Training A Secure Model against Data-Free Model Extraction

Zhenyi Wang[1], Li Shen[2*], Junfeng Guo[1], Tiehang Duan[3], Siyu Luan[4], Tongliang Liu[5], and Mingchen Gao[6]

[1] University of Maryland, College Park, USA `wangzhenyineu@gmail.com`, `junfeng.guo.dc@gmail.com`
[2] Sun Yat-sen University, China `mathshenli@gmail.com`
[3] Mayo Clinic, USA `tiehang.duan@gmail.com`
[4] University of Copenhagen, Denmark `siyuluan.ku@gmail.com`
[5] The University of Sydney, Australia `tongliang.liu@sydney.edu.au`
[6] University at Buffalo, USA `mgao8@buffalo.edu`

Abstract. The objective of data-free model extraction (DFME) is to acquire a pre-trained black-box model solely through query access, without any knowledge of the training data used for the victim model. Defending against DFME is challenging because the attack query data distribution and the attacker's strategy remain undisclosed to the defender beforehand. However, existing defense methods: (1) are computational and memory inefficient; or (2) can only provide evidence of model theft after the model has already been stolen. To address these limitations, we thus propose an Attack-Aware and Uncertainty-Guided (AAUG) defense method against DFME. AAUG is designed to effectively thwart DFME attacks while concurrently enhancing deployment efficiency. The key strategy involves introducing random weight perturbations to the victim model's weights during predictions for various inputs. During defensive training, the weights perturbations are maximized on simulated out-of-distribution (OOD) data to heighten the challenge of model theft, while being minimized on in-distribution (ID) training data to preserve model utility. Additionally, we formulate an attack-aware defensive training objective function, reinforcing the model's resistance to theft attempts. Extensive experiments on defending against both soft-label and hard-label DFME attacks demonstrate the effectiveness of AAUG. In particular, AAUG significantly reduces the accuracy of the clone model and is substantially more efficient than existing defense methods.

1 Introduction

Deep-learning models are becoming increasingly prevalent in commercial systems, offering services through APIs that grant users only black-box access to the underlying pre-trained models and forming intellectual property [14, 57, 58]. Users can interact with these models solely by making queries without being

[*] Correspondence author.

privy to details such as model weights or architectures. To harness the significant business value offered by these commercial models, model extraction (ME) attacks aim to extract/steal these black-box pre-trained models through query access alone. Recently, research on data-free model extraction (DFME) has uncovered that attackers can extract victim pre-trained models without any knowledge of the actual training data used for the victim model. On the other hand, the owners of these commercial systems invest substantial resources, including time and money, into training and fine-tuning their models. Their competitive advantages can be severely compromised if attackers manage to steal these valuable pre-trained models with DFME [11].

Protecting against DFME is a challenging task due to the fact that the attacker's query data distribution and strategies are unknown to the defender beforehand. Nonetheless, the existing defense methods face several limitations: (1) *Computational and Memory Inefficiency*: Some current defense methods are associated with significant computational and memory overhead since they need test time perturbation with multiple backpropagation and ensemble of multiple models [23, 36, 41]. These resource-intensive processes may not be practical for many real-world applications. (2) *Limited to Detecting/Verifying Theft*: Some defense mechanisms can, at best, only provide evidence of model theft after it has already occurred [1, 18, 19, 34, 49]. These reactive measures could not prevent the initial extraction of the model, allowing potential misuse of the stolen model.

In essence, the challenge in defending against DFME lies in finding efficient, robust, and proactive methods that can prevent model extraction from happening in the first place. In contrast to these existing approaches, our objective is to enhance security against DFME by training a pre-trained model that is inherently resistant to model theft. Consequently, our approach has the potential to significantly improve the efficiency during deployment.

In response to the limitations of existing defense methods, we propose an innovative defense approach named as Attack-Aware and Uncertainty-Guided (AAUG) defense. Specifically, we introduce learnable random weight perturbations to the weights of the victim model. These perturbations enable the model to produce predictions for various inputs by utilizing different functions represented by the perturbed model weights. The addition of random weight perturbations significantly expands the function space of the deployed victim model. This expanded function space renders the model extraordinarily challenging to steal, as attackers are confronted with an uncountable multitude of potential functions to steal. This is in stark contrast to conventional approaches that rely on a single model (function) during deployment, which is considerably more susceptible to theft. However, applying uniform random weight perturbations of the same scale to model weights for all inputs presents challenges. Excessive weight perturbation can significantly diminish the model utility, while insufficient perturbation is ineffective in safeguarding against DFME. Instead, our approach focuses on two key aspects: (i) Maximizing weight uncertainty on simulated out-of-distribution (OOD) data to introduce ambiguity in predictions when facing simulated attack queries. (ii) Minimizing weight uncertainty on in-distribution

(ID) data to preserve the utility of the victim model. Furthermore, we introduce a novel attack-aware defensive training objective that involves bi-level optimization. Specifically, we aim to minimize the performance on the ID training data through upper-level optimization, after simulating attacker behavior by training on the simulated OOD data in lower-level optimization. This approach ensures that the victim model becomes highly resistant to theft.

We perform extensive experiments on both soft-label and hard-label DFME scenarios and show that our method significantly outperforms baselines by a large margin. Specifically, AAUG reduces the clone model accuracy by up to 40%, surpassing the best baseline by up to 32% on CIFAR10. AAUG significantly reduces the clone model accuracy by up to 37%, outperforming the best defense method by up to 18% on CIFAR100. AAUG significantly reduces the clone model accuracy by more than 20% on ImageNet-100, exceeding the best baseline performance by more than 14%. Notably, AAUG is far more efficient than existing defense methods and achieves $25 \sim 103\times$ speed up compared to baseline methods. At the same time, AAUG maintains the victim model utility with slightly better performance compared to other defense methods. Furthermore, we extend our method to defend against traditional data-based model extraction and stealing self-supervised learning models. The results show that AAUG can significantly reduce the clone model quality extracted by attackers.

We summarize our main contributions as follows:

- We propose a novel and efficient DFME defense framework by learning a secure victim model.
- To implement the proposed framework, we introduce an attack-aware and uncertainty-guided defensive training objective designed to protect the victim model, making it substantially more resistant to model extraction.
- We perform comprehensive experiments on defending against both soft-label and hard-label DFME attacks. Results show that AAUG substantially reduces the accuracy of clone model and is significantly more efficient than existing defense methods. Further, AAUG can be extended to defend against data-based model extraction and protect self-supervised learning models.

