Efficient Model Stealing Defense with Noise Transition Matrix

Dong-Dong Wu[1], Chilin Fu[2], Weichang Wu[2], Wenwen Xia[2]
Xiaolu Zhang[2], Jun Zhou[2*], Min-Ling Zhang[1*]
[1]School of Computer Science and Engineering, Southeast University, Nanjing, China
[2]Ant Group
{dongdongwu,zhangml}@seu.edu.cn,{chilin.fcl, jiuyue.wwc, wenzhen.xww}@antgroup.com
{yueyin.zxl, jun.zhoujun}@antfin.com

Abstract

With the escalating complexity and investment cost of training deep neural networks, safeguarding them from unauthorized usage and intellectual property theft has become imperative. Especially the rampant misuse of prediction APIs to replicate models without access to the original data or architecture poses grave security threats. Diverse defense strategies have emerged to address these vulnerabilities, yet these defenses either incur heavy inference overheads or assume idealized attack scenarios. To address these challenges, we revisit the utilization of noise transition matrix as an efficient perturbation technique, which injects noise into predicted posteriors in a linear manner and integrates seamlessly into existing systems with minimal overhead, for model stealing defense. Provably, with such perturbed posteriors, the attacker's cloning process degrades into learning from noisy data. Toward optimizing the noise transition matrix, we proposed a novel bi-level optimization training framework, which performs fidelity on the victim model while the surrogate model adversarially. Comprehensive experimental results demonstrate that our method effectively thwarts model stealing attacks and achieves minimal utility trade-offs, outperforming existing state-of-the-art defenses.

1. Introduction

Machine learning (ML) models, especially deep neural networks (DNNs), have attained human-level capabilities across various domains, such as pattern recognition tasks [24, 27] and generation tasks [2, 52], becoming indispensable in facilitating convenience for society [53]. With the burgeoning of ML application scenarios and the proliferation of data, training large and sophisticated models and making them publicly accessible has become a new profit trend for companies. However, considering the steep costs of tuning models, directly divulging model parameters potentially engenders no returns on investment and even adverse societal impacts from malicious use. To alleviate those concerns, prediction APIs have emerged as an alternative by enabling black-box inferences through querying. This allows models to provide services without exposing the underlying parameters and details, and has been harnessed in Machine Learning as a Service (MLaaS) platforms (e.g., Amazon AWS, Microsoft Azure, Google Cloud).

Unfortunately, recent research has demonstrated that prediction APIs are still vulnerable to intellectual property theft through *model stealing attacks* [39, 51, 54, 62]. Despite not exposing model details, attackers can query the APIs using carefully-crafted samples or surrogate samples to obtain responded predictions. Through this, they can then train a clone model on the gathered query-response pairs to replicate the victim model's functionality with high accuracy and fidelity, without needing to access the original training data or model architecture [22, 39]. Such a clone model poses the following security threats: (a) acquiring a functionally similar copy of the victim model, illegally circumventing the expensive process of manual data curation and undermining the intellectual property of MLaaS; (b) extracting confidential data through membership inference attacks [47], model inversion attack [6] and property inference attack [7]; (c) conducting adversarial attacks [31, 33, 35, 36, 49, 55] by creating malicious examples based on the white-box substitute model [42].

To address such vulnerabilities, considerable researches [3, 5, 9, 17, 29, 50, 64] have focused on thwarting model stealing attacks. These defenses can be categorized into three strategies: detection, perturbation, and post-hoc verification. Detection-based defenses [19, 20, 23, 23] are predicated on the idea that malicious queries exhibit aberrant statistical distributions compared to normal queries. If a query sequence appears anomalous, the defender may discard the requests or corrupt the outputs. However, if an attacker carefully con-

* Corresponding author.
This work is done when Dong-Dong Wu works as an intern at Ant Group.

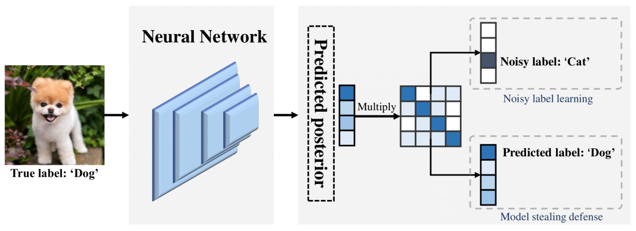


Figure 1. Illustrations of the noise transition matrix serving for noisy label learning and model stealing defense. In noisy label learning, it transforms the predicted posteriors to align with the noisy label. In model stealing defense, it perturbs the predicted posterior to impede attacker training while preserving the original predicted label for benign users.

structs queries matching the benign distribution, detection techniques will fail to discern the attack, yielding excessive false positives. Perturbation-based defenses [30, 40, 51] can preserve or constraint model accuracy to foil model stealing attacks. The accuracy-preserving defense truncates predicted posteriors to a minuscule fraction of their original size, which may reduce model lucidity and utility for critical applications like autonomous driving. The accuracy-constraint defense [38, 40] introduces perturbation into predicted posteriors, which incurs additional computational burdens or substantially degrading utility. Furthermore, the latency resulting from such perturbation methods leads to extended user response times, making the approach impractical. Post-hoc verification techniques such as watermarking [15] remain useful for post-attack prevention and validation. However, their retroactive nature means they cannot take proactive measures before an attack occurs. Considering inference latency and utility trade-offs, in this paper, we concentrate on a novel and efficient model stealing defense strategy.

To overcome the limitations of existing defenses, we explore an efficient method to inject noise into predicted posteriors via linear mapping, while preserving utility for benign users. Motivated by memorization effects [11, 63] that neural networks can easily overfit noisy data, thus hampering model generalization, we revisit the utilization of noise transition matrix (NTM) in noisy labels learning [43, 58], where the matrix models the noise process. Unlike related work, we aim to leverage the NTM to introduce noise into predicted posteriors for model defense, as shown in Figure 1. Our goal is to find an optimized NTM that injects noise while retaining original prediction labels with minimal utility trade-offs. Intuitively, regardless of whether the victim model produces extremely confident one-hot vectors or uncertain flat vectors, we can readily derive a uniform NTM where the diagonal elements remain maximal for each row to meet our defense needs. To estimate the NTM with maximum utility, we propose a bi-level optimization method that alternately optimizes it and updates the surrogate network weights. Specifically, we first make a one-step-forward virtual optimization of weights, then optimize the matrix guided by the loss with the one-step-forward weights fixed. Finally, we optimize the unrolled weights with the updated matrix. The contributions of this work are as follows:

- We propose a novel perturbation-based defense that efficiently introduces noise through linear mapping, adding minimal overhead. Furthermore, it deteriorates attacker training to a level equivalent to training on noisy-labeled data with instance-wise weight.
- Our method pioneers using the transition matrix to thwart model stealing attacks. The explicit matrix helps understand noise relationships among classes and makes the defense strategy more interpretable
- A bi-level optimization approach is proposed to optimize the transition matrix, which is model-agnostic and can adapt to different backbone networks.
- Extensive experiments exhibit our method consistently mitigates stealing and outperforms state-of-the-art defenses.

