Fully Exploiting Every Real Sample: SuperPixel Sample Gradient Model Stealing

Yunlong Zhao[1] Xiaoheng Deng[1] * Yijing Liu[2] Xinjun Pei[1] Jiazhi Xia[1] Wei Chen[2]
[1]Central South University [2]Zhejiang University
{zhaoyl741, dxh, pei_xinjun, xiajiazhi}@csu.edu.cn,{liuyj86, chenvis}@zju.edu.cn

Abstract

Model stealing (MS) involves querying and observing the output of a machine learning model to steal its capabilities. The quality of queried data is crucial, yet obtaining a large amount of real data for MS is often challenging. Recent works have reduced reliance on real data by using generative models. However, when high-dimensional query data is required, these methods are impractical due to the high costs of querying and the risk of model collapse. In this work, we propose using sample gradients (SG) to enhance the utility of each real sample, as SG provides crucial guidance on the decision boundaries of the victim model. However, utilizing SG in the model stealing scenario faces two challenges: 1. Pixel-level gradient estimation requires extensive query volume and is susceptible to defenses. 2. The estimation of sample gradients has a significant variance. This paper proposes Superpixel Sample Gradient stealing (SPSG) for model stealing under the constraint of limited real samples. With the basic idea of imitating the victim model's low-variance patch-level gradients instead of pixel-level gradients, SPSG achieves efficient sample gradient estimation through two steps. First, we perform patch-wise perturbations on query images to estimate the average gradient in different regions of the image. Then, we filter the gradients through a threshold strategy to reduce variance. Exhaustive experiments demonstrate that, with the same number of real samples, SPSG achieves accuracy, agreements, and adversarial success rate significantly surpassing the current state-of-the-art MS methods. Codes are available at https://github.com/zyl123456aB/SPSG_attack.

1. Introduction

Machine Learning as a Service (MLaaS), enhances efficiency in both work and daily life [17, 30, 32, 53]. These invaluable MLaaS models have become targets for malicious users to steal. Model Stealing (MS) [3, 4, 27, 28, 33, 40, 44–46, 48, 51, 52] involves constructing a proxy model similar to the victim model by acquiring query results of input samples. Beyond the direct utilization of the proxy model, malicious users can also generate a series of attacks based on the proxy model and transfer them to the victim model, including membership inference attacks [36], adversarial attacks [54], and model inversion attacks [11], etc.

The basic paradigm of MS is to construct a sample attack set and train a proxy model through the samples in the attack set and the corresponding query results of the victim model. Data-free MS is based on generative networks and uses noise to synthesize artificial images for the training of the proxy model. Although data-free MS claims that real samples are not needed, in practical applications that require high-dimension samples, data-free MS still has the inevitable real sample demand. As shown in Table 1, on the one hand, high-dimensional inputs would significantly increase the query volume for data-free MS. For color images of 224x224 pixels, the query volume for data-free MS could reach tens of millions. On the other hand, due to the inherent risk of model collapse in GANs [14], training proxy models with data-free MS is prone to failure. By giving data-free MS a small amount (10k to 20k) of real samples related to the domain of the victim model as image priors, data-free MS can reduce the query cost and model collapse risk and improve the stealing effect. However, acquiring high-quality real samples that meet specific MLaaS requirements is challenging. Different MLaaS have varying requirements for input images, and blurred input images can distort MLaaS query results. Privacy and copyright protections further complicate the acquisition of high-quality real samples. Additionally, even with a plethora of real samples, the marginal benefit of each real sample for improving the proxy model diminishes as the cardinality of real samples increases. Therefore, expensive real samples should be fully utilized.

To make full use of each real sample, our idea is to obtain more information about the model from real samples. However, for black-box models, it is challenging to obtain other types of information to train proxy models. For example, all

*Corresponding author is Xiaoheng Deng

Table 1. Results of data-free MS without using real samples and with using 10k domain-relevant real samples. Results are queries, real samples, the failure times, and test accuracy (in %), of each method with querying probability. The failure times are determined by the number of model collapses observed over 10 training runs. we report the other average result computed over 10 runs. The training strategies and experimental configurations used are described in the experimental section 4. (1k=1000)

Data-free MS(probability)	CUBS200(0)			Indoor67(0)			CUBS200(10k)			Indoor67(10k)		
	queries	failures	accuracy	queries	failures	accuracy	queries	failures	accuracy	queries	failures	accuracy
ZSDB3KD [46]	5183k	3	48.72	5771k	4	51.77	1202k↓	2↓	51.47↑	1112k↓	2↓	59.41↑
DFMS [33]	4987k	2	51.28	4771k	2	55.21	1021k↓	1↓	55.21↑	1009k↓	0↓	61.11↑
EDFBA [52]	1623k	2	51.46	1792k	1	56.27	459k↓	0↓	55.61↑	349k↓	0↓	61.17↑
DS [4]	4310k	3	50.26	4291k	2	54.21	1077k↓	1↓	54.97↑	1021k↓	1↓	60.27↑
DFME [40]	4102k	2	48.82	4671k	2	52.78	1489k↓	1↓	52.31↑	1339k↓	1↓	58.71↑

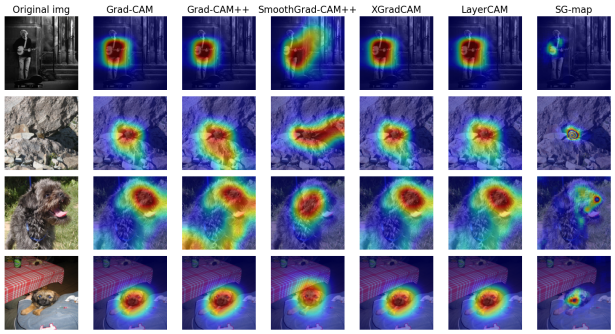


Figure 1. The columns from left to right are grad-CAM [34], grad-CAM++ [6], Smooth-gradCAM [26], X-gradCAM [12], layer-CAM [20], and SG-map. The neural network is ResNet34 pre-trained on ILSVRC-2012.