2 Related Work

2.1 Data-Free Model Extraction Attack

The goal of model extraction (ME) is to acquire and replicate the functionality of a target black-box pre-trained model. Existing ME approaches can be classified into two classes: (1) *data-based model extraction* (DBME), wherein attackers can utilize real dataset associated with the pre-trained model to steal the victim model [5, 17, 20, 29, 33, 37, 39, 43, 51, 55]. (2) *data-free model extraction* (DFME). DFME represents a further relaxation of attack data requirements. It eliminates the necessity for knowledge about the training dataset distribution and any relevant surrogate dataset associated with the victim pre-trained model. DFME presents two distinct settings. (1) *Hard-Label Setting*: In this scenario, the victim model only provides users with its top-1 class prediction, as exemplified by works such as ZSDB3KD [59] and DFMS-HL [47]. (2) *Soft-Label Setting*: In this setting, the victim model shares its softmax probability scores with users [15,61], as seen in approaches like MAZE [22] and DFME [52]. DFME becomes an active model extraction research problem due to its low requirement of attack query data. However, this also increases the difficulty of defending against DFME due to the fact that the attack query data distribution could be any distribution, and remains unknown in advance.

2.2 Model Extraction Defense

We classify existing model extraction defense into two classes: (1) model extraction prevention defense; (2) model extraction detection/verification defense.
Model Extraction Prevention Defense aims to *prevent* model extraction (ME) attacks before they can occur. There are several methods within this category. (1) *Output Perturbation-based methods* : These methods focus on perturbing model output probabilities. For instance, Deceptive Perturbation [27] softens model outputs in an attempt to hinder model extraction, but it is ineffective, as demonstrated in previous research [36]. Similarly, methods like Prediction Poisoning (P-poison) [41] and GRAD [36] undertake complex optimization during deployment. (2) *Ensemble-based methods* : Ensemble of Diverse Models (EDM) [23] is an example in this category. It trains a diverse set of models to create a discontinuous decision boundary.

However, these methods have their limitations when applied to defense against DFME: (1) *Output Perturbation-based methods*: P-poison [41] and GRAD [36] significantly increase computation and memory costs since they need multiple backpropagation during test time. Adaptive misinformation (AM) [24] adaptively delivers incorrect predictions to queries identified as OOD, while ID queries are provided with accurate predictions. There are two major differences between AM [24] and our AAUG. First, AM needs to train and maintain two models, one is the victim model and another one is a poison model, which adaptively adds misinformation to the model outputs. Second, AM is a deterministic defense method. In contrast, AAUG is an uncertainty-based random defense method, which varies for different query inputs, providing stronger defense compared to deterministic defense. (2) *Ensemble-based methods* are memory inefficient due to the requirement of storage of multiple pre-trained models. (3) *Some methods require the knowledge of attack query dataset.* GRAD requires knowledge of the attack query set, while ensemble-based methods like EDM [23] depend on knowing the prior distribution of the attack query distribution, which remains unknown to defenders in the data-free setting.
Model Extraction Detection/Verification Defense focus on detecting or verifying the theft of a pre-trained model after it has already been stolen. Existing methods can be classified into two classes: (1) *Detection-based methods* :

Methods like [21,42] aim to identify attack queries from benign ones. These methods need to know the prior attack query data distribution. Yet, attackers can easily circumvent and evade these detection by adaptively changing their query data distribution, making these defenses ineffective in practice. (2) *Verification-based methods* typically operate through methods such as watermark-based [1] methods [18,49], proof-of-learning [19], and dataset inference [34]. However, it's crucial to note that these verification defenses operate exclusively to verify whether a model has been stolen after the fact. It does not possess the capability to prevent the initial model theft.

Another line of related defense methods to our approach is random defense against adversarial attacks [12, 13, 31, 56]. A more detailed discussion and comparison of these methods can be found in the Appendix.

Our method falls under the category of model extraction *prevention defense*, which can substantially reduce the clone model accuracy. In contrast, detection/verification defense can only *passively* detect and verify model extraction, but could not reduce clone model accuracy and prevent model extraction.

3 Preliminary

3.1 DFME Attack

Attacker Knowledge and Goal Given a victim model $\mathcal{V}(\boldsymbol{x}, \boldsymbol{\phi}_V)$ parameterized by $\boldsymbol{\phi}_V$. The attacker uses some data samples $\{\boldsymbol{x}_i\}_{i=1}^N$ to query the victim model to obtain the corresponding outputs $\{\boldsymbol{y}_i\}_{i=1}^N$ and construct a new dataset $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^N$. Then, the attacker uses this dataset to train a clone model $\mathcal{C}(\boldsymbol{x}, \boldsymbol{\phi}_C)$ parameterized by $\boldsymbol{\phi}_C$. Given an in-distribution test dataset associated with $\mathcal{V}(\boldsymbol{x}, \boldsymbol{\phi}_V)$. The attacker's goal is to replicate the functionality of the black-box victim model with the clone model, obtaining test accuracy on the in-distribution test set as high as possible. The attacker does not know the victim model training dataset, architecture, training algorithm, and hyperparameters. Below, we present two existing representative DFME attack settings and related works, including soft-label DFME, e.g., MAZE [22], and DFME [52]; and hard-label DFME, e.g., DFMS-HL [47].

Soft-Label DFME. MAZE [22] and DFME [52] are two representative soft-label DFME methods. Their basic idea is to use a generator to synthesize pseudo data to query the victim model. Then, the attacker learns a clone model to mimic the outputs of the victim model. The pseudo data generator $G(\boldsymbol{\xi}, \boldsymbol{\phi}_G)$ parameterized by parameters $\boldsymbol{\phi}_G$ takes random noise $\boldsymbol{\xi} \sim \mathcal{N}(0, I)$ as input and outputs pseudo data $\boldsymbol{x} = G(\boldsymbol{\xi}, \boldsymbol{\phi}_G)$, which is used for querying the victim model. The attacker minimizes the following loss function for clone model and data generator alternatively to steal the victim model:

$$\mathcal{L}_C = \mathbb{KL}(\mathcal{V}(\boldsymbol{x}, \boldsymbol{\phi}_V), \mathcal{C}(\boldsymbol{x}, \boldsymbol{\phi}_C)), \qquad \mathcal{L}_G = -\mathbb{KL}(\mathcal{V}(\boldsymbol{x}, \boldsymbol{\phi}_V), \mathcal{C}(\boldsymbol{x}, \boldsymbol{\phi}_C)) \tag{1}$$

Since the victim model only provides query access without providing weights and architecture. The gradient could not be directly backpropagated to update generator parameters. Thus, they adopt the zeroth-order gradient method [30]:

$$\nabla_{\boldsymbol{x}} \mathcal{L}_G = \sum_{i=1}^{i=m} \frac{L_G(\boldsymbol{x} + \mu_i \Delta, \boldsymbol{\phi}_V) - L_G(\boldsymbol{x}, \boldsymbol{\phi}_V)}{\Delta} \mu_i \tag{2}$$

where μ_i is random perturbation directions and Δ is small step size. Then, the attacker performs the gradient descent step by utilizing the above estimated zero-order gradient to produce the clone model.
Hard-Label DFME. Due to the space limitations, we put the state-of-the-art hard-label DFME method, DFMS-HL [47] in Appendix.