2. Related Work

Model Stealing Attacks aim to replicate the functionality of a victim model by interrogating its prediction APIs. The attacker sends queries and collects the resultant predictions, typically probability vectors or labels, as query-response pairs. These collected pairs are used to train a clone model that approximates the victim model. Tramèr et al. [51] first proposed the idea of model stealing and demonstrated the feasibility of extracting simple models utilizing queries. Juuti et al. [19] presented extraction attacks leveraging jacobian-based data augmentation [42], synthesizing data by perturbing a small set of in-distribution samples. However, these attacks presume the ability to optimally probe the APIs, which could be detected by monitoring for anomalous query patterns [19, 41]. Therefore, we concentrate on scenarios without carefully engineered input queries or sequences. One classic scenario to achieve this is by utilizing knowledge distillation techniques [12], where some studies exhibit model stealing is viable using weakly correlated [39] or random [25] queries, even when the teacher model declares itself undistillable [13, 28].

Model Stealing Defenses can be taxonomized into three primary strategies: detection, perturbation, and post-hoc verification. Detection strategies [20, 21] aim to identify malicious queries or nefarious users by scrutinizing the queries received by the victim model. For example, Kariyappa and Qureshi [20] regarded malicious queries as out-of-distribution samples and devised an adaptive misinformation (AM) defense to mislead attackers with modified outputs. However, such detection strategies rely on strong assumptions about adversarial query patterns that may not reflect reality. Perturbation strategies mitigate model stealing attacks by returning perturbed posterior, which can be further subcategorized into accuracy-preserving [30] and accuracy-constrained defenses [29, 38, 40]. Accuracy-preserving defenses guarantee that the maximal class of the perturbed predictions matchs the original prediction. The most straightforward such defense is truncating the posterior or only outputting the predicted label [51]. However, this diminishes utility for benign users, precluding specific applications and harming external transparency. In contrast, accuracy-constrained defenses permit a higher level of perturbation, trading off benign accuracy for enhanced security against model stealing attacks. In general, existing defense strategies that poison the attacker's update gradient [38, 40] are computationally expensive, even if they yield a comparative defense. Post-hoc verification, including watermarking [15, 32, 50], dataset inference [37], and proof of learning [16], embeds concealed patterns in the model during training [1] or inference [50] to distinguish a stolen model. Unfortunately, these approaches only work after a model theft, requiring the model owner to obtain partial or full access to the stolen model. This means if an attacker uses the stolen model as a proxy to attack the victim model, it is impossible for the model owner to protect themselves with these defenses. Overall, while existing ante-hoc defensive approaches mitigate knowledge leakage, they rely on strong assumptions or incur high computational costs during inference.

Noise Transition Matrix (NTM) is often utilized to solve classification with noisy labels or ambiguous labeled data [10, 46, 61], where the matrix reflects probabilities of true label flipping to the other label. A typical application is loss correction [43], which modifies the loss of each example by multiplying the estimated matrix by the model output. For NTM estimation, there are two main approaches. One is a two-step solution that pre-estimates the matrix with anchor point assumptions, then uses it to train the classifier [56, 65]. For instance, backward correction [43] initially approximates the NTM using the softmax output of the model trained without correction. It then retrains the model while correcting the original loss based on the estimated matrix. The other jointly estimates the matrix and classifier parameters in a unified framework without employing anchor points [18, 48, 60]. Differently, our work aims to estimate and apply the transition amtrix to intentionally inject noise into predicted posteriors, thus defending against model stealing.

3. Threat Model

We consider a model stealing scenario involving a defender and an attacker interacting on image classification tasks. The defender grants access to its black-box model through prediction API. Leveraging the API, the attacker collects query-response pairs, aiming to mimic the defender model (victim model) functionalities by training a clone model on this dataset. To evade detection defenses [19], we assume the attacker adopts non-adaptive querying strategies without anomalous patterns [39, 42]. Therefore, we consider the knockoff nets attack without adaptive querying [39] , which resembles knowledge distillation [12] where the victim model distills knowledge into the attacker model.

3.1. Attacker's Objective

Let $\boldsymbol{\theta}_f$ and $\boldsymbol{\theta}_h$ denote the parameters of the defender's model f and attacker's clone model h, respectively. The attacker prepares a query dataset $\mathcal{D}_{\text{query}}$, and sends queries $\boldsymbol{x} \in \mathcal{D}_{\text{query}}$ in random order to obtain the defender's posterior $f(\boldsymbol{x};\boldsymbol{\theta}_f)$ from the prediction API. The classification loss of the attacker's clone model on the query $\boldsymbol{x}$ is $\mathcal{L}_{\text{kna}} = -\sum_i f(\boldsymbol{x};\boldsymbol{\theta}_f)_i \log h(\boldsymbol{x};\boldsymbol{\theta}_h)_i$. The attacker aims to minimize the classification loss on the defender's test set. However, since the test set is unavailable, the attacker's learning objective becomes minimizing the loss based on the collected query-response dataset $\mathcal{D}_{\text{pair}} = \{(\boldsymbol{x}, f(\boldsymbol{x};\boldsymbol{\theta}_f)) | \boldsymbol{x} \in \mathcal{D}_{\text{query}}\}$. Therefore, the attacker's objective can be formulated as:

$$\min_{\boldsymbol{\theta}_h} \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_{\text{pair}}}[\mathcal{L}_{\text{kna}}(h(\boldsymbol{x};\boldsymbol{\theta}_h), f(\boldsymbol{x};\boldsymbol{\theta}_f))] \tag{1}$$

Query Distribution. In practice, due to the knowledge limitation of attackers about the defender's training set, the collected query dataset $\mathcal{D}_{\text{query}}$ may be similar [30, 51] or unrelated [20, 40] to the defender's training distribution. Hence, we consider both possibilities in our threat model. We refer to attackers as *distribution-aware* if their queries share semantic content with the defender's training distribution. Attackers are referred to as *knowledge-limited* if their queries are semantically disjoint from the defender's training distribution.

3.2. Defender's Objective

In defenses against model stealing, the defender aims at preventing its model functionality from being stolen. In real-world scenarios, attackers may disguise themselves as benign users, executing attacks anytime. Therefore, the defender must persistently mitigate model stealing attempts while preserving the API's utility for legitimate users. Specifically, the defender needs to keep its own small classification loss, while maximizing the classification loss of the attacker's cloned model $\boldsymbol{\theta}_h^*(\boldsymbol{\theta}_f)$ obtained from Eq. (1) on the test set $\mathcal{D}_{\text{test}}$. The defender's objective can be formulated as:

$$\begin{aligned} &\min_{\boldsymbol{\theta}_f} \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_{\text{test}}}[\mathcal{L}_{\text{CE}}(f(\boldsymbol{x};\boldsymbol{\theta}_f), y)] \\ &\text{s.t.} \ \max \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_{\text{test}}}[\mathcal{L}_{\text{CE}}(h(\boldsymbol{x};\boldsymbol{\theta}_h^*(\boldsymbol{\theta}_f)), y)], \end{aligned} \tag{2}$$

where $\mathcal{L}_{\text{CE}}$ means the cross-entropy loss function.

