondy, B. Schiele, and M. Fritz, “Prediction poisoning: Towards defenses against dnn model stealing attacks,” in *Proc. ICLR*, 2019.

[17] S. Kariyappa and M. K. Qureshi, “Defending against model stealing attacks with adaptive misinformation,” in *Proc. IEEE/CVF CVPR*, 2020, pp. 770–778.

[18] H. Yan, X. Li, H. Li, J. Li, W. Sun, and F. Li, “Monitoring-based differential privacy mechanism against query flooding-based model extraction attack,” *IEEE Trans. Dependable Secure Comput.*, 2021.

[19] B. G. Atli, S. Szyller, M. Juuti, S. Marchal, and N. Asokan, “Extraction of complex dnn models: Real threat or boogeyman?” in *Proc. AAAI Workshop*, 2020, pp. 42–57.

[20] H. Zhang, G. Hua, X. Wang, H. Jiang, and W. Yang, “Categorical inference poisoning: Verifiable defense against black-box dnn model stealing without constraining surrogate data and query times,” *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 1473–1486, 2023.

[21] F. Pernici, M. Bruni, C. Baecchi, and A. D. Bimbo, “Regular polytope networks,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 9, pp. 4373–4387, 2022.

[22] F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” in *Proc. IEEE/CVF CVPR*, 2015, pp. 815–823.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Proc. NeurIPS*, vol. 25, 2012.

[24] A. Kuznetsova *et al.*, “The open images dataset v4,” *International Journal of Computer Vision*, vol. 128, no. 7, pp. 1956–1981, 2020.