Model Stealing Defense against Exploiting Information Leak through the Interpretation of Deep Neural Nets

Jeonghyun Lee , Sungmin Han , Sangkyun Lee*
School of Cybersecurity, Korea University
{nomar0107, sungmin_15, sangkyun}@korea.ac.kr

Abstract

Model stealing techniques allow adversaries to create attack models that mimic the functionality of black-box machine learning models, querying only class membership or probability outcomes. Recently, interpretable AI is getting increasing attention, to enhance our understanding of AI models, provide additional information for diagnoses, or satisfy legal requirements. However, it has been recently reported that providing such additional information can make AI models more vulnerable to model stealing attacks. In this paper, we propose DeepDefense, the first defense mechanism that protects an AI model against model stealing attackers exploiting both class probabilities and interpretations. DeepDefense uses a misdirection model to hide the critical information of the original model against model stealing attacks, with minimal degradation on both the class probability and the interpretability of prediction output. DeepDefense is highly applicable for any model stealing scenario since it makes minimal assumptions about the model stealing adversary. In our experiments, DeepDefense shows significantly higher defense performance than the existing state-of-the-art defenses on various datasets and interpreters.

1 Introduction

The uprising use of artificial intelligence technology has brought up many technical challenges [Stoica *et al.*, 2017]. Among them, the interpretability of AI has been getting focused recently. For instance, DARPA announced their XAI (eXplainable AI) program to promote interpretable AI research so that one can build more effective intelligent autonomous systems [Gunning, 2016]. Furthermore, the EU GDPR (General Data Protection Regulation) assures the right to get explanations on AI's decisions when requested by a person legally affected by the decision [GDPR, 2016]. Interpretable AI can be a valuable tool for inspecting fairness issues in automated decisions [Chouldechova and Roth, 2020].

*Corresponding author

A common way to interpret why a machine learning model has made a particular decision is to create an attribution map representing the importance of input features of a given data point for the decision. There have been a plethora of recent studies in this direction. For example, input saliency methods [Lundberg and Lee, 2017; Petsiuk *et al.*, 2018; Kapishnikov *et al.*, 2019] try to estimate the saliency of input features by perturbing a subset of the features and inspecting the change in the model's outcome. Attention-based methods use the output of an intermediate layer of a DNN, with gradients or other importance assessment of activation maps [Zhou *et al.*, 2016; Selvaraju *et al.*, 2017; Sattarzadeh *et al.*, 2021]. The relevance-propagation methods [Bach *et al.*, 2015; Montavon *et al.*, 2018; Nam *et al.*, 2020] use back-propagation of the relevance scores from the outcome throughout the layers of a neural net. Some hybrids of the above-mentioned approaches [Lee *et al.*, 2021; Gur *et al.*, 2021; Zhang *et al.*, 2021] have been applied for improved attribution quality.

One of the main issues is that the interpretation of the model, provided together with the model's class prediction scores, can reveal excessive information about a prediction model, making it vulnerable to model stealing attacks. Model stealing is a type of attack on AI models (especially ML-as-a-service models) creating an attack model using responses from a victim model with regard to queries made by an adversary, whose purpose is to avoid query charges, to violate training data privacy, or to prepare another type of attack such as evasion [Tramèr *et al.*, 2016; Papernot *et al.*, 2017; Orekondy *et al.*, 2019; Pal *et al.*, 2020]. Recently, [Milli *et al.*, 2019] has reported that the interpretation as additional information can be exploited by an adversary to enhance the performance of the clone model, showing that interpretations can be as informative as gradients in model stealing.

In this paper, we propose a novel defense mechanism against model stealing attacks for deep learning models that provide both class probability scores and interpretations in the form of attribution maps. Our contribution can be summarized as follows:

- We propose DeepDefense, the first defense mechanism to protect AI models against model stealing attackers exploiting both class probability scores and interpretations.

- In particular, we do not require assumptions about attackers' models or data, unlike the existing defense against model stealing.
- We formulate our DeepDefense as an optimization problem and provide an efficient strategy to solve it.
- Our DeepDefense outperforms the existing state-of-the-arts model stealing defense methods in experiments.

2 Related Work

Model stealing attacks [Tramèr *et al.*, 2016] suggested a model stealing attack method against confidential machine learning models such as in ML-as-a-service. [Orekondy *et al.*, 2019] suggested the knockoff attack framework where an adversary chooses his/her data distribution, sampling strategy, and the architecture of the attack model that can be different from the victim's choices. The authors showed that the prediction performance of the attack model improved when the adversary used data similar to the training data and models with larger learning capacity than the victim's model.

Recently, [Milli *et al.*, 2019] showed that a new type of model stealing attack where an adversary can use not only the victim's prediction output but also its interpretation could improve the prediction performance of the attack model.

Model stealing defenses Prediction Poisoning [Orekondy *et al.*, 2020] provides a perturbed softmax outcome instead of the model's original prediction output, so that it will distort the gradient of the attacker's training loss. To achieve the goal, the defender must have information about the attacker's model and loss function, which is hardly available.