4. Approach

To develop an efficacious and inference-efficient defense against model stealing, we propose an *Efficient Model de-*

fense with noise transition MAtrix (EMMA) to mitigate model stealing attacks. As depicted in the "Defense phase" of Figure 2, the attacker iteratively queries the victim model using unlabeled data to obtain posteriors via the prediction API. The victim model receives these queries and outputs corresponding predicted probability vectors. These posteriors then multiply the optimized matrix T^n, becoming perturbed posteriors that remain useful to benign users, and returned to the attacker. However, if the attacker collects these query-responses as training data to reconstruct a clone model, in that case, this dataset is equivalent to noisy labeled data with instance-wise weight, thus impeding cloning model training. To optimize the transition matrix with maximum utility, as shown in the "Optimizing phase", we devise a bi-level optimization between the victim and surrogate model which is devised to represent the attacker model. In each round, given initial T^t and surrogate weights $\boldsymbol{\theta}_g^t$, the optimized matrix T^{t+1} stems from two gradient descents (GD) on the attack loss and defense loss, respectively. The surrogate model then updates through a GD on the attack loss. This process loops until reaching the termination condition, yielding the final optimized matrix T^n. In this section, we first elucidate the defense mechanism of incorporating the transition matrix during the defensive phase. Then, a bi-level optimization approach is expounded to alternately optimize the matrix and update the surrogate model.

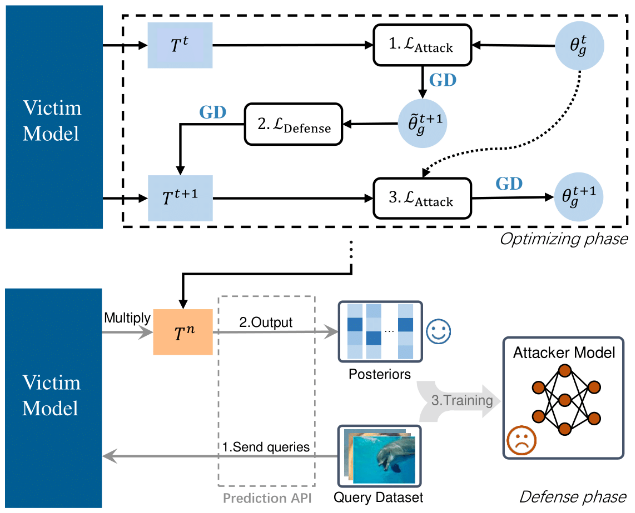


Figure 2. An illustration of "Optimizing phase" and "Defense phase" in the framework. "Optimizing phase" demonstrates the process of updating the noise transition matrix T and surrogate network weights $\boldsymbol{\theta}_g$ alternatively. "Defensive phase" shows the interaction process where the attacker tries to steal the functionality from the defender.

4.1. The underlying mechanism

To understand how the transition matrix defends against attacks, we analyze the classification risks of two scenarios: "learning on noisy labels" and "learning on query-response pairs with perturbed posteriors". It is found that the two classification risks differ only by an instance-wise weight coefficient multiplied to the latter. Let $\tilde{y} \in \mathbf{Y}$ denote the noisy label, where $\tilde{y}$ is selected based on the transition matrix and the true label. Theorem 1 formalizes the defensive mechanism.

Theorem 1. *Let $\mathcal{L}(\cdot,\cdot)$ denote the classification loss function. With sufficiently many queries, the classification risk $\mathfrak{R}(h)$ of learning with noisy labels is $\mathbb{E}_{(\boldsymbol{x},\tilde{y})}[\mathcal{L}(h(\boldsymbol{x}),\tilde{y})]$. The classification risk $\mathfrak{R}_T(h)$ of learning with query-response pairs is $\mathbb{E}_{(\boldsymbol{x},\tilde{y})}[q(\boldsymbol{x},\tilde{y})\cdot\mathcal{L}(h(\boldsymbol{x}),\tilde{y})]$, where $q(\boldsymbol{x},\tilde{y})$ denotes the instance-wise weight, proportional to the closeness between the predicted posterior and the true label.*

The proof can be found in Appendix 8. Theorem 1 demonstrates that training a clone model on such query-response pairs with perturbed posteriors is equivalent to training it on noisy-labeled data with instance-wise weights. It can be further deduced that the more closely the victim model's predicted posteriors match the true label, the more this training process resembles learning with noisy labels. In addition, the clone model's generalization can be futher degraded by increasing the transition matrix noise rate or exploring more difficult noise type through inter-class correlations, such as using class-conditional noise instead of random noise.

4.2. The overall training framework

Since the attacker's network is unknown, we cannot directly measure its cloning performance. Instead, we represent the attacker with a surrogate model g parameterized by $\boldsymbol{\theta}_g$ and design optimal perturbations for it. By using the surrogate model to simulate the cloning process, we can continuously optimize the transition matrix during this process to minimize the utility trade-off between attacker performance and defender utility. To achieve this, we first introduce the respective objective functions for the attacker and defender. Then, a bi-level optimization approach is proposed.

Given the perturbed posterior $f(\boldsymbol{x};\boldsymbol{\theta}_f)T$, the attacker's objective function is defined as the loss on the gathered query-response pairs, formulated as:

$$\mathcal{L}_{\text{attack}}(\boldsymbol{\theta}_g,T) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_{\text{pair}}}[\mathcal{L}_{\text{kna}}(g(\boldsymbol{x};\boldsymbol{\theta}_g), f(\boldsymbol{x};\boldsymbol{\theta}_f)T)]. \quad (3)$$

Notably, the optimization of $\boldsymbol{\theta}_g$ is influenced by T. To defend against model stealing, we would like T to regularize $\boldsymbol{\theta}_g$ to have poor generalizability. To achieve this, we instantiate a misinformation loss that inhibits the attacker model's outputs on the victim model's original predicted label $k = \arg\max_i f(\boldsymbol{x},\boldsymbol{\theta}_f)_i$. Formally, we derive the following modified negative log-likelihood loss:

$$\mathcal{L}_{\text{mis}}(\boldsymbol{\theta}_g) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_{\text{pair}}}[-\log(1-g(\boldsymbol{x};\boldsymbol{\theta}_g)_k)]. \quad (4)$$

The underlying principle is quite simple: *The log-likelihood of the original prediction label on the output should be minimized*. Despite its simple form, optimizing this loss has been empirically shown to be an effective defense against model stealing attacks [20, 59].