feature maps from the intermediate layers of the black-box model [19, 22, 29, 49] are unknowable. Our focus shifts to the input of the model, specifically, the Sample Gradient (SG) backpropagated from the model's output layer to the input sample. SG is used to assist in generating the interpretability heat map of the model in many interpretability works. Actually, SG itself contains a lot of model interpretability decision information. As shown in Figure 1, we compare the heat map generated by average pooling SG with other CAM methods, and we can see that SG fully reflects the model's decision. However, the obstacles for imitating sample gradients in MS include: (1) The most primitive method to obtain sample gradients involves perturbing each pixel individually and acquiring query results for all perturbed images, a process with enormous query costs. Specially, obtaining sample gradients for a single image requires over a hundred thousand queries. Furthermore, inputting perturbed images at the pixel level into the victim model is akin to inputting adversarial images, which could be easily detected and thwarted by defense mechanisms like Prada [21]. (2) Sample gradients have significant variance due to certain specific or redundant neurons backpropagating. Then, we introduce a novel SuperPixel [1, 9, 42] Sample Gradient Model stealing (SPSG) to solve the issues. SPSG comprises two modules: superpixel gradient querying (SPGQ) and sample gradient purification (SGP). For issue (1), SPGQ module first segments the image into multiple superpixels based on a segmentation algorithm. Then, it applies perturbations to these superpixels and queries the output to obtain the sample superpixel gradients. For issue (2), SGP module eliminates significant variance from the sample gradients by filtering extremum information and normalizing, ensuring the extraction of clean and useful gradient information. Then, purified superpixel gradients are associated with the pixel gradients of the proxy model to train proxy models.

Our contributions are enumerated as follows:

- We design SPSG to extract the maximum amount of available model information from each real sample. The superpixel querying module significantly reduces the query volume required to acquire the tacit knowledge in one sample gradient from 10^6 to 10^2 while simultaneously evading defenses like Prada [21]. Meanwhile, the gradient purification module effectively removes noise from the sample gradients. The effectiveness of the gradient purification module is further validated through ablation experiments.
- Through various experiments, SPSG enables the proxy model to achieve accuracy, agreement, and attack success rate substantially surpassing state-of-the-art algorithms with the same number of real samples. Specifically, when stealing a resnet34 model trained on CUBS-200 using 20,000 real samples, SPSG achieves an accuracy of 61.21% and an agreement of 67.48%, significantly outperforming the second-best method with an accuracy of 56.39% and agreement of 58.44%. To match the accuracy achieved by SPSG with 10,000 real samples, the second-best method requires at least 20,000 real samples.

2. Related work

2.1. Sample Gradient

Sample gradients, obtained through the backpropagation of a model's final loss function, depend on the parameters and structure of the neural network. They are primarily used in adversarial training and model interpretability.

Adversarial Training Based on Sample Gradients. By

introducing small perturbations along the direction of sample gradients, new samples capable of deceiving the model can be generated. Techniques like FGSM [24] are sensitive to the sign of sample gradients, whereas FGM [15] normalizes the gradients. PGD [5] and FreeAT [35] implement multiple iterations with smaller step sizes on sample gradients, keeping the perturbations within a specified range. YOPO [50] reduces gradient calculation costs by leveraging the network's structure, with perturbations related only to the first layer. FreeLB [55] accumulates gradients during training, giving perturbations a more directional tendency.

Model Interpretability Based on Sample Gradients. Techniques like "Saliency Map" [37] create saliency maps by calculating gradients of input images to highlight the model's focus areas in image classification tasks. Guided Backpropagation [39] helps understand the decision-making process and the features learned by each convolutional layer. GRAD-CAM [34] and SmoothGrad [38], while not directly using sample gradients, generate heatmaps using feature map gradients to show the focused image areas. [10] proposes a method for explaining black-box models through input perturbations, generating interpretability masks to illustrate the model's focus areas.

2.2. Superpixel Segmentation

Superpixels were introduced to create image over-segmentation based on similarity criteria. Algorithms like SLIC [2] efficiently generate superpixels by clustering pixels in a five-dimensional color and image plane space. Subsequent methods [9, 42] have improved superpixel segmentation.

2.3. Model Stealing

Data-Free Model Stealing. Data-free model stealing techniques [3, 4, 33, 40, 44, 46, 51, 52] do not require any original training data. Attackers generate synthetic queries, often through prior knowledge or assumptions about the data distribution, to probe the model and reconstruct its functionality. All data-free model stealing (MS) inevitably draws on the concept of Generative Adversarial Networks (GANs). Therefore, there is an unavoidable risk of model collapse. Given that the querying cost required for a single instance of model theft is quite high, a collapse during the extraction process would further increase the querying costs. Additionally, since the training and querying phases in data-free MS are coupled, each training of a proxy model requires a new round of queries, which also increases the query volume.

Data-Driven Model Stealing. Data-driven model stealing attacks [27, 28, 45], utilize real data, allowing for all attack set samples to be queried before training the proxy model. Therefore, it is not necessary to query the victim model again with each training of a new proxy model. The real samples can be domain-irrelevant to the victim model. Although domain-relevant real samples can achieve certain improvements in the effectiveness of the theft, domain-irrelevant real samples are more commonly used. Data-driven MS can also further refine the selection of real samples based on information from the training process of the proxy model to enhance the stealing effect.