3.2 Defense against DFME

Defender Knowledge and Goal. Different from the attacker's goal, the defender's goal is to minimize the in-distribution test accuracy that the attacker can get. Many established model extraction defenses [6, 23, 24] typically operate under the assumption that the attack query data are out-of-distribution (OOD) data. This assumption is based on the practical challenges associated with acquiring in-distribution (ID) training data in real-world scenarios. First, in numerous applications, ID data is strictly safeguarded as confidential and private due to business considerations and the high cost associated with data labeling. Second, model owners typically do not release ID data due to significant privacy and security concerns. Consequently, we align with these prevailing assumptions in our approach. Furthermore, prevalent techniques in both data-based model extraction [40, 43] and data-free model extraction [22, 47, 52] commonly rely on OOD data for extracting victim models. OOD data is favored because gathering OOD data is considerably more cost-effective and accessible for attackers. In many cases, it is even impossible to get ID data since the API is only trained by internal private datasets, e.g., the speech recognition systems like Siri and Cortana [32]. In addition, defender needs to maintain the victim model utility (i.e., the in-distribution test accuracy on the ID test set) so that the victim model still performs well on benign inputs after applying the defense strategy.

4 Methodology

We discuss the base model training in most existing works in Section 4.1. We discuss its limitations and provide solution to defend against DFME with: (1) uncertainty-guided training; (2) attack-aware defensive training in Section 4.2. Finally, we explain why AAUG can defend against DFME in Section 4.3.

4.1 Base Model Training

In most existing works, the pre-trained black-box model (victim model) is optimized by the following loss function:

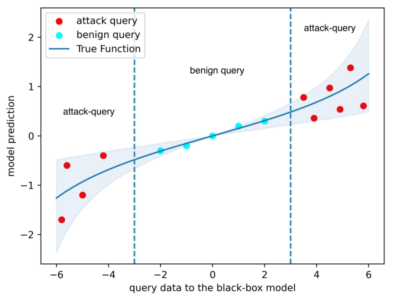
$$\min_{\boldsymbol{\phi}_V} [\mathcal{L}_0 = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_{id}} \mathcal{L}(\boldsymbol{x}, y, \boldsymbol{\phi}_V)] \tag{3}$$

where $\mathcal{L}(\boldsymbol{x}, y, \boldsymbol{\phi}_V)$ is the cross-entropy loss for classification, where $\mathcal{D}_{id}$ is the ID dataset used for training the victim model. However, simply optimizing Eq. (3) on ID dataset ignores the potential DFME attack performed by attackers, making the victim model easily stolen. To address this issue, we propose a new learning objective to make the victim model hard to be stolen.

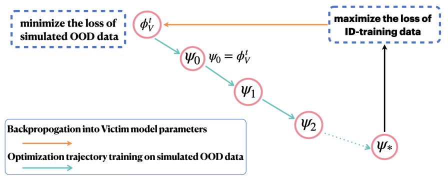
4.2 AAUG Defense Training

Below, we propose an Attack-Aware and Uncertainty-Guided defense method, named AAUG, to train the victim model. Precisely, AAUG method consists of: (1) OOD data simulation; (2) uncertainty-guided model training; and (3) attack-aware defensive training. We detail each component in the following.

OOD Data Simulation. We use a data generator $h_{\boldsymbol{\omega}}$ parameterized by $\boldsymbol{\omega}$ with random noise η as inputs, i.e., $\widehat{\boldsymbol{x}} = h_{\boldsymbol{\omega}}(\eta)$; $\eta \sim \mathcal{N}(0, I)$ to simulate the OOD data distribution. Due to space limitations, we provide more details about OOD data generator training details in Appendix.



(a) Illustration of **uncertainty-guided training** to defend against DFME. We aim to maximize the weight uncertainty for attack queries (red points) to defend against DFME, and minimize the weight uncertainty for benign queries (light blue points) to maintain model utility. After that, the function is hard to fit (extract) for attack queries since the prediction spreads wider. The blue line indicates the underlying ground-truth function. The shaded area indicates the function space estimation uncertainty.



minimize the loss of simulated OOD data
maximize the loss of ID-training data
ϕ_V^t
ψ_0
$\psi_0 = \phi_V^t$
ψ_1
ψ_2
ψ_*
Backpropogation into Victim model parameters
Optimization trajectory training on simulated OOD data

(b) Illustration of **attack-aware defensive training** at iteration t. Initialize the lower-level optimization with $\boldsymbol{\psi}_0 = \boldsymbol{\phi}_V^t$ (including the deterministic model parameters and random perturbation parameters). To simulate the model extraction attack process, we train the model on the simulated OOD data for several stochastic gradient descent steps, obtaining the simulated optimal attack model parameters $\boldsymbol{\psi}_*$. Then, for the upper-level optimization, we maximize the loss function on the ID-training data with respect to the victim model, $\boldsymbol{\phi}_V^t$, to make it hard to be stolen.

Fig. 1: Illustration of our defensive training components

Uncertainty-Guided Training. Uniformly adding the same scale of rand weight perturbation to the weights for every input does not work well. On the one hand, too large weight perturbation would significantly reduce the model utility; on the other hand, too small random weight perturbation will make the method ineffective in defending against DFME. Intuitively, the ID data should have low weight uncertainty to maintain the victim model utility; the OOD data (attack query) is expected to have high weight uncertainty or function space uncertainty to increase the difficulty of function fitting. To clarify, we illustrate our idea in Figure 1a.

Specifically, we propose a data-specific weight uncertainty to achieve this goal. For the i^{th} layer of the victim model with weights $\boldsymbol{W}_i$ and bias $\boldsymbol{b}_i$. We add layer-specific weight perturbation as follows:

$$\widehat{\boldsymbol{W}_i} = \boldsymbol{W}_i + g_{\boldsymbol{\theta}_i}(\boldsymbol{x}) \cdot \boldsymbol{\epsilon}, \;\; \boldsymbol{\epsilon} \sim \mathcal{N}(0, I) \tag{4}$$

where $\widehat{\boldsymbol{W}_i}$ is the perturbed model weights. $g_{\boldsymbol{\theta}_i}$ denotes the network with learnable weights $\boldsymbol{\theta}_i$ for generating data-specific weight perturbation magnitude. $g_{\boldsymbol{\theta}_i}$ is a small-scale CNN network in our experiment with negligible number of parameters compared to the victim model. We only learn per-layer scale of the weight perturbation to the victim model to minimize the increased number of learnable parameters and make the network sufficiently difficult to be stolen. By adopting data-specific weight uncertainty, the victim model can generate different model weights for different inputs. The victim model can make prediction as $\mathcal{V}_i(\boldsymbol{x}, \boldsymbol{\phi}_V(\boldsymbol{x}))$, where $\boldsymbol{\phi}_V(\boldsymbol{x})$ is the victim model weights specially generated for input $\boldsymbol{x}$. This significantly enlarges the function space (uncountably infinite functions) that the attacker should fit compared to the standard victim model, which only needs to fit a single function, protecting the victim model from being stolen. We define the following loss function to achieve this goal:

$$\min_{\{\boldsymbol{\theta}_i\}} \left[\mathcal{L}_1 = \sum_{i=1}^{i=N} (\mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_{id}} \| g_{\boldsymbol{\theta}_i}(\boldsymbol{x}) \| - \mathbb{E}_{(\widehat{\boldsymbol{x}},\widehat{y})\sim\mathcal{D}_{ood}} \| g_{\boldsymbol{\theta}_i}(\widehat{\boldsymbol{x}}) \|) \right] \tag{5}$$

as the weight uncertainty gap between the ID and simulated OOD training data. $\widehat{y}$ is the victim model prediction on the simulated OOD data $\widehat{\boldsymbol{x}}$, i.e., $\widehat{y} = \mathcal{V}(\widehat{\boldsymbol{x}}, \boldsymbol{\phi}_V)$. N denotes the number of layers; $\mathcal{D}_{id}$ denotes the ID training data; $\mathcal{D}_{ood}$ denotes the simulated OOD training data. Minimizing this loss function indicates that the weight perturbation on the OOD dataset should be as large as possible to increase the difficulty of model extraction for attacker. At the same time, it constrains the weight perturbation on the ID dataset should be as small as possible to maintain the victim model utility.

Attack-Aware Defensive Training. To further increase the challenge of function fitting for attacker, we propose a bi-level attack-aware defensive training objective to make the victim model secure:

$$\max_{\boldsymbol{\phi}_V} [\mathcal{L}_2 = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_{id}} \mathcal{L}(\boldsymbol{x}, y, \mathbb{A}(\boldsymbol{\phi}_V))] \tag{6}$$

$$\text{s.t. } \mathbb{A}(\boldsymbol{\phi}_V) = \arg\min_{\boldsymbol{\psi}} \mathbb{E}_{(\widehat{\boldsymbol{x}},\widehat{y})\sim\mathcal{D}_{ood}} \mathcal{L}(\widehat{\boldsymbol{x}}, \widehat{y}, \boldsymbol{\phi}_V, \boldsymbol{\psi}) \tag{7}$$

where $\mathcal{L}(\widehat{\boldsymbol{x}}, \widehat{y}, \boldsymbol{\phi}_V, \boldsymbol{\psi}) = \mathcal{L}(\widehat{\boldsymbol{x}}, \widehat{y}, \boldsymbol{\psi})$ and $\boldsymbol{\psi}$ is initialized as $\boldsymbol{\phi}_V$ for optimization. The lower-level optimization (Eq. (7)) is to simulate the attacker behavior to optimize the performance on the simulated OOD datasets; $\mathbb{A}(\boldsymbol{\phi}_V)$ denotes the optimal learned parameters initialized from $\boldsymbol{\phi}_V$ by training on the simulated

OOD data. The upper-level optimization (Eq. 6) is to *worsen* the performance on the ID dataset to make the optimized attack model generalize poorly on the ID-test data. We illustrate the attack-aware defensive training in Figure 1b. This bi-level optimization is similar to MAML [8] which is a min-min optimization, while ours is a max-min optimization. To save memory and reduce computation cost, we adopt the method similar to FOMAML [8] to optimize the objective.
Overall Learning Objective. We summarize the base model training, uncertainty-guided learning objective, and attack-aware defensive training into a single loss function. Denote the collection of learnable parameters $\boldsymbol{W}_i$, $\boldsymbol{b}_i$, $\boldsymbol{\theta}_i$ as $\boldsymbol{\phi}_V$, i.e., $\boldsymbol{\phi}_V = \{\boldsymbol{W}_i, \boldsymbol{b}_i, \boldsymbol{\theta}_i\}_{i=1}^{i=N}$. The learning objective for the victim model is as follows:

$$\min_{\boldsymbol{\phi}_V} [\mathcal{L}(\boldsymbol{\phi}_V) = \mathcal{L}_0 + \tau\mathcal{L}_1 - \gamma\mathcal{L}_2] \tag{8}$$

where $\mathcal{L}_0$, $\mathcal{L}_1$ and $\mathcal{L}_2$ are defined as above in Eq. (3), (5) and (6) respectively, γ and τ are two regularization constant. We summarize the details in Algorithm 1. In line 3-4, we randomly sample ID and simulated OOD dataset. In line 5-8, we calculate the base model training loss, uncertainty-guided loss and attack-aware defensive training loss function, respectively. Then, we update the victim model via SGD optimizer with respect to the model parameters.

Algorithm 1 AAUG **Training**.

```
1: REQUIRE: Victim model $V$ with parameters $\boldsymbol{\phi}_V$; weight perturbation network
   $g_{\boldsymbol{\theta}}$; victim model learning rate $\eta$; $M$ is the number of training iterations; the in-
   distribution (ID) training dataset, $\mathcal{D}_{id}$; pre-trained OOD dataset generator.
2: for $k = 1$ to $M$ do
3:     Sample a new mini-batch data $(\boldsymbol{x}, y)$ from $\mathcal{D}_{id}$
4:     Sample a mini-batch of OOD data from the pre-trained OOD generator.
5:     calculate the base model training loss function $\mathcal{L}_0$ (Eq. (3)).
6:     calculate the uncertainty-guided loss function $\mathcal{L}_1$ (Eq. (5))
7:     calculate the defensive training objective loss function $\mathcal{L}_2$ (Eq. (6) and Eq. (7))
8:     calculate the total loss function (Eq. (8)) and update the victim model param-
   eters by gradient decent with the loss function $\mathcal{L}(\boldsymbol{\phi}_V)$
9: end for
```

AAUG Deployment. During testing, we randomly sample a set of weights from each layer as Eq. (4) to obtain the victim model weights $\boldsymbol{\phi}_V$. Then, we use $\boldsymbol{\phi}_V$ to make prediction as $\mathcal{V}(\boldsymbol{x}, \boldsymbol{\phi}_V(\boldsymbol{x}))$.

4.3 Why AAUG defend against DFME

Weight smoothing defends against model extraction Our method can be viewed as random smoothing operated on neural network weights as:

$$\int \mathcal{V}(\boldsymbol{x}, \boldsymbol{\phi}_V + g_{\boldsymbol{\theta}}(\boldsymbol{x}) \cdot \boldsymbol{\epsilon}) d\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, I) \tag{9}$$

Eq. (9) indicates that AAUG makes it considerably more challenging for attackers to steal the victim model. This increased difficulty stems from the fact that computing Eq. (9) necessitates a substantially larger query budget to perform the integral calculation. As a result, our method can be widely applied to defend against a variety of model extraction attacks as it *does not* rely on the assumption that the attack query data are OOD data.