Moreover, to maintain fidelity for the victim model, a straightforward strategy is aligning the perturbed output with the predicted label k. This is implemented by minimizing the divergence between the output and the predicted label, formulated as:

$$\mathcal{L}_{\text{align}}(T) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_{\text{pair}}}[-\log(f(\boldsymbol{x};\boldsymbol{\theta}_f)T)_k]. \tag{5}$$

In summary, the overall objective function of the defender can be represented as the following defense loss:

$$\mathcal{L}_{\text{defense}}(\boldsymbol{\theta}_g, T) = \lambda\cdot\mathcal{L}_{\text{align}}(T) + (1-\lambda)\cdot\mathcal{L}_{\text{mis}}(\boldsymbol{\theta}_g), \tag{6}$$

where the multiplicative factor λ is used to balance the contribution of these two loss terms.

Bi-level optimization. Given the defense loss $\mathcal{L}_{\text{defense}}$, the surrogate model weights $\boldsymbol{\theta}_g$ need to be updated first before optimizing the transition matrix T. For a mini-batch sample, we propose that surrogate model weights could be treated as a latent variable and optimized alternatively with the matrix. This yields the following bi-level optimization problem:

$$\begin{aligned} &\arg\min_{T} \mathcal{L}_{\text{defense}}(\boldsymbol{\theta}_g^*, T) \\ &\text{subject to } \boldsymbol{\theta}_g^* = \arg\min_{\boldsymbol{\theta}_g} \mathcal{L}_{\text{attack}}(\boldsymbol{\theta}_g, T) \end{aligned} \tag{7}$$

Bi-level optimization problem has been proved strongly NP-hard [14], so getting such an exact solution for problem (7) is impossible in polynomial time. Fortunately, the inner optimization problem in (7) can be transformed into an optimized result $\boldsymbol{\theta}_g(T)$ as a function of T. Then $\boldsymbol{\theta}_g(T)$ can then substitute $\boldsymbol{\theta}_g$ in the outer optimization to acquire an objective solely in terms of T. This allows approximating the solution using gradient descent. Therefore, we employ an alternating optimization strategy to approximately solve this problem. In each iteration $\boldsymbol{\theta}_g$ and T are updated on-the-fly in an alternating manner.

Optimizing the transition matrix. In step $t+1$ of optimizing T^t, guided by the attack loss, surrogate model weights $\boldsymbol{\theta}_g^t$ are updated in a one-step-forward manner to obtain $\tilde{\boldsymbol{\theta}}_g^{t+1}$. Specifically, with the fixed T^t, then $\boldsymbol{\theta}_g^t$ is updated via gradient descent with learning rate α as:

$$\tilde{\boldsymbol{\theta}}_g^{t+1}(T^t) = \boldsymbol{\theta}_g^t - \alpha\cdot\nabla_{\boldsymbol{\theta}_g^t}\mathcal{L}_{\text{attack}}. \tag{8}$$

Note that the surrogate model g does not actually update in this step, but makes preparations for estimating T^{t+1}. We then optimize T^t with $\tilde{\boldsymbol{\theta}}_g^{t+1}$ fixed under the defense loss. The motivation is that we would like to find a T^{t+1} that achieves minimal utility trade-offs between the defender and attacker. The transition matrix T^t is updated via gradient descent with learning rate β:

Algorithm 1 The overall training procedure

```
Input: Query dataset $\mathcal{D}_{\text{query}}$;
       The surrogate model $g$ and its parameters $\boldsymbol{\theta}_g$;
       The noise transition martix $T$.
       The number of iterations $I$;
       The multiplicative factor $\lambda$;
Procedure:
1: Initialize $T$ and $\boldsymbol{\theta}_g$;
2: for $i = 1$ to $I$ do
3:     Fetch a random batch $\mathcal{B}$ from $\mathcal{D}_{\text{query}}$;
4:     Calculate the one-step-forward surrogate model
       weights via Eq. (8);
5:     Update the transition matrix via Eq. (9-12);
6:     Update surrogate model weights via Eq. (13);
7: end for
Output: Learned noise transition matrix.
```

$$w^{t+1} = T^t - \beta\cdot\nabla_{T^t}\mathcal{L}_{\text{Defense}}. \tag{9}$$

Apply chain rule to Eq. (9), we can get:

$$\begin{aligned} w^{t+1} &= T^t - \beta\cdot\{\lambda\cdot\nabla_{T^t}\mathcal{L}_{\text{align}} \\ &+ (1-\lambda)\cdot\nabla_{\tilde{\boldsymbol{\theta}}_g^{t+1}}\mathcal{L}_{\text{mis}}\cdot(-\alpha\,\nabla^2_{\boldsymbol{\theta}_g^t,T^t}\,\mathcal{L}_{\text{attack}})\}. \end{aligned} \tag{10}$$

Note that w^{t+1} is not the final transition matrix, as its entries may not satisfy the properties of a valid transition matrix. Specifically, all elements must be non-negative and each row must be normalized. To address this, we first replace any negative elements in w^{t+1} with zeros via:

$$\tilde{w}^{t+1} = \max(w^{t+1}, 0). \tag{11}$$

Then, each row is normalized to get the final valid transition matrix. For the j^{th} row of $\tilde{w}^{t+1}$, which indicates the probabilities of class j transitioning to other classes, a normalization is performed to obtain the final matrix $T_{j,i}^{t+1}$:

$$T_{j,i}^{t+1} = \frac{\tilde{w}_{j,i}^{t+1}}{\sum_i \tilde{w}_{j,i}^{t+1} + \delta(\sum_i \tilde{w}_{j,i}^{t+1})}, \delta(a) = \begin{cases} 1, \text{if } a=0 \\ 0, \text{if } a\neq 0 \end{cases} \tag{12}$$

where $\delta(\cdot)$ is used to avoid division by 0.

Updating surrogate model weights. After optimizing the matrix, we conduct the surrogate model update by keeping T^{t+1} fixed and minimizing the attack loss. New surrogate model weights $\boldsymbol{\theta}_g^{t+1}$ are obtained via gradient descent with learning rate γ:

$$\boldsymbol{\theta}_g^{t+1} = \boldsymbol{\theta}_g^t - \gamma\cdot\nabla_{\boldsymbol{\theta}_g^t}\mathcal{L}_{\text{attack}}. \tag{13}$$

The whole training procedure is summarized in Algorithm 1.