3. SuperPixel Sample Gradient Model Stealing

3.1. Overview

In the fundamental paradigm of offline Model Stealing (MS), MS begins with the construction of a query set comprising all input samples and their corresponding query results. Subsequently, this pre-assembled query set is utilized to train different proxy models. SPSG, falling under the category of offline MS, mainly encompasses two distinct modules: SuperPixel Gradient Query (SPGQ) and Sample Gradient Purification (SGP), as shown in Figure 2. SPGQ is designed for the assembly of the query set. In addition to acquiring the predictive probabilities or hard labels for each sample's output, SPGQ is also adept at obtaining superpixel gradients for each sample at a low query cost, while simultaneously circumventing adversarial attack monitoring. On the other hand, SGP is employed for the training of the proxy model. Within this module, the superpixel sample gradients from the query set undergo a denoising process. This process ensures the retention of the extremal portions of the superpixel gradients across each channel of the image. Based on every superpixel range filtered by the victim model, the pixel gradients of the proxy model are averaged to obtain the simulated superpixel gradients. In the final stage, we introduce a novel loss function that establishes a connection between simulated superpixel gradients and their ground-truth. Through this connection, the proxy model is effectively trained, culminating in a comprehensive and robust offline MS framework.

3.2. SuperPixel Gradient Query

For black-box models, the finite difference method for calculating pixel gradients of input images is a primitive yet effective approach. The finite difference method estimates gradients by applying a small perturbation to the input sample and observing the resultant output changes. Finite difference includes forward difference, central difference, and backward difference. In this paper, we default to using the forward difference. Specifically, for an input sample x and a small perturbation ε, an approximation of the gradient for each pixel i in channel $c = 1 \vee 2 \vee 3$ can be calculated using the following formula:

$$q_i^c = \frac{\partial f}{\partial x} \approx \frac{f(x + \varepsilon \cdot e_i^c) - f(x)}{\varepsilon} \tag{1}$$

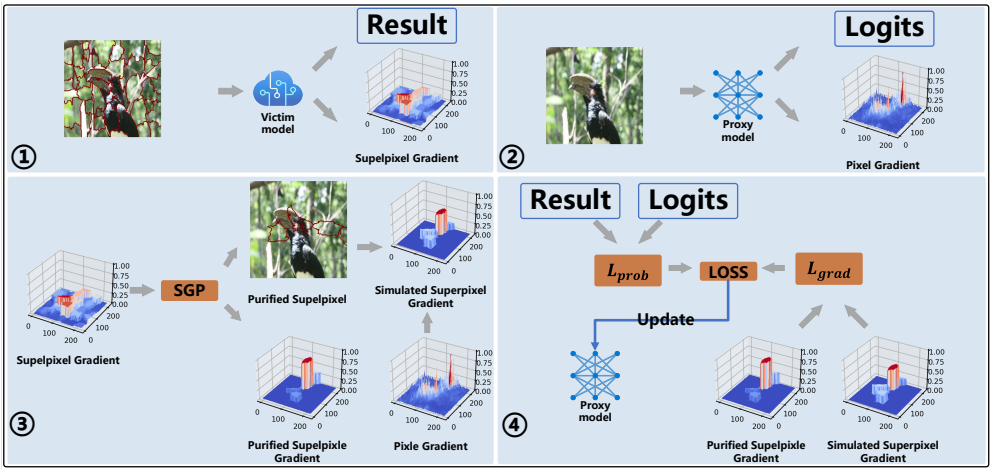


Figure 2. Four steps of SPSG. The first step is to obtain superpixel gradients and query results through SGPQ. The second step involves acquiring pixel gradients and output logits of the proxy model through backpropagation on the input sample. The third step is to obtain purified superpixel gradients and simulated superpixel gradients of the proxy model using SGP. The fourth step involves updating the proxy model based on the loss function. The gray arrow represents the direction from input to output.

Here, ε must be sufficiently small to capture the function's variations at x, yet not too small to avoid numerical precision issues. In this paper, the value of ε is set to $1e-5$. e_i is a standard basis vector with only the i-th pixel in channel c as 1. However, pixel-level forward differences require one perturbed query per input dimension, rendering this method computationally expensive in high-dimensional input spaces. In addition, MLaaS can track and store a series of pixel-level perturbed images of the input. Under normal circumstances, the pairwise distance between queried images with the same label follows a Gaussian distribution. However, with the addition of tiny perturbations, the pairwise distance no longer follows a Gaussian distribution and tends to an extreme distribution. Prada [21] can detect such changes in distribution and outputs a noisy prediction result upon detection of change.

Therefore, we propose SuperPixel Gradient Query (SPGQ), whose core idea is to extend the pixel-level forward difference to the superpixel level. Superpixels are a concept in image processing and computer vision, referring to the technique of combining similar pixels into a unified region or cluster. Each superpixel $P_j = \bigcup_i^{N_j} p_i$ contains adjacent pixels p_i of N_j number, similar in color, brightness, texture, or other attributes. SPGQ performs forward differences on three different channels of each superpixel independently. For all pixels within the same channel of each superpixel, the same disturbance is added: $E_j^c = \sum_i^{N_j} e_i^c$. The victim model is queried to calculate the difference and obtain an approximate gradient:

$$g_j^c = \frac{\partial f}{\partial P_j^c} \approx \frac{f\left(x + \varepsilon E_j^c\right) - f\left(x\right)}{\varepsilon} \tag{2}$$

Since a superpixel typically contains 900 or more pixels within a channel, perturbing all these pixels will not produce an extremely small pairwise distance, which would not affect the Gaussian distribution of the queried images. Thus, superpixel queries can circumvent Prada attacks.

3.3. Sample Gradient Purification

Sample gradients have a significant variance, and numerical discrepancies between different models can be inherited through the backpropagation process, manifesting in the sample gradients. superpixel gradient also has the same drawbacks. Hence, we have formulated a Sample Gradient Purification Mechanism to mitigate the interference from variance and other extraneous factors.