Existing model stealing attacks [Papernot *et al.*, 2017; Orekondy *et al.*, 2019] rely on queries of which the distribution is quite different to that of victim (so called out-of-distribution with respect to the victim's data distribution). Based on the observation, Adaptive Misinformation [Kariyappa and Qureshi, 2020] suggests producing an incorrect outcome as a defense if a query is detected as out-of-distribution (OOD). Ensemble of Diverse Models [Kariyappa *et al.*, 2021] uses an ensemble of several classifiers to increase the diversity of prediction results on OOD queries.

To our best knowledge, the current defenses against model stealing consider attackers using only the prediction outcome from a victim. In this paper, we propose a novel defense against model stealing that uses both attribution map and prediction outcome of a victim.

Deep neural network interpreters Among many interpreters available for deep neural networks, we focus on Gradient $\odot$ Input, Grad-CAM, and Relevance-CAM as the representatives of gradient-based, activation-based, and hybrid methods, respectively.

Gradient $\odot$ Input makes use of input gradients multiplied with the input, which is known to be more robust to gradient saturation and thereby provides better visual quality than gradients only [Shrikumar *et al.*, 2017]. Grad-CAM [Selvaraju *et al.*, 2017] is one of the most popular methods for interpreting a deep neural network based on its activation maps. It generalizes the idea of CAM [Zhou *et al.*, 2016], making it unnecessary to modify the underlying neural network to have the global average pooling (GAP) structure at the last layer. Relevance-CAM [Lee *et al.*, 2021] is a combination of the Grad-CAM and a relevance-propagation method, Contrastive LRP [Gu *et al.*, 2018], replacing the channel weights of Grad-CAM with the relevance scores from Contrastive LRP.

3 Threat Model

Adversary We consider a black-box scenario where an adversary can send query inputs $x \in \mathcal{X}$ to a victim and receives the victim's outputs in the forms of class probabilities $f(x; w) \in \Delta^K$ and attribution maps $I(x; w) \in \mathbb{R}^{H \times W}$. The goal of the adversary is to train a surrogate model (we call an attack model) $f_A(\cdot; w_A)$ maximizing its prediction accuracy on unseen test data to be sampled from the victim's data distribution P. To achieve the goal, the adversary uses a transfer-set $\{(x_i, f(x_i; w), I(x_i; w))\}_{i=1}^B$ using the victim's outputs $f(x_i; w)$ and $I(x_i; w)$ of attacker's queries x_i to train the attack model f_A with the following loss [Milli *et al.*, 2019]:

$$\mathcal{L}(f_A(x; w_A), f(x)) + \lambda \|I(x) - I_A(x; w_A)\|_F^2. \quad (1)$$

Here, $\mathcal{L}$ is the cross entropy and $\| \cdot \|_F$ is the Frobenius norm. In line with the literature [Tramèr *et al.*, 2016; Orekondy *et al.*, 2019], we assume that the victim's data distribution P is unknown to the adversary, and the queries x_i of the transfer-set are sampled from the attacker's data distribution P_A, which is likely to be quite different to P: in this sense, we often refer the attacker's queries as out-of-distribution (OOD) with respect to P.

Defender The goal of a defender to model stealing is to reduce the test accuracy of the attack model while preserving the functionality and interpretability of the victim model in service for benign users. To preserve functionality, we like to keep the order of the top-k softmax probabilities. In terms of interpretability, we try to preserve the order of values in an attribution map since the values represent the relative importance of input features.

Knowledge of adversary's data, model, and loss function The state-of-the-art model stealing defense methods rely on specific assumptions about the adversary. For example, Prediction Poisoning (PP) [Orekondy *et al.*, 2020] requires the knowledge of the adversary's model and loss function to compute the loss gradients of the attack model. In addition, Adaptive Misinformation (AM) [Kariyappa and Qureshi, 2020] and Ensemble of Diverse Models (EDM) [Kariyappa *et al.*, 2021] rely on an effective OOD query detector, assuming that the defender has enough information about the statistical characteristic of the adversary's queries to build an effective detector. However, these assumptions can break easily in the real world, where attackers can try multiple models, loss functions, and queries to break the defense. With this in mind, we design our defense method not to require such assumptions about the adversary.