Method	CIFAR100 → CIFAR10			CIFAR10 → CIFAR100			Caltech256 → CUB200		
	ΔAcc = 1	ΔAcc = 2	ΔAcc = 5	ΔAcc = 1	ΔAcc = 2	ΔAcc = 5	ΔAcc = 1	ΔAcc = 2	ΔAcc = 5
Random	84.17	77.21	74.21	43.23	39.81	32.12	33.50	31.14	28.01
AM	77.10	72.93	57.56	39.82	32.98	19.36	26.17	21.27	14.31
MAD	80.57	77.54	69.07	36.56	32.59	26.65	37.16	35.23	31.50
NASTY	76.65	74.31	64.19	38.53	33.61	26.95	31.06	28.23	24.70
GARD^2	77.42	73.88	71.40	41.21	39.29	35.94	35.69	33.67	32.34
MIN-IP	78.88	73.40	70.66	42.33	38.67	34.26	34.73	32.26	28.94
EMMA	**61.79**	**52.91**	**42.21**	**28.01**	**18.34**	**13.37**	**19.10**	**15.79**	**13.11**
	ImageNet-C10 → CIFAR10			ImageNet-C100 → CIFAR100			ImageNet-CUB200 → CUB200		
	ΔAcc = 1	ΔAcc = 2	ΔAcc = 5	ΔAcc = 1	ΔAcc = 2	ΔAcc = 5	ΔAcc = 1	ΔAcc = 2	ΔAcc = 5
Random	88.99	88.18	87.07	63.16	61.76	58.64	46.58	44.06	40.49
AM	87.55	84.60	**76.72**	60.57	56.09	47.10	45.12	39.65	30.15
MAD	88.67	85.70	78.44	58.74	56.23	50.85	50.98	48.57	44.47
NASTY	87.33	85.33	81.68	59.79	56.05	48.95	46.56	43.23	37.76
GARD^2	87.29	86.06	85.93	58.39	56.71	53.63	50.27	47.55	43.11
MIN-IP	87.40	86.81	84.45	61.17	56.35	51.98	50.09	46.45	42.06
EMMA	**84.10**	**80.60**	77.52	**56.73**	**49.18**	**31.48**	**36.35**	**33.10**	**26.33**

Table 1. Accuracy (%) of attacker model on defender's test set under knowledge-limited (top row) and distribution-aware (bottom row) attack scenarios with varying budgets. The best result in each column is highlighted in **bold**.

Eval Data	Defender Accuracy	Attacker Accuracy	
		KL	DA
CIFAR10	95.33 ± 0.13	85.17 ± 0.83	91.78 ± 0.19
CIFAR100	75.96 ± 0.09	51.95 ± 1.12	66.82 ± 0.16
CUB200	81.44 ± 0.46	48.93 ± 0.37	62.12 ± 0.22

Table 2. Accuracy (%) of the defender model and the attacker model with no defense applied.

5. Experiments

To demonstrate the effectiveness of our proposed defense method, we conduct experiments on multiple image datasets and present the results in this section.

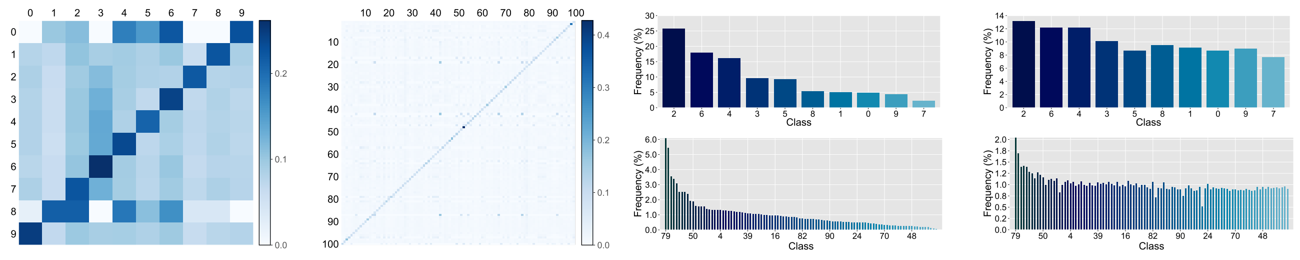
5.1. Experiment Setup

Datasets. We employ three datasets for evaluation: CIFAR10, CIFAR100 [26] and CUB200 [57]. For each dataset, we explore two attacker scenarios - knowledge-limited (KL) and distribution-aware (DA). The KL scenario uses query sets of CIFAR100, CIFAR10, and Caltech256 [8], paired with CIFAR10, CIFAR100, and CUB200 evaluation datasets respectively. The DA scenario employs tailored query sets from ImageNet-1K [4] manually selected by prior work [38], specifically ImageNet-C10, ImageNet-C100, and ImageNet-CUB200 containing 183,763, 161,653, and 30,000 examples matched to the evaluation datasets, with their further details provided in the Appendix 9.

Compared methods. We compare our method against the following six model stealing defense baselines: (1) *No Defense*, a naive approach where no defense is attemptted. (2) *Random*, an interpolation-based method that perturbs the defender's predicted posterior by interpolating it with a one-hot posterior vector, where the index of single non-zero entry is randomly selected from the non-argmax prediction labels. (3) *AM* [20], a detection-based method that identifies out-of-distribution queries and interpolates with posteriors from a misinformation network, where the interpolation weight is denoted by the confidence of the argmax prediction label. (4) *MAD* [40], an accuracy-constrained method that adds controlled noise to original posteriors so that it maximizes the angular deviation between the perturbed and original gradient signal on a surrogate model. The surrogate model is randomly initialized. (5) *NASTY* [34], a self-undermining knowledge distillation approach which trains a nasty teacher network by maximizing the difference between the output of the nasty teacher and a normal pretrained network. (5) *GARD*^{2} [38], a coordinated gradient redirection defense method, which perturbs original posteriors by setting the target gradient direction to be the all-ones vector. This redirects the attacker to update towards a consistent target direction. (6) *MIN-IP* [38], an uncoordinated gradient redirection defense method, which perturbs original posteriors by setting the target gradient direction opposite the original gradient direction, redirecting the update gradient of attackers opposite to the original gradients. The surrogate model of *MIN-IP* and *GARD*^{2} train on the attacker's queries with early stopping after 10 epochs.

For fair comparisons, we basically use the same network architecture, learning rate, optimizer, and augmentation strategy across all the comparing methods.

Implementation. Our experiment pipeline has three stages:



(a) Heatmap of transition matrix. (b) Heatmap of transition matrix. (c) Average predicted posterior (d) Average perturbed posterior

Figure 3. Qualitative analysis of the transition matrix. (a-b): Heatmaps of optimized transition matrixes for distribution-aware attacks (a) Imagenet-C10→CIFAR10 and (b) Imagenet-C100→CIFAR100, respectively. (c-d): Average (c) predicted and (d) perturbed posteriors across the query set of (top) Imagenet-C10 and (bottom) Imagenet-C100 datasets.

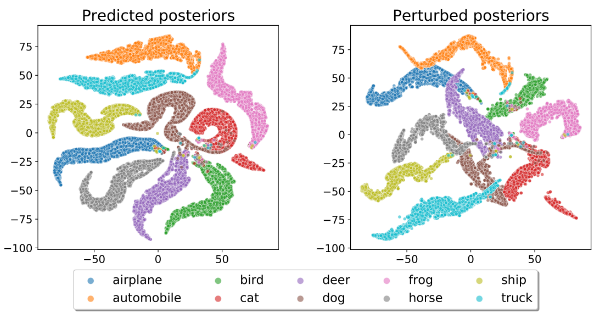


Figure 4. Visualizations of predicted and perturbed posteriors.