Concerning the gradient of every queried superpixel channel $G^c = \bigcup_j^J g_j^c$, we initially perform a denoising operation. The core objective of denoising is to preserve the extreme values of the sample gradients, eliminating non-extreme gradient values, as the extreme portions encapsulate the focal points of the model. Additionally, due to the divergent implications of positive gradients (indicating facilitation of the loss function) and negative gradients (indicating impediment), it becomes imperative to independently execute denoising operations on the sets of positive and negative gradients. We commence by calculating the extreme values of every channel's positive and negative gradients: $g_+^c = \max\left(g_j^c \in \{g_j^c | g_j^c \geq 0\}\right)$ and $g_-^c = \max\left(g_j^c \in \{g_j^c | g_j^c < 0\}\right)$. The filtered gradients are given by:

$$G_+^c = \{g_j^c | g_j^c > \beta g_+^c\} \qquad G_-^c = \{g_j^c | g_j^c < \beta g_-^c\} \tag{3}$$

In Equation 3, β is a predefined hyperparameter within

the range $0 < \beta < 1$, typically set to 0.5 by default. The Discarded gradient set $(G^c)' = \left\{g_j^c | g_j^c \leq \beta g_+^c\right\} \bigcup \left\{g_j^c | g_j^c \geq \beta g_-^c\right\}$, is assigned the value of zero. Simultaneously, to avert inheriting numerical discrepancies from the models and potential exaggeration or diminution of gradient values due to gradient explosion or vanishing, normalization of the filtered sample gradients is requisite. The normalization operation is as follows:

$$G_+^c = \left\{ \frac{g_j^c}{g_+^c} | g_j^c \in G_+^c \right\} \qquad G_-^c = \left\{ \frac{g_j^c}{g_-^c} | g_j^c \in G_-^c \right\} \quad (4)$$

The gradient values closer to 1 signify a higher degree of facilitation or impediment of the model's loss function at that particular pixel position. Following denoising and normalization, we obtain the purified sample gradients.

The role of SGP is crucial. We demonstrate the effectiveness of SGP's purification in the subsequent ablation study Section 4.7.

3.4. Objective Function for Training

After obtaining the superpixel level gradient of an image in the victim model f_v, the proxy model f_s cannot directly mimic and learn the dark knowledge of the superpixel gradient. Firstly, we use a function similar to the one used for training the victim model to calculate the sample pixel gradients of the proxy model. Regardless of whether the query result is hard labels or probabilities, we use a cross-entropy function similar to that used in training the victim model to calculate the sample gradients for the proxy model :

$$f(y, p) = -\log(y_p) \quad (5)$$

For equation 5, y is the proxy model's K-dimensional vector output. p is the victim model's predicted label. Therefore, for the pixel-level gradients $Q = \bigcup_j^J \bigcup_i^{N_j} q_i = \partial f(y,p)/\partial x$ obtained through backpropagation in the white-box proxy model, we adopt a mean coverage method to obtain the same format of the simulated-superpixel gradient. Specifically, based on the superpixel partition of the victim model, we take the mean of all pixel gradients within the same superpixel in the proxy model, replace the original pixel gradients, and obtain the simulated superpixel gradient:

$$Q' = \bigcup_j^J \bigcup_c^3 \left\{ \frac{\sum_i^{N_j} q_i^c}{N_j} | p_i^c \in P_j \right\} \quad (6)$$

Our ultimate objective function or loss function consists of L_{grad} and initial L_{prob}. For hard-label query mode (giving top-1 probability $\hat{y}$ and predicted label p) and probability query mode (giving K-dimensional probability $\hat{y}$), L_{prob} is different:

$$L_{prob}(y, \hat{y}) = \begin{cases} -y_p \bullet \log(\hat{y}) - \log(y_p) & \text{if of hard-label} \\ -\sum_k^K (y_k) \bullet \log(\hat{y}_k) & \text{if of probability} \end{cases} \quad (7)$$

Our gradient loss function L_{grad} is set in two parts: the first part ensures similarity in the gradient values for each superpixel, and the second part ensures similarity in the overall gradient of the sample. For the first part, f_v and f_s having similar gradients for each superpixel is equivalent to the query results of the image with the perturbed superpixel being similar. Define $x_j^c = x + \varepsilon E_j^c$ and $G_{all}^c = G_+^c \bigcup G_-^c$. Then, we can get:

$$L_{grad,1} = \sum\nolimits_j^J \sum\nolimits_c^3 \left(L_{prob}\left(f_s\left(x_j^c\right), f_v\left(x_j^c\right)\right) | g_j^c \in G_{all}^c \right) \quad (8)$$

For the second part, we need to calculate the cosine similarity between G and Q':

$$L_{grad,2} = 1 - \cos(G, Q') \quad (9)$$

Based on $L_{grad} = L_{grad,1} + L_{grad,2}$, the final loss function of the victim model is:

$$L = L_{prob}(f_s(x), f_v(x)) + L_{grad} \quad (10)$$

Through this objective function, the proxy model can effectively mimic the SG knowledge of the victim model. We document the visual changes of the simulated superpixel SG of the proxy model during the training process in Supplementary Material.

4. Experiment

4.1. Experiment Setup

Victim Model. We employ four datasets used in Knockoff for our experimentation: Caltech256 (256 classes) [16], CUB-200-2011 (200 classes) [43], Indoor Scenes (67 classes) [31], and Diabetic Retinopathy (5 classes) [7]. For Diabetic Retinopathy, we strip 200 images from the training set for each category, forming a test set that in total contains 1000 images. A resnet34 [18] model trained on these four datasets serves as our victim model. The training procedure for the victim model mimics that of Knockoff's victim model. Specifically, the model is trained for 200 epochs using an SGD optimizer with a momentum of 0.5 and an initial learning rate of 0.1 that decays by a factor of 0.1 every 60 epochs. The well-trained victim model is available for download at Knockoff Code. The victim model has achieved accuracies of 78.4%, 77.1%, 76.0%, and 59.4% on the datasets mentioned sequentially above.