Defend against Soft-Label DFME. We summarize the reason for the AAUG's effectiveness in three fold: **(i)** First, clone model needs to learn an uncountably infinite number of functions which is much harder than the standard victim model stealing, which only needs to learn a single function. **(ii)** Second, AAUG increases the error of the zeroth-order gradient by the following equation:

$$\nabla_{\boldsymbol{x}}\mathcal{L}_G = \sum_{i=1}^{i=m} \frac{L_G(\boldsymbol{x}+\mu_i\Delta, \boldsymbol{\phi}_V^1) - L_G(\boldsymbol{x}, \boldsymbol{\phi}_V^2)}{\Delta}\mu_i \tag{10}$$

where $\boldsymbol{\phi}_V^1$ and $\boldsymbol{\phi}_V^2$ are two different sets of victim model parameters sampled from the learned weight perturbation distribution. Compared to Eq. (2), Eq. (10) would increase the error of zeroth-order gradient estimation since Eq. (10) uses different set of model parameters. **(iii)** Third, our method proposes an attack-aware defensive training objective to make the model itself hard to steal.

Defend against Hard-Label DFME. Due to space limitations, we put how AAUG defends against hard-label DFME in Appendix.

5 Experiments

5.1 Experimental Setup

Datasets. We use **MNIST**, **CIFAR10**, **CIFAR100** [26], **ImageNet-100** (100 classes) [53] and **ImageNet-1K** [4] to evaluate the defense effectiveness.

DFME Attack Baselines. (1) soft-label DFME: MAZE [22] and DFME [52]; and (2) hard-label DFME: DFMS-HL [47]. We do not compare to ZSDB3KD [59] since it requires a large amount of queries. Further, DFMS-HL significantly outperforms ZSDB3KD in terms of clone model accuracy.

ME Defense Baselines. We compare the AAUG method with several existing ME defense baselines to defend against DFME: (1) *Undefended*: the victim model without using any defense strategy; (2) Random Perturb (*RandP*) [41]: randomly perturb the output probabilities. (3) *P-poison* [41]: perform optimization to perturb the output probabilities; (4) GRAD [36]: gradient redirection defense. We set a large l_1 perturbation budget equal to 1.0 for those defense baselines to generate strong defense against DFME, i.e., $||\boldsymbol{y} - \hat{\boldsymbol{y}}||_1 \leq 1.0$; where $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$ are the unmodified and modified output probabilities, respectively. More details of the baselines are placed in Appendix.

Implementation Details. Following [52], for soft-label DFME methods, we set the number of queries to be $2M$ for MNIST, $20M$ for CIFAR10, $200M$ for CIFAR100 and $200M$ for ImageNet-100, respectively. For hard-label setting, following [47], we set the attack query budget to be $8M$ for CIFAR10, $10M$ for CIFAR100 and $10M$ for ImageNet-100. We perform each experiment for five runs with a mean and standard deviation of results. Due to space limitations, we put more details of hard-label DFME training, OOD data generator and victim model training details in Appendix.

5.2 Defense Against DFME

Clone Model Accuracy. To evaluate the effectiveness of AAUG against various DFME attack methods, we perform extensive experiments on various datasets with both soft-label and hard-label DFME attack settings. We present the results on CIFAR10 in Table 1, CIFAR100 in Table 3, and ImageNet-100 datasets in Table 4. We provide results on MNIST in Appendix. For CIFAR10, CIFAR100 and ImageNet-100 datasets, we use ResNet34-8x [10] as the victim model architecture and adopt ResNet18-8x [10], MobileNetV2 [46], and DenseNet121 [16] as different clone model architecture. For the MNIST dataset, we use LeNet5 as the victim model and LeNet5, LeNet5-Half, and LeNet5-1/5 as different clone model architectures. LeNet5-Half and LeNet5-1/5 have a half and fifth number of convolutional filters in each layer of that LeNet5, respectively.

We observe that AAUG outperforms existing strong defense methods by a large margin. In particular, AAUG reduces the clone model accuracy by up to 40%, surpassing the best baseline by up to 32% on CIFAR10. AAUG subsantialy reduces the clone model accuracy by up to 37%, outperforming best methods by up to 18% on CIFAR100. AAUG significantly reduces the clone model accuracy by more than 20% on ImageNet-100, exceeding the best baseline performance by more than 14%. For hard-label setting, our method still outperforms Rand, P-poison, and GRAD since these baseline methods maintain the hard-label of most query input (*top-1 label not changed*), thus these defense methods do not work well. The effectiveness of our defense method is due to our attack-aware and uncertainty-guided training strategy. (1) RandP preserves utility for nearly all query data, which is unnecessary for attack queries as the attacker can still extract useful information. (2) P-poison needs a randomly initialized surrogate attacker model, which acts as an adversarial model. (3) GRAD requires knowledge of the

Table 1: Clone model accuracy on **CIFAR-10** with ResNet34-8x as the victim model

Attack	**Defense**	Clone Model Architecture		
		ResNet18-8X	MobileNetV2	DenseNet121
DFMS-HL	undefended ↓	84.67 ± 1.90%	79.28 ± 1.87%	68.87 ± 2.08%
	RandP ↓	84.02 ± 2.31%	78.71 ± 1.93%	68.16 ± 2.23%
	P-poison ↓	84.06 ± 1.87%	79.02 ± 1.96%	68.05 ± 2.17%
	GRAD ↓	84.28 ± 1.95%	78.83 ± 1.91%	68.11 ± 1.93%
	AAUG ↓	**74.38 ± 1.72%**	**68.36 ± 1.65%**	**62.36 ± 1.81%**
DFME	undefended ↓	87.36 ± 0.78%	75.23 ± 1.53%	73.89 ± 1.29%
	RandP ↓	84.28 ± 1.37%	70.56 ± 2.23%	70.03 ± 2.38%
	P-poison ↓	78.06 ± 1.73%	66.32 ± 1.36%	68.75 ± 1.40%
	GRAD ↓	79.33 ± 1.68%	65.82 ± 1.67%	69.06 ± 1.57%
	AAUG ↓	**46.82 ± 1.79%**	**45.17 ± 1.93%**	**40.91 ± 1.89%**
MAZE	undefended ↓	45.17 ± 0.73%	23.28 ± 1.67%	20.03 ± 1.79%
	RandP ↓	28.76 ± 2.38%	22.03 ± 1.50%	18.79 ± 1.38%
	P-poison ↓	26.81 ± 2.19%	20.89 ± 1.58%	**17.08 ± 2.28%**
	GRAD ↓	26.06 ± 1.81%	21.18 ± 1.58%	18.09 ± 1.72%
	AAUG ↓	**21.67 ± 2.02%**	**19.20 ± 1.96%**	17.32 ± 1.83%