4 Proposed Method: DeepDefense

In this paper, we try to disturb the adversary's training by providing misdirected information of the victim model in both of

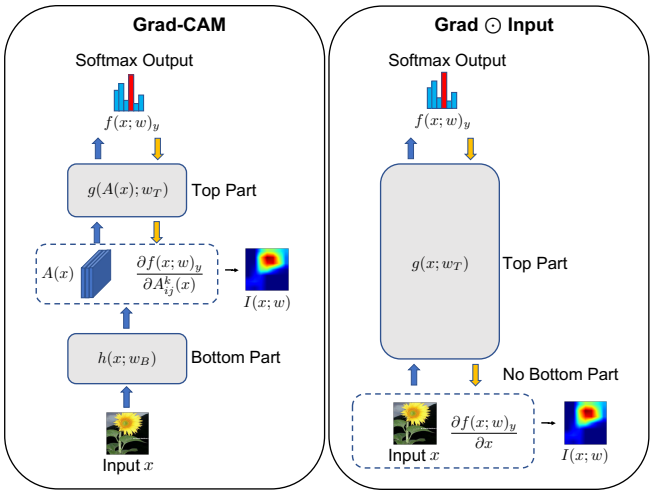

Figure 1: An illustration describing the top part $g(\cdot; w_T)$ and the bottom part $h(\cdot; w_B)$ of the vicim model $f(\cdot; w)$, $w = (w_T, w_B)$, on Grad-CAM (left) and Grad $\odot$ Input (right).

the prediction output, the softmax probabilities and an attribution map, without asking for the knowledge of the adversary's model, loss, and transfer-set.

4.1 Misdirection Model

To deal with model stealing adversaries who can exploit both softmax outputs and attribution maps of the victim model $f(\cdot; w)$, we propose a novel defense method called DeepDefense (DD). DD uses of the *misdirection model* $\widetilde{f}(\cdot; \widetilde{w})$ to make prediction outcomes for the queries from all users, preventing users from directly accessing the victim model. The purpose of the misdirection model is to disturb adversary's training of the attack model by misdirecting the key information of the victim model, while revealing only the information about the order of the top-k softmax output and the attribution values by accessing the gradient, the softmax output, and the attribution map of the victim model for a given query input.

This section describes the detailed requirements that our misdirection model $\widetilde{f}(\cdot; \widetilde{w})$ has to satisfy.

Gradient misdirection As in Prediction Poisoning (PP) [Orekondy *et al.*, 2020], we define the gradient w.r.t the weight parameters as the critical information to misdirect, which is an essential element for training models with gradient-based optimization algorithms. Before describing how to misdirect the gradient information, we first explain why we consider the weight gradient in two parts.

For a given input x, most interpreters generate their attribution maps by combining the information of two parts of given model f: the output of the bottom part $h(\cdot; w_B)$ and the gradient of the top part $\nabla g(\cdot; w_T)_y$ for the predicted class y. Figure 1 shows an illustration, where w_B and w_T are the weight sub-vectors corresponding to the bottom and the top parts, respectively. For instance, Grad-CAM generates an attribution map of the form:

$$I(x; w)_{\text{Grad-CAM}} := \max\Big(\sum_k \alpha_k^y A^k(x), 0\Big).$$

Here, α_k^y is the average of gradients of the output $f(x; w)_y$ of the class y w.r.t the activations in the k-th channel, $\alpha_k^y = \frac{1}{Z}\sum_{i,j} \frac{\partial f(x;w)_y}{\partial A^k_{ij}(x)}$, where $Z = \sum_{i,j} 1$ and $A^k_{ij}(x)$ is the (i,j)-th value of the activation map $A^k(x)$. In this case, the bottom part and the top part are defined as $h(x; w_B) = A(x)$ and $g(A(x); w_T) = f(x; w)$ respectively as shown in the figure. This can be generalized to the other types of interpreter. In the case of Gradient $\odot$ Input, however, the back-propagation of the top part of the model reaches to the input layer, which indicates $g(x; w_T) = f(x; w)$ and $h(x; w_B) = x$. For Relevance-CAM, $g(\cdot; w_T)$ and $h(\cdot; w_B)$ are defined in same way with Grad-CAM since the only difference is how α_k^y is computed.

To misdirect the gradient information, we orthogonalize the direction of the gradient between the victim model $\nabla_w f(\cdot; w)$ and the misdirection model $\nabla_{\widetilde{w}} \widetilde{f}(\cdot; \widetilde{w})$. Here, our conjecture is that the degree of freedom to misdirect the gradient of each part will be different considering the quality of the output, from our experience that attribution maps tend to be more sensitive to the activation than the top-part gradient. Therefore, we consider the orthogonalization of the gradient of each part seperately so that the defender can control the amount of misdirection. Formally, for a given query input x_q, DD generates the misdirection model $\widetilde{f}$, where the gradients w.r.t the weight parameters of each part $\widetilde{w}_B$ and $\widetilde{w}_T$ are orthogonalized with those of the original victim model as follows:

$$\begin{aligned} \nabla_{\widetilde{w}_B} \widetilde{f}(x_q; \widetilde{w})_y &\perp \nabla_{w_B} f(x_q; w)_y, \\ \nabla_{\widetilde{w}_T} \widetilde{f}(x_q; \widetilde{w})_y &\perp \nabla_{w_T} f(x_q; w)_y. \end{aligned} \tag{2}$$

Functionality preservation To guarantee the functionality of the misdirection model for each query input x_q, DD tries to match the order of top-k softmax probabilities of the misdirection model to those of the victim model: this can be described as the constraints:

$$\widetilde{f}(x_q; \widetilde{w})_{s_1} \geq \cdots \geq \widetilde{f}(x_q; \widetilde{w})_{s_k} \geq \max_{j \in S'} \widetilde{f}(x_q; \widetilde{w})_j. \tag{3}$$

Here, $S' := \{1, \ldots, K\} \setminus \{s_1 \ldots s_k\}$ and s_i is the index of i-th largest softmax probability of the victim model and K is the number of entire classes.