Baseline and Attack Dataset. Baselines are categorized based on the necessity of real data. Data-free baselines include DFME [40], DS [4], DFMS [33], EDFBA [52], and ZSDB3KD [46], while Knockoff [27], ActiveThief [28], Black-Box Dissector [45], and InverseNet [13] are Data-driven baselines that require real data. Notably, DFMS, EDFBA, ZSDB3KD, DS, Black-Box Dissector, inverseNet, ActiveThief, and Knockoff remain functional even when

Table 2. The agreement (in %), test accuracy (in %), and queries of each method with querying probability or hard label. For our model, we report the average result as well as the standard deviation computed over 10 runs. (**Boldface**: the best value.)

Method (probability)	CUB200 (10k)			CUB200 (15k)			CUB200 (20k)		
	Agreement	Acc	Queries	Agreement	Acc	Queries	Agreement	Acc	Queries
ZSDB3KD	51.47	49.32	1202k	52.31	50.67	1098k	55.33	53.08	1021k
DFMS	55.21	53.19	1021k	57.41	55.36	1003k	58.44	55.98	1007k
EDFBA	55.61	53.12	459k	57.67	55.69	451k	58.12	55.72	451k
DS	54.97	53.98	1077k	57.86	55.71	1021k	57.04	56.39	997k
DFME	52.31	50.17	1489k	53.01	51.27	1344k	55.27	53.18	1311k
SPSG(Ours)	**60.71**±0.51	**55.47**±0.23	132k±0.01k	**65.98**±0.34	**59.34**±0.52	195k±0.01k	**67.48**±0.21	**61.21**±0.11	271k±0.01k
Method (probability)	Indoor(10k)			Indoor(15k)			Indoor(20k)		
	Agreement	Acc	Queries	Agreement	Acc	Queries	Agreement	Acc	Queries
ZSDB3KD	59.41	58.37	1112k	63.61	63.07	988k	67.13	65.08	977k
DFMS	61.11	60.12	1009k	64.58	62.26	993k	68.14	67.18	972k
EDFBA	**61.17**	**60.42**	349k	64.17	62.12	311k	67.10	64.72	302k
DS	60.27	59.98	1021k	62.12	61.78	1010k	66.04	65.91	982k
DFME	58.71	58.17	1339k	60.21	59.26	1294k	66.27	64.16	1209k
SPSG(Ours)	58.81±0.11	57.99±0.13	137k±0.01k	**64.98**±0.34	**63.34**±0.12	181k±0.01k	**70.11**±0.21	**70.27**±0.11	267k±0.01k
Method (probability)	Caltech256 (10k)			Caltech256 (15k)			Caltehch256 (20k)		
	Agreement	Acc	Queries	Agreement	Acc	Queries	Agreement	Acc	Queries
knockoff	51.47	49.32	10k	52.31	50.67	15k	55.33	53.08	20k
ActiveThief	55.21	53.19	10k	57.41	55.36	15k	58.44	55.98	20k
Black-Box Dissector	55.61	53.12	150k	57.67	55.69	220k	58.12	55.72	300k
InverseNet	56.19	55.32	150k	57.79	55.82	220k	58.73	56.74	300k
SPSG(Ours)	**60.71**±0.51	**55.47**±0.23	132k±0.01k	**65.98**±0.34	**59.34**±0.52	185k±0.01k	**70.21**±0.21	**63.21**±0.11	271k±0.01k
Method (probability)	Diabetic5(10k)			Diabetic5(15k)			Diabetic5(20k)		
	Agreement	Acc	Queries	Agreement	Acc	Queries	Agreement	Acc	Queries
knockoff	31.12	29.32	10k	34.43	32.17	15k	39.56	38.17	20k
ActiveThief	32.21	31.19	10k	35.32	34.61	15k	38.14	37.18	20k
Black-Box Dissector	34.81	33.27	150k	36.10	36.27	220k	40.22	39.86	300k
InverseNet	35.67	34.02	150k	36.81	36.01	220k	41.12	40.16	300k
SPSG(Ours)	**36.12**±0.31	**35.25**±0.12	122k±0.01k	**38.13**±0.27	**37.24**±0.12	179k±0.01k	**42.14**±0.21	**41.27**±0.11	278k±0.01k
Method (hard-label)	CUB200 (10k)			CUB200 (15k)			CUB200 (20k)		
	Agreement	Acc	Queries	Agreement	Acc	Queries	Agreement	Acc	Queries
ZSDB3KD	24.45	23.56	1299k	26.33	26.01	1098k	31.31	29.78	931k
DFMS	26.21	25.11	1331k	29.31	28.33	1003k	31.24	30.18	939k
EDFBA	25.34	23.54	559k	27.61	26.43	490k	31.12	30.72	431k
DS	24.56	23.53	1237k	27.34	26.78	1191k	30.64	29.78	902k
knockoff	21.11	19.27	10k	24.49	22.76	15k	26.33	25.92	20k
ActiveThief	23.21	22.89	10k	26.55	25.16	15k	27.18	26.96	20k
Black-Box Dissector	25.91	23.57	150k	27.43	26.26	220k	31.59	30.46	300k
InverseNet	26.01	24.07	150k	26.93	26.12	220k	31.43	30.97	300k
SPSG(Ours)	**26.78**±0.16	**25.42**±0.23	132k±0.01k	**29.97**±0.34	**29.84**±0.52	195k±0.01k	**34.66**±0.21	**34.12**±0.19	271k±0.01k

the query results are hard labels. For CUB-200-2011 and Indoor Scenes, we conduct comparisons in data-free MS. Even though data-free MS does not necessitate real data, we employ publicly available, potentially related real images as weak image priors for the generator for fairness in comparison. Specifically, for Indoor Scenes, We use indoor scene images from SUN [47], which are distinct from Indoor Scenes categories. For CUB-200-2011, We utilize bird images whose classes are included in NAbird [41] but not present in CUB. In Caltech256 and Diabetic Retinopathy, comparisons are made for Data-driven MS. Among them, the number of queries for InverseNet and Black-Box Dissector is not determined by the number of real samples. To ensure fairness, we set the query volume for these two methods slightly higher than that of SPSG. ILSVRC-2012 [8] training set, consisting of approximately 12 million images, serves as our attack dataset in Data-driven MS. Moreover, we ensure the consistency of real samples across different methods.