First, the defender model trains on each evaluation dataset. Second, defense methods generate perturbed posteriors for query dataset and evaluation dataset at various defense budgets. Finally, the attacker model trains on the collected query-response pairs. We denote attacks with query data $\mathcal{Q}$ and evaluation data $\mathcal{D}$ as $\mathcal{Q} \to \mathcal{D}$.

We use ResNet34 for CIFAR10/100 and ResNet50 pretrained on ImageNet for CUB200. For all models training, we utilize SGD as the optimizer with a momentum of 0.9 and weight decay of 5e-4 for 100 epochs. Initial learning rates are set as 0.01 for CUB200 and 0.1 for other datasets, with a cosine annealing schedule. For optimizing the transition matrix, the learning rate γ is set as 0.3 for CIFAR10 and 1.0 for other datasets. The surrogate model's one-step-forward learning rate α and the actual learning rate β are set the same value as specified in the model training settings. Additionally, an early stopping mechanism is set to terminates optimization when the average change in T between consecutive epochs falls below a specified threshold. The threshold is 1e-2 for CIFAR10 and 1e-4 for other datasets. Furthermore, the balancing factor λ is adjusted to achieve different utility budgets. Recent work [38] proposed a novel approach to initialize the surrogate model. They suggested to train the surrogate model through distilling from the victim model with early stopping. This would make it more representative of the actual attacker's behavior. Following this approach, we initialize our surrogate model by training it on the query dataset with the same early stopping mechanism.

5.2. Experiment Results

Table 2 presents the accuracy of the defender model and the attacker model, without any defense applied under two different query distributions. This shows the upper bound for the usage performance of the defender and stealable performance of the attacker. Table 1 compares the accuracy of an attacker's clone model at different defender accuracy drop budgets ΔAcc. The budget refers to the average decrease in accuracy the defender causes across the test set by adding perturbations. Corresponding visualizations of these results are provided in the Appendix 10. As shown, our defense method achieves the lowest utility trade-off across budgets. For example, in the CIFAR100→CIFAR10 attack scenario, our method reduces the attacker model's accuracy by 23% while only decreasing the defender's accuracy by 1%.

5.3. Qualitative Analysis

Perturbed posteriors make cloning difficult. We visualize the predicted and perturbed posteriors using t-SNE in Figure 4. The perturbed posteriors clearly exhibit degraded inter-class distances compared to the predicted posteriors. This degradation obfuscates the decision boundaries between classes, making it difficult for the attacker to clone the defender model by imitating the posteriors. Notably, such effective defenses are achieved solely through linear mapping of the posteriors, demonstrating the simplicity and efficiency of our proposed noise perturbation strategy.

The transition matrix induces challenging noise. We analyze the defense details of two attack scenarios: Imagenet-C10→CIFAR10 and Imagenet-C100→CIFAR100 in Figure 3. Specifically, Figure 3 (a-b) visualizes corresponding optimized transition matrixes. It can be observed that the diagonal elements are maximized for fidelity, while latent inter-label relationships are also exploited. As shown the optimized matrix in Figure 3 (a), despite it causes a 2.35% drop in the defender's accuracy on the test set, the optimization makes it have a noise rate as high as 77%. This negatively affects the cloning process of the attacker, reducing the cloned model's accuracy by 14%.

In Figures 3 (c-d), we visualize the average posterior of predicted and perturbed predictions. Given that ImageNet-C10/C100 follow long-tail distributions as illustrated in appendix 9, the predicted posteriors also exhibit long-tailed shapes as expected. However, the perturbations significantly increase the confidence of tail classes, distorting the true inter-class relationships. This distortion of class relationships also makes stealing more difficult for the attacker.

5.4. Ablation Study

Effectiveness of optimized transition matrix. To validate the effectiveness of the optimized transition matrix, we compare the defense performance between it and the uniform transition matrix with an identical noise rate. The results in Table 3 show that the optimized transition matrix can further reduce the attacker's accuracy while maintaining the defender's accuracy. This demonstrates that the optimized matrix provides improved defense over the uniform matrix. Overall, this shows that capturing the inter-class relationships during optimization contributes to generating a matrix that is more adversarial against the attacker.

Attack Scenarios	Optimized matrix		Uniform matrix	
	Defender	Attacker	Defender	Attacker
CIFAR100→CIFAR10	93.87±0.14	32.41±7.54	94.81±0.06	71.43±2.26
CIFAR10→CIFAR100	74.48±0.25	15.65±0.72	74.75±0.07	35.67±3.44
Caltech256→CUB200	80.45±0.13	18.25±0.43	80.89±0.22	30.15±0.21

Table 3. Defender and attacker accuracy (%) with optimized and uniform transition matrix under attack scenarios for CIFAR10, CIFAR100 and CUB200 evaluation datasets.

Effects of the number of queries. We also verify the effectiveness of our defense against attacks with varying number of queries in Figure 5. The EMMA-top1 result is depicted as a reference, which denotes the victim model only outputting the predicted label rather than the full posterior. As formalized in Theorem 1, when the victim model assigns full confidence to the true label, the attacker's cloning task degrades to learning from noisy labels. Thus, EMMA-top1 signifies the optimal defense under the same transition matrix. EMMA defense consistently forces degradation of the attacker accuracy across all query set sizes, remaining close to the EMMA-top1. As the query increases, the attacker model obtained from the victim model suffers more accuracy drops, revealing that our defense induces more significant detrimental impacts when the attacker uses more queries.

Transferability across attack scenarios. To explore the transition matrix's transferable defense capabilities, we test it against an attack type different from the one it was optimized for. As shown in Figure 6, we use the matrix optimized under DA attack to defend against KL attack, denoted as "DA→KL". Compared to a matrix specifically optimized for KL attack, denoted as "KL→KL", it still maintained

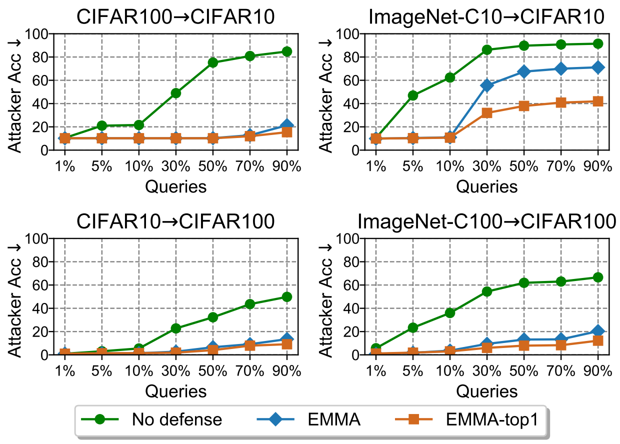


Figure 5. Effects of the number of queries.

considerable defensive efficacy, exhibiting the optimized transition matrix's robust transferable defense capabilities across different attack types.