Interpretability preservation For a given input x_q, the values in the attribution map $I(x_q; w)$ represent the relative importance of input features. Therefore, we try to preserve the information by matching the the entire order of attribution values of the misdirection model to that of the victim model:

$$\widetilde{I}(x_q; \widetilde{w})_{a_1} \geq \widetilde{I}(x_q; \widetilde{w})_{a_2} \geq \cdots \geq \widetilde{I}(x_q; \widetilde{w})_{a_{H \times W}}. \tag{4}$$

Here a_i indicates the index of i-th largest value in the attribution map $I(\cdot; w)$ from the victim model.

4.2 Optimization Problem

To generate the misdirection model $\widetilde{f}(\cdot; \widetilde{w})$ that satisfies (2), we define the orthogonalization loss $\mathcal{L}_{\text{orth}}$ for a query x_q as follows:

$$\begin{aligned} \mathcal{L}_{\text{orth}}(x_q, \widetilde{w}) := \alpha \Big|\cos\angle(\nabla_{w_B} f(x_q; w)_y, \nabla_{\widetilde{w}_B} \widetilde{f}(x_q; \widetilde{w})_y)\Big| \\ + (1-\alpha)\Big|\cos\angle(\nabla_{w_T} f(x_q; w)_y, \nabla_{\widetilde{w}_T} \widetilde{f}(x_q; \widetilde{w})_y)\Big| \end{aligned} \tag{5}$$

Here, $\cos\angle(a,b)$ is the cosine angle of two vectors a and b, and $\alpha \in [0,1]$ is a hyperparameter to balance the degree of orthogonalization of gradients in the top and the bottom parts. However, optimizing $\mathcal{L}_{\text{orth}}$ subject to the constraints in (3) and (4) can be difficult due the non-linearity in the constraints.

Reformulation to an unconstrained optimization problem We reformulate the optimization into an unconstrained optimization using penalty functions to facilitate optimization.

For the functionality preservation constraints (3), we define a penalty function $\mathcal{L}_{\text{pred}}$ such that $\mathcal{L}_{\text{pred}}(x_q;\widetilde{w}) = 0$ if the top k order of $\widetilde{f}(x_q;\widetilde{w})$ is preserved and $\mathcal{L}_{\text{pred}}(x_q;\widetilde{w}) > 0$ otherwise:

$$\begin{aligned}\mathcal{L}_{\text{pred}}(x_q,\widetilde{w}) := &\sum_{i=1}^{k-1}(\widetilde{f}(x_q;\widetilde{w})_{s_{i+1}} - \widetilde{f}(x_q;\widetilde{w})_{s_i})^+ \\ &+ (\max_{j\in S'}\widetilde{f}(x_q;\widetilde{w})_j - \widetilde{f}(x_q;\widetilde{w})_{s_k})^+.\end{aligned} \tag{6}$$

Here, $(x)^+ := \max(0,x)$. For the interpretation preservation constraints (4), we define $\mathcal{L}_{\text{int}}$ as follows:

$$\mathcal{L}_{\text{int}}(x_q,\widetilde{w}) := \sum_{i=1}^{H\times W-1}\mathcal{L}_\delta\Big((\widetilde{I}(x_q;\widetilde{w})_{a_{i+1}} - \widetilde{I}(x_q;\widetilde{w})_{a_i})^+\Big). \tag{7}$$

Here, $\mathcal{L}_\delta$ is the Huber loss [Huber, 1964] defined as follows:

$$\mathcal{L}_\delta(a) := \begin{cases}\frac{1}{2}a^2 & \text{if } |a|\le\delta \\ \delta(|a| - \frac{1}{2}\delta) & \text{otherwise.}\end{cases}$$

The Huber loss here is to prevent relatively large attribution values from being excessively dominant (we use $\delta = 0.3$ in our experiments).

Using the reformations (6) and (7) of the constraints, we define the loss function of DeepDefense as follows:

$$\begin{aligned}\mathcal{L}_{\text{DD}}(x_q,\widetilde{w}) := &\mathcal{L}_{\text{orth}}(x_q,\widetilde{w}) \\ &+ \lambda_1\mathcal{L}_{\text{pred}}(x_q,\widetilde{w}) + \lambda_2\mathcal{L}_{\text{int}}(x_q,\widetilde{w}).\end{aligned} \tag{8}$$

Here $\lambda_1 > 0$ and $\lambda_2 > 0$ are hyperparameters. For each query input x_q, we initialize $\widetilde{f}(\cdot;\widetilde{w})$ with the original victim model $f(\cdot;w)$ then minimize $\mathcal{L}_{DD}(x_q;\widetilde{w})$ by iteratively updating $\widetilde{w}$ with a gradient-descent algorithm.