Training Paradigm and Evaluation Metrics. Training paradigms are categorized based on the usage of data generators. Generally speaking, only data-free MS necessitates the use of a generator. Under the generator paradigm, the Generator is configured as BigGAN and trained using Adam with a learning rate of 0.001, β_1 = 0.5, and β_2 = 0.999. The batch size of BigGAN is 128. For data-driven MS, proxy models are trained from scratch on the attack dataset using SGD with a momentum of 0.5, a learning rate of 0.01 (decaying by a factor of 0.1 every 60 epochs), 200 epochs, and a batch size of 64. Evaluation is based on the accuracy of the proxy model on the corresponding test set and the similarity in predictions between the proxy and victim models. We also report the success rate of adversarial attacks on the victim model as indicators of the transferability of the proxy model. The superpixel segmentation for our method defaults to quickshift [42].

4.2. Experiment Results with Data-free MS

We evaluate the accuracy and agreement of the proxy models generated by data-free baselines and our algorithm under

scenarios with 10k, 15k, and 20k real samples. As illustrated in Table 2, our method almost outperforms all other MS methods across both metrics. While the performance of data-free MS plateaus with increasing numbers of real samples, our method demonstrates less pronounced diminishing returns. Additionally, we document the query volume required by all algorithms. Our method's query volume is determined by the number of queries consisting of an image and its perturbed versions using different superpixels, whereas the query volume for data-free MS depends on when the model converges. Our method requires significantly fewer queries than data-free MS, with our query volume being approximately 25% of that required by the most efficient data-free MS. Notably, despite the use of weak image priors, DFMS and ZSDB3KD sometimes fail to train, severely limiting their practical applicability.

4.3. Experiment Results with Data-driven MS

We assess the accuracy and agreement of the proxy models generated by SPSG and other data-driven MS methods under scenarios with 10k, 15k, and 20k real samples. As shown in Table 2, while our method does not minimize query volume, it yields substantial improvements. The marginal effects of data-driven MS are not pronounced across the 10k to 20k real samples range. Consequently, we observe the performance across a broader range of real sample numbers (100k-200k), finding that our algorithm consistently outperforms others in terms of accuracy. Specifically, as shown in Figure 3 and 4, at 140k real samples for Diabetic Retinopathy, Knockoff achieves its peak accuracy of 54.7%, whereas our algorithm surpasses this accuracy at 130k real samples.

4.4. Experiment Results in Hard-label Query

We evaluate all algorithms functional under hard-label queries, where the query not only returns the predicted label but also the associated confidence. We argue that providing users with the confidence associated with the predicted label is more practical for MLaaS. Despite high confidence not necessarily equating to accuracy, low confidence allows users to disregard the model's prediction. Table 2 presents our results on CUB-200-2011, showcasing our algorithm's superiority with 10k or more real samples.

4.5. Transferability of Adversarial Samples

We assess the transferability of adversarial samples generated on CUBS-200-2011 test set. The evaluation encompasses the success rates of adversarial attacks generated from different methods (FGSM, BIM, PGDK), with a perturbation bound of ε=10/255 and a step size of α=2/255. The adversarial attacks in our experiments are untargeted attacks. In untargeted attacks, adversarial samples are generated only on images correctly classified by the attacked model. As Table 3 demonstrates, the adversarial samples generated by our proxy model exhibit higher transferability to the victim model, affirming the practical applicability of our method in real-world scenarios.

Table 3. The ASR(in %) of MS methods with different adversarial attacks on CUB200. (**Boldface**: the best value.)

Method	CUBS200 (hard-label)			CUBS200 (probability)		
	FGSM	BIM	PGD	FGSM	BIM	PGDk
ZSDB3KD	21.31%	23.62%	23.58%	37.64%	37.10%	39.21%
DFMS	24.34%	22.93%	21.22%	35.22%	34.33%	36.11%
EDFBA	24.21%	23.72%	25.51%	37.88%	39.09%	38.51%
DS	25.21%	28.77%	23.51%	37.88%	39.09%	39.51%
knockoff	21.31%	24.77%	22.51%	35.12%	36.71%	37.57%
ActiveThief	24.34%	26.77%	27.52%	37.88%	36.02%	36.29%
Black-Box Dissector	24.91%	26.57%	26.41%	37.21%	36.11%	36.46%
InverseNet	24.86%	26.76%	27.22%	38.03%	37.21%	37.47%
Ours	**25.47%**	**29.03%**	**29.31%**	**38.21%**	**39.43%**	**39.68%**

Table 4. The agreement (in %) and test accuracy (in %) of different segment methods on CUB-200-2011.

Method	CUBS200 (hard-label)		CUBS200 (probability)	
	agreements	accuracy	agreements	accuracy
quickshift (132k)	26.78%	25.42%	60.71%	55.47%
felzenszwalb (1871k)	30.34%	28.93%	63.22%	59.33%
slic (371k)	29.21%	27.72%	62.28%	58.09%
Grid (2700k)	12.31%	9.72%	21.68%	19.09%

4.6. Model Stealing in Real-World Scenarios

We trained a model on the Oxford 102 Flowers dataset [25] using the Microsoft Custom Vision [23] service and designated it as a black-box victim model. The ILSVRC-2012 dataset served as the attack dataset, with the inference results of the Oxford 102 Flowers test set used as the metric. Hyperparameter settings were consistent with previous experiments. The victim model's test accuracy was 86.34%. As Table 5 demonstrates, compared to the second-best methods using 30k real samples, our method showed a 4.17% increase in test accuracy for the proxy model. This result indicates that our method possesses stronger practical applicability in real-world scenarios.