Table 2: Victim model utility measured by the test accuracy

Method	**MNIST**	**CIFAR10**	**CIFAR100**	**ImageNet-100**
undefended ↑	98.91 ± 0.16%	94.91 ± 0.37%	76.71 ± 1.25%	60.57 ± 1.67%
RandP ↑	98.52 ± 0.19%	93.98 ± 0.28%	75.23 ± 1.39%	58.96 ± 1.83%
P-poison ↑	98.87 ± 0.35%	94.58 ± 0.61%	75.42 ± 1.21%	58.79 ± 1.79%
GRAD ↑	98.73 ± 0.31%	**94.65 ± 0.67%**	75.60 ± 1.45%	58.85 ± 1.80%
AAUG ↑	**98.89 ± 0.42%**	94.21 ± 0.56%	**75.73 ± 0.93%**	**59.06 ± 1.86%**

attack query set to train the surrogate model, leading to discrepancies compared to the DFME attacker model. These methods perform worse because defender lacks information about attack query data and model. In contrast, AAUG does not need a surrogate model, enhancing its effectiveness.

Model Utility Evaluation. To evaluate the victim model utility with the AAUG defense, we evaluate the test accuracy with various defense strategies in Table 2. AAUG achieves slightly better benign test accuracy. In addition, we also evaluate the l_1 norm of $||\boldsymbol{y} - \hat{\boldsymbol{y}}||_1$ (the lower, the better); where $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$ are the original and perturbed output probabilities, respectively. We present the results in Appendix. The results show that our method achieves better defense with smaller output perturbation. This is beneficial for benign inputs since they may need helpful logits to study the helpful knowledge in the model outputs.

Effect of Different Victim Model Network Architecture. To evaluate the effect of different victim model architectures, we use WideResNet-28-2 [62] as the victim model architecture on the CIFAR10 dataset. We present the results in Appendix. AAUG still substantially outperforms various strong baselines under different DFME attack strategies.

Table 3: Clone model accuracy on **CIFAR-100** with ResNet34-8x as the victim model.

Attack	**Defense**	Clone Model Architecture		
		ResNet18-8X	MobileNetV2	DenseNet121
DFMS-HL	undefended ↓	$72.57 \pm 1.28\%$	$62.71 \pm 1.68\%$	$63.58 \pm 1.79\%$
	RandP ↓	$72.43 \pm 1.43\%$	$62.06 \pm 1.82\%$	$63.16 \pm 1.73\%$
	P-poison ↓	$71.83 \pm 1.32\%$	$61.83 \pm 1.79\%$	$62.73 \pm 1.91\%$
	GRAD ↓	$71.89 \pm 1.37\%$	$62.60 \pm 1.71\%$	$62.57 \pm 1.80\%$
	AAUG ↓	$\mathbf{45.68 \pm 1.42\%}$	$\mathbf{46.98 \pm 1.83\%}$	$\mathbf{49.56 \pm 1.91\%}$
DFME	undefended ↓	$58.72 \pm 2.82\%$	$28.36 \pm 1.97\%$	$27.28 \pm 2.08\%$
	RandP ↓	$41.69 \pm 2.91\%$	$22.75 \pm 2.19\%$	$23.61 \pm 2.70\%$
	P-poison ↓	$38.72 \pm 3.06\%$	$20.87 \pm 2.61\%$	$21.89 \pm 2.93\%$
	GRAD ↓	$39.07 \pm 2.72\%$	$20.71 \pm 2.80\%$	$22.08 \pm 2.78\%$
	AAUG ↓	$\mathbf{21.31 \pm 2.03\%}$	$\mathbf{10.38 \pm 2.16\%}$	$\mathbf{9.87 \pm 2.19\%}$

Table 4: Clone model accuracy on **ImageNet-100** with ResNet34-8x as the victim model.

Attack	**Defense**	Clone Model Architecture		
		ResNet18-8X	MobileNetV2	DenseNet121
DFMS-HL	undefended ↓	$46.72 \pm 4.86\%$	$40.35 \pm 4.97\%$	$38.71 \pm 3.85\%$
	RandP ↓	$45.09 \pm 4.93\%$	$39.51 \pm 4.83\%$	$38.08 \pm 3.95\%$
	P-poison ↓	$45.16 \pm 5.03\%$	$39.06 \pm 4.72\%$	$37.78 \pm 4.26\%$
	GRAD ↓	$45.32 \pm 5.21\%$	$39.17 \pm 4.85\%$	$37.85 \pm 4.32\%$
	AAUG (Ours) ↓	$\mathbf{39.23 \pm 4.83\%}$	$\mathbf{35.81 \pm 4.69\%}$	$\mathbf{32.30 \pm 4.56\%}$
DFME	undefended ↓	$35.89 \pm 3.97\%$	$28.71 \pm 3.25\%$	$25.05 \pm 3.68\%$
	RandP ↓	$30.76 \pm 4.09\%$	$22.06 \pm 3.83\%$	$20.23 \pm 3.97\%$
	P-poison ↓	$29.36 \pm 4.23\%$	$21.83 \pm 3.77\%$	$20.01 \pm 3.89\%$
	GRAD ↓	$29.87 \pm 3.76\%$	$21.65 \pm 3.75\%$	$19.82 \pm 3.77\%$
	AAUG (Ours) ↓	$\mathbf{15.58 \pm 3.81\%}$	$\mathbf{15.23 \pm 3.86\%}$	$\mathbf{14.30 \pm 3.73\%}$

5.3 Defense Against DBME

In this section, we extend our method to defend against data-based model extraction (DBME) to show the general applicability of our proposed method. The attacker can access a small subset of in-distribution training data or a surrogate dataset. Specifically, we aim to defend against KnockoffNets [39] and JBDA [44] attacks. Due to the space limitations, we put more results in Appendix. We can observe that AAUG outperforms compared methods by more than 5% in many cases. This is because AAUG can be perceived as a form of random smoothing, as in Eq. 9, thereby rendering the victim model notably more challenging to steal. This heightened difficulty arises from the requirement of a significantly larger query budget to compute Eq. (9) for the integral calculation.

5.4 Adversary Countermeasures

To evaluate the effect of adversary countermeasures on our defense strategies, we provide two potential countermeasures adopted by the attacker: (1) only use hard-label instead of the soft-label outputs; (2) also add a similar weight perturbation to the clone model. Specifically, we present the results of adaptive attack by learning an additional weight perturbation in Appendix. Our method still significantly reduces the effectiveness of various DFME attack methods since learning a large function space with uncountably infinite functions is much harder than learning a single function. The gap between the clone model and the victim model becomes even larger. We present the hard-label adaptive attack in Appendix. In this case, with only a hard label, the attacker can use much less information than the soft label. Further, the random weight perturbation significantly increases the likelihood of changing the label prediction. The attacker learns incorrect information and cannot extract the victim model.