Selection of sensitive layers Involving entire layers of victim model in (5) can be costly since the optimization of (8) will include a large number of variables for deep neural nets. In the spirit that each layer may have different sensitivity to the prediction outcome, we define the sensitivity score S_ℓ of the layer ℓ:

$$S_\ell := \frac{1}{N}\sum_{i=1}^{N}\|\nabla_{w_\ell} f(x_i;w)_{y_i}\|_1.$$

Here, w_ℓ is a sub-vector of weight parameters corresponding to the ℓ-th layer and y_i is the predicted class of the input x_i. N is the total number of data points involved in measuring S_ℓ. To choose a reasonable number of layers ℓ to include in optimization, we use the cumulative sensitivity ratio which is defined as

$$\text{CS}(\ell) := \frac{\sum_{i=1}^{\ell} S_{(i)}}{\sum_{i=1}^{L} S_{(i)}} \times 100\ (\%), \tag{9}$$

Dataset	Model	f Test Acc (%)
MNIST	LeNet	99.49
KMNIST	LeNet	99.49
CIFAR-10	WRN16-4	95.21
Flowers-17	ResNet-18	95.96
CUBS-200	ResNet-34	73.94

Table 1: Datasets, model architecture and the baseline test accuracy of the victim models.

where $S_{(1)} \ge S_{(2)} \ge \cdots \ge S_{(L)}$ are the layer sensitivity scores ranked in decreasing order and L is the number of the candidate layers. In our experiments, we use the smallest ℓ for which the CS(ℓ) is at least 90%.

5 Experiments

Datasets, models and interpreters To train the victim models, we used five image classification datasets: MNIST , KMNIST , CIFAR-10 , Flowers-17 , and CUBS-200 . The victim model's architecture and the test accuracy for each dataset are described in Table 1. We used three popular interpreters: Grad-CAM [Selvaraju *et al.*, 2017], Relevance-CAM (denoted by Rel-CAM) [Lee *et al.*, 2021] and Gradient $\odot$ Input. For Grad-CAM and Rel-CAM , we obtained the activation maps from the penultimate layer of each model.

Attack strategy As mentioned in Section 3, we assume a model stealing adversary who can use both the softmax outputs and the attribution maps, training an attack model minimizing the loss in (1). Since we assume that the adversary cannot access to the victim's data distribution P, the adversary chooses the transfer-set from P_A which is likely to be different from P in the real-world. This is a similar setup to [Orekondy *et al.*, 2019]. We used Fashion-MNIST, MNIST, CIFAR100 as the transfer-sets for MNIST, KMNIST, and CIFAR10, and ImageNet as the transfer-set for Flowers-17 and CUBS-200. We set the adversary's query budget to $B = 10000$ unless stated otherwise. For each attack scenario, we set the adversary to have the same neural net architecture and the interpreter to those of the victim which is one of the worst-case setups for the defender.

Defense performance To demonstrate the effectiveness of DeepDefense (DD), we evaluate the defense performance on four values of query budget B, 2500, 5000, 7500, and 10000. There is no other model stealing defense method considering the adversary that exploits both the victim's softmax probabilities and attribution maps to our best knowledge. Therefore, we compare the defense performance of DD with state-of-the-art defense methods which consider the adversary that exploits the softmax output only, Prediction Poisoning (PP), Adaptive Misinformation (AM), and Ensemble of Diverse Models (EDM). For a fair comparison, we carefully adjusted the hyperparameters of the methods so that the defense will be done without losing the test accuracy of the victim models. For DD, we set $k = 1$ to preserve the index of maximum softmax probability.

As shown in Figure 2, DD outperformed the other defense methods on the entire datasets and interpreters. Noticeably,