Table 5. Test accuracy of all baselines in Real-World Scenarios.

Method (probability)	Real Sample Number				
	10k	15k	20k	25k	30k
KnockoffNets	58.45%	62.12%	66.48%	69.12%	74.50%
ActiveThief	60.90%	64.21%	68.51%	70.48%	75.24%
Black-Box Dissector	60.21%	64.27%	67.52%	69.88%	73.09%
SPSG	**62.49%**	**65.93%**	**69.34%**	**71.37%**	**79.41%**

4.7. Ablation Study

Resistance to Prada. We document the monitoring of the finite difference query and SPGQ by Prada, represented by the distribution of image distances. As shown in Figure 6, Finite difference queries are completely detectable by Prada, exhibiting a significant deviation from the Gaussian distribution. In contrast, superpixel queries initially have a few detectable instances but subsequently evade detection

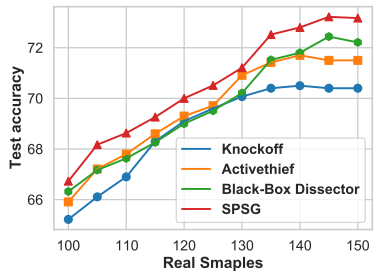


Figure 3. Baselines in CUB200

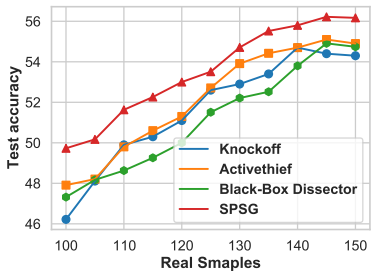
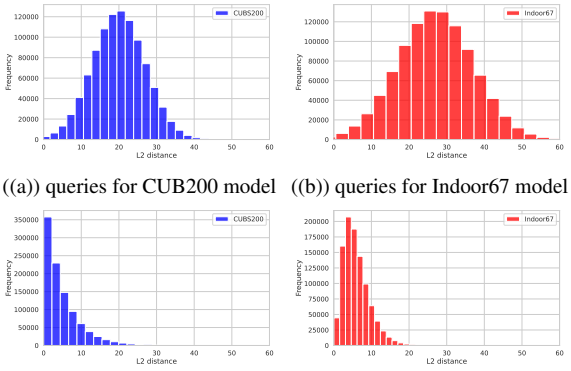


Figure 4. Baselines in Diabetic5

Figure 5. Ablation study



((a)) queries for CUB200 model ((b)) queries for Indoor67 model

((c)) queries for CUB200 model ((d)) queries for Indoor67 model

Figure 6. The first and second row show the 1000k queries' image distance distribution for SPGQ and finite difference queries, respectively.

entirely, with their image distribution closely aligning with the Gaussian distribution.

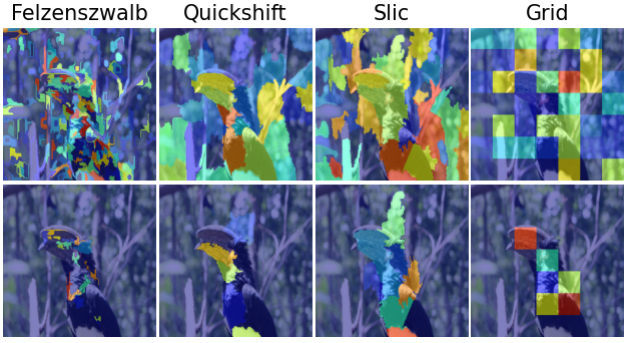


Figure 7. The first row shows the superpixel SG heatmap for different segment methods, while the second row shows the purified superpixel SG map for different segment methods.

Superpixel Segmentation. We employed the superpixel gradients obtained through queries under quick-shift [42], felzenszwalb [9], slic [1], and grid segmentation methods. Experimental results in Table 4 indicate that the more superpixels used, the more apparent the effect of model stealing becomes. However, regardless of the number of grid pixels divided, the grid segmentation method demonstrated a poor stealing effect. This is attributed to the grid segmentation's disregard for image attributes such as texture and color, which are closely associated with the model's decision-making process.

Impact of SGP. We conduct experiments on CUB-200-2011 dataset to compare the performance of SPSG without SGP and the complete SPSG. The experimental results shown in Figure 5 reveal a significant degradation in the effectiveness of SPSG when SGP is omitted. This decline can be attributed to the retention of gradient variance, which proves to be particularly detrimental to dark knowledge extraction. In Supplementary Material, we further explore the applications of SGP, including knowledge distillation.

5. Conclusion

SPSG significantly outperforms existing MS algorithms across various datasets, demonstrating its effectiveness even in hard-label query scenarios. The success of SPSG in adversarial attacks showcases its practical utility, while its capability to evade Prada highlights its stealthiness. In essence, SPSG provides a novel approach to enhancing MS performance by effectively mimicking additional information from victim models. We hope our proposed method will encourage proactive measures to protect models against unauthorized access and theft.

Acknowledgments This work was supported by the National Natural Science Foundation of China Project (62172449, 62372471, 62172441), the Joint Funds for Railway Fundamental Research of National Natural Science Foundation of China (Grant No. U2368201), special fund of National Key Laboratory of Ni&Co Associated Minerals Resources Development and Comprehensive Utilization(GZSYS-KY-2022-018, GZSYS-KY-2022-024), Key Project of Shenzhen City Special Fund for Fundamental Research(JCYJ20220818103200002), the National Natural Science Foundation of Hunan Province(2023JJ30696), and the Science Foundation for Distinguished Young Scholars of Hunan Province (NO. 2023JJ10080).