5.5 Defense against Extracting Self-Supervised Learning Model

To evaluate the effectiveness of our method against stealing self-supervised learning model, we conduct model extraction and defense on the model pre-trained with SimSiam [3] on ImageNet-1K [4]. Following the protocol presented in [5], we evaluate the quality of the extracted encoder through different downstream task performance on the dataset, CIFAR10, CIFAR100 and SVHN. The results are presented in Appendix. We can observe that under various defense strategies, our method significantly outperforms the SOTA defense methods by more than 6%. Due to the space limitations, we put more details in Appendix.

5.6 Ablation Study

Effect of Different Query Budgets by Attacker. To evaluate how the clone model accuracy changes with different query budgets, we evaluate the clone model performance with different defense strategies under different query budgets. We show the results in Appendix. We observe that the baseline methods become less effective with more attack query data. On the contrary, our method is still effective even with more query budgets. The attacker learns misleading information due to the high weight uncertainty on the attack query data and attack-aware defensive training.
Hyperparameter Sensitivity. To evaluate the hyperparameter sensitivity of γ and τ to the test accuracy of the victim model and clone model. We perform experiments with different values of γ and τ in Appendix, respectively.
Test Time Speed Comparisons. To evaluate the test running time efficiency, we evaluate the running time of different methods in Appendix. We can observe that AAUG is substantially more efficient than baseline methods. Notably, AAUG is far more efficient than these compared defense methods and achieves $25 \sim 103\times$ speed up compared to baseline methods on CIFAR10 and CIFAR100.

P-poison and GRAD are much slower since they perform an expensive optimization during testing, which involves many times of time-consuming forward and backward propagation.
Training Efficiency. To evaluate the training efficiency, we compare the training efficiency of AAUG (Eq. (8)) to standard base model training (Eq. (3)) in Appendix. AAUG increases the base model training cost by 1.2 $\times$. The main computation cost comes from solving Eq. (6) and (7). However, AAUG is substantially more efficient during testing. This additional cost is worth it.

Improving Scalability by Fine-Tuning the Victim Model Retraining the victim model with AAUG is computationally expensive. To enhance training efficiency, we opted to fine-tune the pre-trained model instead of retraining from scratch. We peform experiments by fine-tuning the ResNet for 20 epochs, compared to the 200 epochs required for full retraining. The defense results on CIFAR10 are presented in Appendix, and the fine-tuning cost is detailed in Appendix. The defense performance achieved with fine-tuning is similar and comparable to retraining. Additionally, by fine-tuning the victim model, we reduced the training cost by 83%.

6 Conclusion

This paper addresses the problem of defending against DFME. We introduce a defense strategy, termed Attack-Aware and Uncertainty-Guided defense (AAUG), aimed at training a secure model, thereby enhancing the resilience of the victim model against theft attempts. AAUG consists of uncertainty-guided training and attack-aware defensive training. Extensive experiments with various datasets on defending against both soft-label and hard-label DFME methods demonstrate the effectiveness and efficiency of AAUG. Furthermore, we extend our method to defend against traditional data-based model extraction (DBME) and extracting self-supervised learning pre-trained models.

Limitations: This work needs to further reduce the clone model accuracy and maximally protect the victim model security. We leave this research direction as future work. Potential directions would include exploring more efficient and effective network architecture to achieve more effective defense.

Acknowledgments This work was partially supported by NSF-2239537.

References

1. Adi, Y., Baum, C., Cisse, M., Pinkas, B., Keshet, J.: Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In: 27th USENIX Security Symposium (USENIX Security 18) (2018)
2. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International Conference on Machine Learning (2020)
3. Chen, X., He, K.: Exploring simple siamese representation learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 15750–15758 (2021)
4. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
5. Dziedzic, A., Dhawan, N., Kaleem, M.A., Guan, J., Papernot, N.: On the difficulty of defending self-supervised learning against model extraction. In: International Conference on Machine Learning. pp. 5757–5776. PMLR (2022)
6. Dziedzic, A., Kaleem, M.A., Lu, Y.S., Papernot, N.: Increasing the cost of model extraction with calibrated proof of work. In: International Conference on Learning Representations (2022)
7. Fang, G., Song, J., Wang, X., Shen, C., Wang, X., Song, M.: Contrastive model inversion for data-free knowledge distillation. In: International Joint Conferences on Artificial Intelligence (2021)
8. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: International Conference on Machine Learning (2017)
9. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial networks. In: Advances in Neural Information Processing Systems (2014)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (2016)
11. He, Y., Meng, G., Chen, K., Hu, X., He, J.: Towards security threats of deep learning systems: A survey (2020), https://arxiv.org/abs/1911.12562
12. He, Z., Rakin, A.S., Fan, D.: Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 588–597 (2019)
13. Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., et al.: The many faces of robustness: A critical analysis of out-of-distribution generalization. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 8340–8349 (2021)
14. Hong, Z., Wang, Z., Shen, L., Yao, Y., Huang, Z., Chen, S., Yang, C., Gong, M., Liu, T.: Improving non-transferable representation learning by harnessing content and style. In: The Twelfth International Conference on Learning Representations (2024)
15. Hu, Z., Shen, L., Wang, Z., Wu, B., Yuan, C., Tao, D.: Learning to learn from apis: black-box data-free meta-learning. In: International Conference on Machine Learning. pp. 13610–13627 (2023)
16. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: IEEE Conference on Computer Vision and Pattern Recognition (2017)