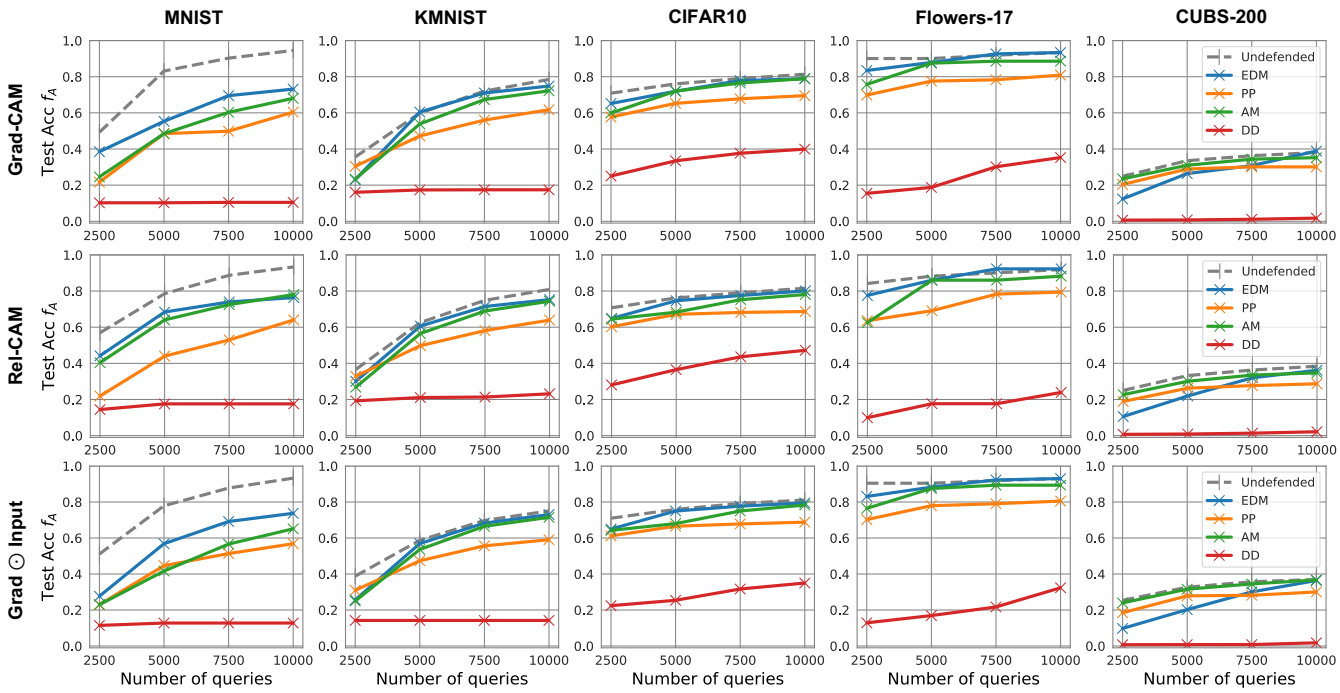

Figure 2: Defense performance comparison of undefended models, PP, AM, EDM and our DeepDefense (DD) in test accuracy of attack models f_A (lower test accuracy of f_A implies better defense).

DD showed consistent defense performance regardless of the query budget. In contrast, AM or EDM allowed the test accuracy of the attack model to be close to that of the undefended cases, especially on the large query budgets. We think that such consistency of DD is due to its property that it tries to leak minimal information at every query point without imposing any specific assumption about the transfer-set.

Interpretability preservation To demonstrate how well DD preserves the interpretability of attribution maps, we used a popular attribution quality measure Average Drop [Chattopadhay *et al.*, 2018] which is defined as $\frac{1}{N}\sum_{i=1}^{N}(y_i^c - \tilde{y}_i^c)^+/y_i^c$. Here y_i^c is the prediction score for class c on the input x_i and $\tilde{y}_i^c$ is the prediction score on the masked input where the pixel values are set to 0 except for the top $p\%$ attribution region of $I(x_q)$. Following the quality evaluation in Grad-CAM [Selvaraju *et al.*, 2017], we set $p = 15$. We use this measure to check quality drops in attribution maps produced by DD.

As shown in Table 2, DD allowed only a small degradation in attribution quality in terms of Average Drop. In particular, there was no statistically significant difference in the measurements between the attribution maps from the victim and the misdirection models (p-value from t-test : 0.7581). We also provide the qualitative comparison of the attribution maps from the victim and the misdirection models in Figure 3. The highlighted regions in the attribution maps of the misdirection model are almost identical to those of the victim model.

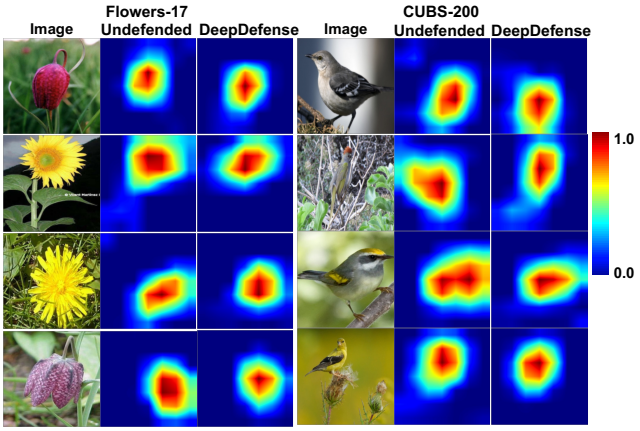

Figure 3: Visual quality comparison of attribution maps from the victim model (the second and the fourth columns) and the misdirection model (the third and the last columns).