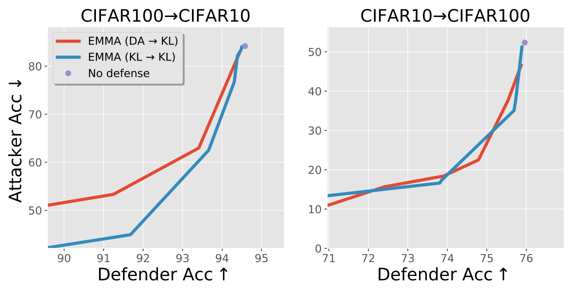


Figure 6. Transferability across attack scenarios.

Due to the space limitation, additional experiments of our method are provided in Appendix.

6. Conclusion

In this work, we revisited the noise transition matrix in a novel way by applying it to model stealing defense for the first time. This deteriorates the attacker's cloning process to a level equivalent to training on noisy labels, hampering attacker model generalization. To optimize the transition matrix, we proposed a novel bi-level optimization training framework that incorporats fidelity training for the victim model and adversarial training for the surrogate model. Experimental results under various attack scenarios showed the effectiveness and robustness of our proposed defense method compared to current state-of-the-art methods, achieving strong defense with minimal utility trade-off.

7. Acknowledgments

The authors wish to thank the anonymous reviewers for their helpful comments and suggestions. This work was supported by the National Science Foundation of China (62225602), and the Big Data Computing Center of Southeast University.

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 Security Symposium*, pages 1615–1631, 2018. 3

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *International Conference on Advanced Neural Information Processing Systems*, pages 1877–1901, 2020. 1

[3] Yanjiao Chen, Rui Guan, Xueluan Gong, Jianshuo Dong, and Meng Xue. D-dae: Defense-penetrating model extraction attacks. In *IEEE Symposium on Security and Privacy*, pages 382–399, 2023. 1

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE/CVF Computer Vision and Pattern Recognition Conference*, pages 248–255, 2009. 6

[5] Adam Dziedzic, Muhammad Ahmad Kaleem, Yu Shen Lu, and Nicolas Papernot. Increasing the cost of model extraction with calibrated proof of work. In *International Conference on Learning Representations*, 2022. 1

[6] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *ACM SIGSAC conference on computer and communications security*, pages 1322–1333, 2015. 1

[7] Karan Ganju, Qi Wang, Wei Yang, Carl A Gunter, and Nikita Borisov. Property inference attacks on fully connected neural networks using permutation invariant representations. In *ACM SIGSAC conference on computer and communications security*, pages 619–633, 2018. 1

[8] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007. 6

[9] Jun Guo, Aishan Liu, Xingyu Zheng, Siyuan Liang, Yisong Xiao, Yichao Wu, and Xianglong Liu. Isolation and induction: Training robust deep neural networks against model stealing attacks. In *ACM International Conference on Multimedia*, 2023. 1

[10] Bo Han, Quanming Yao, Tongliang Liu, Gang Niu, Ivor W Tsang, James T Kwok, and Masashi Sugiyama. A survey of label-noise representation learning: Past, present and future. *arXiv preprint arXiv:2011.04406*, 2020. 3

[11] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor W. Tsang, and Masashi Sugiyama. Robust training of deep neural networks with extremely noisy labels. In *International Conference on Advanced Neural Information Processing Systems*, page 4, 2020. 2

[12] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 2, 3

[13] Surgan Jandial, Yash Khasbage, Arghya Pal, Vineeth N Balasubramanian, and Balaji Krishnamurthy. Distilling the undistillable: Learning from a nasty teacher. In *European Conference on Computer Vision*, pages 587–603, 2022. 2

[14] Robert G Jeroslow. The polynomial hierarchy and a simple model for competitive analysis. *Mathematical programming*, pages 146–164, 1985. 5

[15] Hengrui Jia, Christopher A Choquette-Choo, Varun Chandrasekaran, and Nicolas Papernot. Entangled watermarks as a defense against model extraction. In *USENIX Security Symposium*, pages 1937–1954, 2021. 2, 3

[16] Hengrui Jia, Mohammad Yaghini, Christopher A Choquette-Choo, Natalie Dullerud, Anvith Thudi, Varun Chandrasekaran, and Nicolas Papernot. Proof-of-learning: Definitions and practice. In *IEEE Symposium on Security and Privacy*, pages 1039–1056, 2021. 3

[17] Wenbo Jiang, Hongwei Li, Guowen Xu, Tianwei Zhang, and Rongxing Lu. A comprehensive defense framework against model extraction attacks. *IEEE Transactions on Dependable and Secure Computing*, 2023. 1

[18] Ishan Jindal, Matthew Nokleby, and Xuewen Chen. Learning deep networks from noisy labels with dropout regularization. In *IEEE International Conference on Data Mining*, pages 967–972, 2016. 3

[19] Mika Juuti, Sebastian Szyller, Samuel Marchal, and N Asokan. Prada: protecting against dnn model stealing attacks. In *IEEE European Symposium on Security and Privacy*, pages 512–527, 2019. 1, 2, 3

[20] Sanjay Kariyappa and Moinuddin K Qureshi. Defending against model stealing attacks with adaptive misinformation. In *IEEE/CVF Computer Vision and Pattern Recognition Conference*, pages 770–778, 2020. 1, 2, 3, 5, 6

[21] Sanjay Kariyappa, Atul Prakash, and Moinuddin K Qureshi. Protecting dnns from theft using an ensemble of diverse models. In *International Conference on Learning Representations*, 2020. 2

[22] Sanjay Kariyappa, Atul Prakash, and Moinuddin K Qureshi. Maze: Data-free model stealing attack using zeroth-order gradient estimation. In *CVPR*, 2021. 1, 2

[23] Manish Kesarwani, Bhaskar Mukhoty, Vijay Arya, and Sameep Mehta. Model extraction warning in mlaas paradigm. In *Annual Computer Security Applications Conference*, pages 371–380, 2018. 1

[24] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *International Conference on Computer Vision*, 2023. 1

[25] Kalpesh Krishna, Gaurav Singh Tomar, Ankur P Parikh, Nicolas Papernot, and Mohit Iyyer. Thieves on sesame street! model extraction of bert-based apis. In *International Conference on Learning Representations*, 2020. 2

[26] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6

[27] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *International Conference on Advanced Neural Information Processing Systems*, pages 1106–1114, 2012. 1

[28] Souvik Kundu, Qirui Sun, Yao Fu, Massoud Pedram, and Peter Beerel. Analyzing the confidentiality of undistillable teachers in knowledge distillation. pages 9181–9192, 2021. 2

[29] Jeonghyun Lee, Sungmin Han, and Sangkyun Lee. Model stealing defense against exploiting information leak through the interpretation of deep neural nets. In *International Joint Conference on Artificial Intelligence*, pages 710–716, 2022. 1, 3

[30] Taesung Lee, Benjamin Edwards, Ian Molloy, and Dong Su. Defending against neural network model stealing attacks using deceptive perturbations. In *IEEE Security and Privacy Workshops*, pages 43–49, 2019. 2, 3