17. Jagielski, M., Carlini, N., Berthelot, D., Kurakin, A., Papernot, N.: High accuracy and high fidelity extraction of neural networks. In: USENIX Security 2020 (2020)
18. Jia, H., Choquette-Choo, C.A., Chandrasekaran, V., Papernot, N.: Entangled watermarks as a defense against model extraction. In: 30th USENIX Security Symposium (2021)
19. Jia, H., Yaghini, M., Choquette-Choo, C.A., Dullerud, N., Thudi, A., Chandrasekaran, V., Papernot, N.: Proof-of-learning: Definitions and practice. In: 42nd IEEE Symposium on Security and Privacy (2021)
20. Juuti, M., Szyller, S., Marchal, S., Asokan, N.: Prada: Protecting against dnn model stealing attacks. In: IEEE European Symposium on Security and Privacy (2019), https://arxiv.org/abs/1805.02628
21. Juuti, M., Szyller, S., Marchal, S., Asokan, N.: Prada: Protecting against dnn model stealing attacks. In: IEEE European Symposium on Security and Privacy (2019)
22. Kariyappa, S., Prakash, A., Qureshi, M.K.: Maze: Data-free model stealing attack using zeroth-order gradient estimation. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (2021)
23. Kariyappa, S., Prakash, A., Qureshi, M.K.: Protecting {dnn}s from theft using an ensemble of diverse models. In: International Conference on Learning Representations (2021)
24. Kariyappa, S., Qureshi, M.K.: Defending against model stealing attacks with adaptive misinformation. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (2020)
25. Kong, S., Ramanan, D.: Opengan: Open-set recognition via open data generation. In: IEEE/CVF International Conference on Computer Vision (2021)
26. Krizhevsky, A.: Learning multiple layers of features from tiny images (2009)
27. Lee, T., Edwards, B., Molloy, I., Su, D.: Defending against machine learning model stealing attacks using deceptive perturbations (2018), https://arxiv.org/abs/1806.00054
28. Li, C.L., Chang, W.C., Cheng, Y., Yang, Y., Póczos, B.: Mmd gan: Towards deeper understanding of moment matching network. In: Advances in Neural Information Processing Systems (2017)
29. Li, G., Xu, G., Guo, S., Qiu, H., Li, J., Zhang, T.: Extracting robust models with uncertain examples. In: International Conference on Learning Representations (2023)
30. Liu, S., Chen, P.Y., Kailkhura, B., Zhang, G., Hero III, A.O., Varshney, P.K.: A primer on zeroth-order optimization in signal processing and machine learning: Principals, recent advances, and applications. IEEE Signal Processing Magazine (2020)
31. Liu, X., Cheng, M., Zhang, H., Hsieh, C.J.: Towards robust neural networks via random self-ensemble. In: European Conference on Computer Vision (2018)
32. López, G., Quesada, L., Guerrero, L.A.: Alexa vs. siri vs. cortana vs. google assistant: a comparison of speech-based natural user interfaces. In: Advances in Human Factors and Systems Interaction: Proceedings of the AHFE 2017 International Conference on Human Factors and Systems Interaction, July 17- 21, 2017, The Westin Bonaventure Hotel, Los Angeles, California, USA 8. pp. 241–250. Springer (2018)
33. Lowd, D., Meek, C.: Adversarial learning. In: ACM SIGKDD international conference on Knowledge discovery in data mining. Association for Computing Machinery (2005)
34. Maini, P., Yaghini, M., Papernot, N.: Dataset inference: Ownership resolution in machine learning. In: International Conference on Learning Representations (2021)

35. Marek, P., Naik, V.I., Auvray, V., Goyal, A.: Oodgan: Generative adversarial network for out-of-domain data generation. In: Annual Conference of the North American Chapter of the Association for Computational Linguistics (2021), `https://arxiv.org/abs/2104.02484`
36. Mazeika, M., Li, B., Forsyth, D.: How to steer your adversary: Targeted and efficient model stealing defenses with gradient redirection. In: International Conference on Machine Learning (2022)
37. Oh, S.J., Augustin, M., Schiele, B., Fritz, M.: Towards reverse-engineering black-box neural networks. In: International Conference on Learning Representations (2018)
38. Oord, A.v.d., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 (2018)
39. Orekondy, T., Schiele, B., Fritz, M.: Knockoff nets: Stealing functionality of black-box models. In: IEEE Conference on Computer Vision and Pattern Recognition (2019)
40. Orekondy, T., Schiele, B., Fritz, M.: Knockoff nets: Stealing functionality of black-box models. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
41. Orekondy, T., Schiele, B., Fritz, M.: Prediction poisoning: Towards defenses against dnn model stealing attacks. In: International Conference on Learning Representations (2020)
42. Pal, S., Gupta, Y., Kanade, A., Shevade, S.: Stateful detection of model extraction attacks (2021)
43. Pal, S., Gupta, Y., Shukla, A., Kanade, A., Shevade, S., Ganapathy, V.: Activethief: Model extraction using active learning and unannotated public data. In: AAAI Conference on Artificial Intelligence. vol. 34 (2020)
44. Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z.B., Swami, A.: Practical black-box attacks against machine learning. In: ACM Asia Conference on Computer and Communications Security (2017)
45. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. In: International Conference on Learning Representations (2016)
46. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (2018)
47. Sanyal, S., Addepalli, S., Babu, R.V.: Towards data-free model stealing in a hard label setting. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (2022)
48. Steiner, A.P., Kolesnikov, A., Zhai, X., Wightman, R., Uszkoreit, J., Beyer, L.: How to train your vit? data, augmentation, and regularization in vision transformers. Transactions on Machine Learning Research (2022)
49. Szyller, S., Atli, B.G., Marchal, S., Asokan, N.: Dawn: Dynamic adversarial watermarking of neural networks. In: ACM Multimedia (2021)
50. Tieleman, T., Hinton, G.: Lecture 6.5-: Divide the gradient by a running average of its recent magnitude (2012)
51. Tramèr, F., Zhang, F., Juels, A., Reiter, M.K., Ristenpart, T.: Stealing machine learning models via prediction apis. In: USENIX Security (2016)
52. Truong, J.B., Maini, P., Walls, R.J., Papernot, N.: Data-free model extraction. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (2021)
53. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. Advances in neural information processing systems (2016)

54. Vitter, J.S.: Random sampling with a reservoir. ACM Transactions on Mathematical Software (1985)
55. Wang, B., Gong, N.Z.: Stealing hyperparameters in machine learning. In: In the 39th IEEE Symposium on Security and Privacy (2018)
56. Wang, X., Wang, S., Chen, P.Y., Wang, Y., Kulis, B., Lin, X., Chin, S.: Protecting neural networks with hierarchical random switching: Towards better robustness-accuracy trade-off for stochastic defenses. In: International Joint Conference on Artificial Intelligence (2019)
57. Wang, Z., Shen, L., Liu, T., Duan, T., Zhu, Y., Zhan, D., Doermann, D., Gao, M.: Defending against data-free model extraction by distributionally robust defensive training. In: Thirty-seventh Conference on Neural Information Processing Systems (2023)
58. Wang, Z., Wu, Y., Huang, H.: Defense against model extraction attack by bayesian active watermarking. In: Forty-first International Conference on Machine Learning (2024)
59. Wang, Z.: Zero-shot knowledge distillation from a decision-based black-box model. In: International Conference on Machine Learning (2021)
60. Wu, B., Xu, C., Dai, X., Wan, A., Zhang, P., Yan, Z., Tomizuka, M., Gonzalez, J., Keutzer, K., Vajda, P.: Visual transformers: Token-based image representation and processing for computer vision. arXiv preprint arXiv:2006.03677 (2020)
61. Yang, E., Wang, Z., Shen, L., Yin, N., Liu, T., Guo, G., Wang, X., Tao, D.: Continual learning from a stream of apis. arXiv preprint arXiv:2309.00023 (2023)
62. Zagoruyko, S., Komodakis, N.: Wide residual networks. In: British Machine Vision Conference (2016)