Impact of layer selection on computation cost We evaluated the effectiveness of our layer selection strategy in terms of the defense performance and the run-time with the Flowers-17 dataset. Here, we used Rel-CAM as the interpreter and obtained the activation maps from the three different locations of ResNet-18: the 9-th, the 13-th, and the 17-th (penultimate) layers. The measurements are conducted on a Linux machine with an NVIDIA RTX 2080 Ti GPU. Table 3 shows the defense performance and the run-time for processing single query output with various cumulative sensitivity

Dataset	Grad-CAM		Rel-CAM		Grad $\odot$ Input	
	Avg Drop I	Avg Drop $\widetilde{I}$	Avg Drop I	Avg Drop $\widetilde{I}$	Avg Drop I	Avg Drop $\widetilde{I}$
MNIST	0.7888±0.3691	0.7587±0.4091	0.5621±0.4980	0.5425±0.5019	0.5670±0.4020	0.5613±0.4024
KMNIST	0.7135±0.2834	0.6889±0.3169	0.7516±0.2909	0.7260±0.3962	0.5815±0.3591	0.6067±0.3499
CIFAR-10	0.7622±0.3558	0.7753±0.3779	0.7365±0.3882	0.7042±0.3761	0.8647±0.2859	0.8588±0.2869
Flowers-17	0.5130±0.3018	0.5152±0.3078	0.5033±0.3140	0.5046±0.3140	0.8287±0.2249	0.8256±0.2304
CUBS-200	0.5593±0.4002	0.5747±0.4181	0.5649±0.4027	0.5934±0.4260	0.9581±0.1493	0.9647±0.1307

Table 2: Interpretability comparison of attribution maps from original victim model (I) and DD ($\widetilde{I}$) in Average Drop ($\pm$ std, lower is better).

ℓ	CS (%)	# layers	f_A Test Acc (%)		Run time (sec)	
			PP	DD	PP	DD
9	90	8	60.66	8.82	0.23	1.53
	70	4		11.76		1.04
	50	2		10.29		0.65
13	90	7	61.76	8.09	0.21	1.41
	70	4		8.82		0.97
	50	2		10.29		0.60
17	90	8	62.13	9.19	0.21	1.47
	70	6		8.98		1.38
	50	3		11.40		0.73

Table 3: Defense performance and run time of DeepDefense on the different values of cumulative sensitivity ratio (CS) described in (9). ℓ in the first column indicates the index of the layer where activation maps for the interpretation are obtained.

Dataset	method	top-k	f_A Test Acc (%)
Flowers-17	PP	1	62.13
	DD	1	9.19
		3	41.91
		5	51.84
CUBS-200	PP	1	18.61
	DD	1	0.64
		3	1.10
		5	2.54

Table 4: Defenese performance of DeepDefense on the difference values of k.

ratios (CS) defined as (9).

DD showed consistent defense performance on the change of CS. For instance, the average test accuracy of the attack model on DD with CS of 50% is only 1.96% higher than that on DD with CS of 90%. DD with CS of 50% showed a runtime competent to PP in terms of time cost. Those results imply that our layer selection strategy effectively reduces computation cost without hurting the defense performance of DD too much.

Effect of top-k in functionality preservation We investigate the effect of k in (6) on the defense performance of DD. We evaluate the defense performance of DD on the Flowers-17 and CUBS-200 datasets with Grad-CAM . As shown in Table 4, DD showed better defense performance than PP for all of the three values of k on both datasets. However, larger k resulted in more considerable degradation of defense performance for DD, where the degree of degradation may vary. The result indicates that a proper amount of preservation on functionality should be gauged to guarantee the high defense performance of DD.

Dataset	f_A Test Acc (%)				
	Undef	PP	DD	DD-S	DD-I
MNIST	93.37	64.03	17.56	26.07	82.16
CIFAR-10	81.57	68.68	47.21	55.52	75.50
Flowers-17	91.91	79.41	23.90	41.91	90.44

Table 5: Ablation study. Undef: undefended case, DD-S: applied DD on softmax only, DD-I: applied DD on interpretation only.

Ablation study To study the contribution of the probability and the interpretation output of the misdirection model to the defense by DD, we conducted an ablation study with two variants of DD: (i) DD-S that perturbed the softmax outputs only; (ii) DD-I that perturbed the attribution maps only.

As shown in Table 5, DD-S showed better defense performance than PP. This implies that our approach is competitive to PP even for the threat model that does not consider the attribution maps of the victim model. On the other hand, DD-I allowed the adversary to achieve relatively high test accuracy, even though they were lower than the performance of the undefended models. The results indicate that considering the attribution map alone may not be sufficient to achieve a high defense performance for the adversary who exploits both softmax outputs and attribution maps.

6 Conclusion

We proposed DeepDefense (DD), a novel defense method that protects AI models against model stealing adversaries who can use both prediction probabilities and attribution maps. In DD, only the outputs of the misdirection model are revealed to the users, in which misdirected gradient information is encoded to prevent adversaries from training attack models efficiently. Also, DD preserves probability and attribution outcomes at a similar quality to the original victim model. To find the misdirection model, we formulate an unconstrained optimization problem that can be solved efficiently with simple gradient-descent algorithms. Our experiments showed that DD outperforms the existing model stealing defenses in multiple scenarios with a reasonable cost.