[31] Simin Li, Shuning Zhang, Gujun Chen, Dong Wang, Pu Feng, Jiakai Wang, Aishan Liu, Xin Yi, and Xianglong Liu. Towards benchmarking and assessing visual naturalness of physical world adversarial attacks. In *IEEE/CVF Computer Vision and Pattern Recognition Conference*, pages 12324–12333, 2023. 1

[32] Yiming Li, Linghui Zhu, Xiaojun Jia, Yong Jiang, Shu-Tao Xia, and Xiaochun Cao. Defending against model stealing via verifying embedded external features. In *Association for the Advance of Artificial Intelligence*, pages 1464–1472, 2022. 3

[33] Shunchang Liu, Jiakai Wang, Aishan Liu, Yingwei Li, Yijie Gao, Xianglong Liu, and Dacheng Tao. Harnessing perceptual adversarial patches for crowd counting. In *ACM SIGSAC Conference on Computer and Communications Security*, pages 2055–2069, 2022. 1

[34] Haoyu Ma, Tianlong Chen, Ting-Kuei Hu, Chenyu You, Xiaohui Xie, and Zhangyang Wang. Undistillable: Making a nasty teacher that cannot teach students. In *ICLR*, 2021. 6

[35] Ke Ma, Qianqian Xu, Jinshan Zeng, Xiaochun Cao, and Qingming Huang. Poisoning attack against estimating from pairwise comparisons. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 6393–6408, 2021. 1

[36] Ke Ma, Qianqian Xu, Jinshan Zeng, Guorong Li, Xiaochun Cao, and Qingming Huang. A tale of hodgerank and spectral method: Target attack against rank aggregation is the fixed point of adversarial game. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 4090–4108, 2022. 1

[37] Pratyush Maini, Mohammad Yaghini, and Nicolas Papernot. Dataset inference: Ownership resolution in machine learning. In *International Conference on Learning Representations*, 2021. 3

[38] Mantas Mazeika, Bo Li, and David Forsyth. How to steer your adversary: Targeted and efficient model stealing defenses with gradient redirection. In *International Conference on Machine Learning*, pages 15241–15254, 2022. 2, 3, 6, 7

[39] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Knockoff nets: Stealing functionality of black-box models. In *IEEE/CVF Computer Vision and Pattern Recognition Conference*, pages 4954–4963, 2019. 1, 2, 3

[40] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Prediction poisoning: Towards defenses against dnn model stealing attacks. In *International Conference on Learning Representations*, 2020. 2, 3, 6

[41] Soham Pal, Yash Gupta, Aditya Shukla, Aditya Kanade, Shirish Shevade, and Vinod Ganapathy. Activethief: Model extraction using active learning and unannotated public data. In *Association for the Advance of Artificial Intelligence*, pages 865–872, 2020. 2

[42] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *ACM on Asia conference on computer and communications security*, pages 506–519, 2017. 1, 2, 3

[43] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *IEEE/CVF Computer Vision and Pattern Recognition Conference*, pages 1944–1952, 2017. 2, 3

[44] Sunandini Sanyal, Sravanti Addepalli, and R Venkatesh Babu. Towards data-free model stealing in a hard label setting. In *CVPR*, 2022. 2

[45] Zeyang Sha et al. Can't steal? cont-steal! contrastive stealing attacks against image encoders. In *CVPR*, 2023. 2

[46] Yu Shi, Dong-Dong Wu, Xin Geng, and Min-Ling Zhang. Robust representation learning for unreliable partial label learning. *arXiv preprint arXiv:2308.16718*, 2023. 3

[47] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *IEEE symposium on security and privacy*, pages 3–18, 2017. 1

[48] Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels. *arXiv preprint arXiv:1406.2080*, 2014. 3

[49] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2013. 1

[50] Sebastian Szyller, Buse Gul Atli, Samuel Marchal, and N Asokan. Dawn: Dynamic adversarial watermarking of neural networks. In *ACM International Conference on Multimedia*, pages 4417–4425, 2021. 1, 3

[51] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction {APIs}. In *USENIX security symposium*, pages 601–618, 2016. 1, 2, 3

[52] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *International Conference on Advanced Neural Information Processing Systems*, 2017. 1

[53] M Mitchell Waldrop et al. No drivers required. *Nature*, page 20, 2015. 1

[54] Binghui Wang and Neil Zhenqiang Gong. Stealing hyperparameters in machine learning. In *IEEE symposium on security and privacy*, pages 36–52, 2018. 1

[55] Jiakai Wang, Aishan Liu, Zixin Yin, Shunchang Liu, Shiyu Tang, and Xianglong Liu. Dual attention suppression attack: Generate adversarial camouflage in physical world. In *IEEE/CVF Computer Vision and Pattern Recognition Conference*, pages 8565–8574, 2021. 1

[56] Zhen Wang, Guosheng Hu, and Qinghua Hu. Training noise-robust deep neural networks via meta-learning. In *IEEE/CVF Computer Vision and Pattern Recognition Conference*, pages 4524–4533, 2020. 3

[57] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-ucsd birds 200. 2010. 6

[58] Hongwei Wen, Jingyi Cui, Hanyuan Hang, Jiabin Liu, Yisen Wang, and Zhouchen Lin. Leveraged weighted loss for partial label learning. In *International Conference on Machine Learning*, pages 11091–11100, 2021. 2

[59] Dong-Dong Wu, Deng-Bao Wang, and Min-Ling Zhang. Revisiting consistency regularization for deep partial label learning. In *International Conference on Machine Learning*, pages 24212–24225, 2022. 5

[60] Xiaobo Xia, Tongliang Liu, Nannan Wang, Bo Han, Chen Gong, Gang Niu, and Masashi Sugiyama. Are anchor points really indispensable in label-noise learning? In *International Conference on Advanced Neural Information Processing Systems*, pages 6835–6846, 2019. 3

[61] Hao Yang, You-Zhi Jin, Zi-Yin Li, Deng-Bao Wang, Xin Geng, and Min-Ling Zhang. Learning from noisy labels via dynamic loss thresholding. *IEEE Transactions on Knowledge and Data Engineering*, 2023. 3

[62] Honggang Yu, Kaichen Yang, Teng Zhang, Yun-Yun Tsai, Tsung-Yi Ho, and Yier Jin. Cloudleak: Large-scale deep learning models stealing through adversarial examples. In *Network and Distributed System Security Symposium*, 2020. 1

[63] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, pages 107–115, 2021. 2

[64] Jiliang Zhang, Shuang Peng, Yansong Gao, Zhi Zhang, and Qinghui Hong. Apmsa: adversarial perturbation against model stealing attacks. *IEEE Transactions on Information Forensics and Security*, pages 1667–1679, 2023. 1

[65] Guoqing Zheng, Ahmed Hassan Awadallah, and Susan Dumais. Meta label correction for noisy label learning. In *Association for the Advance of Artificial Intelligence*, pages 11053–11061, 2021. 3