Acknowledgements

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2018R1D1A1B07051383), and by the MSIT (Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program (IITP-2020-0-01749) supervised by the IITP (Institute of Information & Communications Technology Planning & Evaluation).

References

[Bach *et al.*, 2015] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10(7):1–46, 07 2015.

[Chattopadhay *et al.*, 2018] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth Balasubramanian. Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks. In *WACV*, 2018.

[Chouldechova and Roth, 2020] Alexandra Chouldechova and Aaron Roth. A snapshot of the frontiers of fairness in machine learning. *Commun. ACM*, 63(5):82–89, April 2020.

[GDPR, 2016] GDPR. EU General Data Protection Regulation: Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/EC (general data protection regulation), OJ 2016 L 119/1. European Commission, 2016.

[Gu *et al.*, 2018] Jindong Gu, Yinchong Yang, and Volker Tresp. Understanding individual decisions of cnns via contrastive backpropagation. In *ACCV*, 2018.

[Gunning, 2016] David Gunning. Explainable artificial intelligence (XAI), DARPA-BAA-16-53. Defense Advanced Research Projects Agency, August 2016.

[Gur *et al.*, 2021] Shir Gur, Ameen Ali, and Lior Wolf. Visualization of supervised and self-supervised neural networks via attribution guided factorization. In *AAAI*, 2021.

[Huber, 1964] Peter Huber. Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics*, 35(1):73 – 101, 1964.

[Kapishnikov *et al.*, 2019] Andrei Kapishnikov, Tolga Bolukbasi, Fernanda Viégas, and Michael Terry. XRAI: Better attributions through regions. In *ICCV*, 2019.

[Kariyappa and Qureshi, 2020] Sanjay Kariyappa and Moinuddin Qureshi. Defending against model stealing attacks with adaptive misinformation. In *CVPR*, 2020.

[Kariyappa *et al.*, 2021] Sanjay Kariyappa, Atul Prakash, and Moinuddin Qureshi. Protecting dnns from theft using an ensemble of diverse models. In *ICLR*, 2021.

[Lee *et al.*, 2021] Jeong Ryong Lee, Sewon Kim, Inyong Park, Taejoon Eo, and Dosik Hwang. Relevance-CAM: Your model already knows where to look. In *CVPR*, 2021.

[Lundberg and Lee, 2017] Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Neurips*, 2017.

[Milli *et al.*, 2019] Smitha Milli, Ludwig Schmidt, Anca Dragan, and Moritz Hardt. Model reconstruction from model explanations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 2019.

[Montavon *et al.*, 2018] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018.

[Nam *et al.*, 2020] Woo-Jeoung Nam, Jaesik Choi, and Seong-Whan Lee. Relative attributing propagation: Interpreting the comparative contributions of individual units in deep neural networks. In *AAAI*, 2020.

[Orekondy *et al.*, 2019] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Knockoff nets: Stealing functionality of black-box models. In *CVPR*, 2019.

[Orekondy *et al.*, 2020] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Prediction poisoning: Towards defenses against dnn model stealing attacks. In *ICLR*, 2020.

[Pal *et al.*, 2020] Soham Pal, Yash Gupta, Aditya Shukla, Aditya Kanade, Shirish Shevade, and Vinod Ganapathy. Activethief: Model extraction using active learning and unannotated public data. *AAAI*, 2020.

[Papernot *et al.*, 2017] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *ACM on ASIACCS*, 2017.

[Petsiuk *et al.*, 2018] Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. In *BMVC*, 2018.

[Sattarzadeh *et al.*, 2021] Sam Sattarzadeh, Mahesh Sudhakar, Anthony Lem, Shervin Mehryar, Konstantinos Plataniotis, Jongseong Jang, Hyunwoo Kim, Yeonjeong Jeong, Sangmin Lee, and Kyunghoon Bae. Explaining convolutional neural networks through attribution-based input sampling and block-wise feature aggregation. In *AAAI*, 2021.

[Selvaraju *et al.*, 2017] Ramprasaath Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017.

[Shrikumar *et al.*, 2017] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *ICML*, 2017.

[Stoica *et al.*, 2017] Ion Stoica, Dawn Song, Raluca Ada Popa, David Patterson, Michael Mahoney, Randy Katz, Anthony Joseph, Michael Jordan, Joseph Hellerstein, Joseph Gonzalez, Ken Goldberg, Ali Ghodsi, David Culler, and Pieter Abbeel. A berkeley view of systems challenges for AI. Technical Report UCB/EECS-2017-159, University of California, Berkeley, 2017.

[Tramèr *et al.*, 2016] Florian Tramèr, Fan Zhang, Ari Juels, Michael Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction APIs. In *USENIX Security Symposium*, 2016.

[Zhang *et al.*, 2021] Qinglong Zhang, Lu Rao, and Yubin Yang. A novel visual interpretability for deep neural networks by optimizing activation maps with perturbation. In *AAAI*, 2021.

[Zhou *et al.*, 2016] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *CVPR*, 2016.