References

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels. Technical report, 2010. 2, 8

[2] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282, 2012. 3

[3] Antonio Barbalau, Adrian Cosma, Radu Tudor Ionescu, and Marius Popescu. Black-box ripper: Copying black-box models using generative evolutionary algorithms. *Advances in Neural Information Processing Systems*, 33:20120–20129, 2020. 1, 3

[4] James Beetham, Navid Kardan, Ajmal Mian, and Mubarak Shah. Dual student networks for data-free model stealing. *arXiv preprint arXiv:2309.10058*, 2023. 1, 2, 3, 5

[5] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. Ieee, 2017. 3

[6] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 839–847. IEEE, 2018. 2

[7] Jorge Cuadros and George Bresnick. Eyepacs: an adaptable telemedicine system for diabetic retinopathy screening. *Journal of diabetes science and technology*, 3(3):509–516, 2009. 5

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 6

[9] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International journal of computer vision*, 59:167–181, 2004. 2, 3, 8

[10] Ruth C Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE international conference on computer vision*, pages 3429–3437, 2017. 3

[11] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1322–1333, 2015. 1

[12] Kun Fu, Quan Jin, Runze Cui, Fei Sha, and Changqing Zhang. Axiom-based grad-cam: Towards accurate visualization and explanation of cnns. *Pattern Recognition*, 110: 107638, 2021. 2

[13] Xueluan Gong, Yanjiao Chen, Wenbin Yang, Guanghao Mei, and Qian Wang. Inversenet: Augmenting model extraction attacks with training data inversion. In *IJCAI*, pages 2439–2447, 2021. 5

[14] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. 1

[15] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 3

[16] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007. 5

[17] Dongming Han, Jiacheng Pan, Rusheng Pan, Dawei Zhou, Nan Cao, Jingrui He, Mingliang Xu, and Wei Chen. inet: visual analysis of irregular transition in multivariate dynamic networks. *Frontiers of Computer Science*, 16:1–16, 2022. 1

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 5

[19] Byeongho Heo, Minsik Lee, Sangdoo Yun, and Jin Young Choi. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3779–3787, 2019. 2

[20] Ziqi Jiang, Li Zhang, Chuang Zhang, Chunxia Li, Fei Li, and Kuiyuan Yang. Layercam: Exploring hierarchical class activation maps for localization. *IEEE Transactions on Image Processing*, 30:5875–5888, 2021. 2

[21] Mika Juuti, Sebastian Szyller, Samuel Marchal, and N Asokan. Prada: protecting against dnn model stealing attacks. In *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 512–527. IEEE, 2019. 2, 4

[22] Seung Hyun Lee, Dae Ha Kim, and Byung Cheol Song. Self-supervised knowledge distillation using singular value decomposition. In *Proceedings of the European conference on computer vision (ECCV)*, pages 335–350, 2018. 2

[23] Microsoft Corporation. Microsoft custom vision service. https://customvision.ai. 7

[24] Takeru Miyato, Andrew M Dai, and Ian Goodfellow. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*, 2016. 3

[25] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes, 2008. 7

[26] Daniel Omeiza, Simon Speakman, Celia Cintas, and Komminist Weldemariam. Smooth grad-cam++: An enhanced inference level visualization technique for deep convolutional neural network models. In *2019 15th International Conference on Signal-Image Technology Internet-Based Systems (SITIS)*, pages 272–279. IEEE, 2019. 2

[27] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Knockoff nets: Stealing functionality of black-box models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4954–4963, 2019. 1, 3, 5

[28] Soham Pal, Yash Gupta, Aditya Shukla, Aditya Kanade, Shirish Shevade, and Vinod Ganapathy. Activethief: Model extraction using active learning and unannotated public data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 865–872, 2020. 1, 3, 5

[29] Peyman Passban, Yimeng Wu, Mehdi Rezagholizadeh, and Qun Liu. Alp-kd: Attention-based layer projection for

knowledge distillation. In *Proceedings of the AAAI Conference on artificial intelligence*, pages 13657–13665, 2021. 2

[30] Yanni Peng, Xiaoping Fan, Rong Chen, Ziyao Yu, Shi Liu, Yunpeng Chen, Ying Zhao, and Fangfang Zhou. Visual abstraction of dynamic network via improved multi-class blue noise sampling. *Frontiers of Computer Science*, 17(1): 171701, 2023. 1

[31] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *2009 IEEE conference on computer vision and pattern recognition*, pages 413–420. IEEE, 2009. 5

[32] Mauro Ribeiro, Katarina Grolinger, and Miriam AM Capretz. Mlaas: Machine learning as a service. In *2015 IEEE 14th international conference on machine learning and applications (ICMLA)*, pages 896–902. IEEE, 2015. 1

[33] Sunandini Sanyal, Sravanti Addepalli, and R Venkatesh Babu. Towards data-free model stealing in a hard label setting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15284–15293, 2022. 1, 2, 3, 5

[34] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 2, 3

[35] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *Advances in Neural Information Processing Systems*, 32, 2019. 3

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

[37] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013. 3

[38] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017. 3

[39] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014. 3

[40] Jean-Baptiste Truong, Pratyush Maini, Robert J Walls, and Nicolas Papernot. Data-free model extraction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4771–4780, 2021. 1, 2, 3, 5

[41] Grant Van Horn, Steve Branson, Ryan Farrell, Scott Haber, Jessie Barry, Panos Ipeirotis, Pietro Perona, and Serge Belongie. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 595–604, 2015. 6

[42] Andrea Vedaldi and Stefano Soatto. Quick shift and kernel methods for mode seeking. In *Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part IV 10*, pages 705–718. Springer, 2008. 2, 3, 6, 8

[43] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. Technical report, 2011. 5

[44] Wenxuan Wang, Bangjie Yin, Taiping Yao, Li Zhang, Yanwei Fu, Shouhong Ding, Jilin Li, Feiyue Huang, and Xiangyang Xue. Delving into data: Effectively substitute training for black-box attack. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4761–4770, 2021. 1, 3

[45] Yixu Wang, Jie Li, Hong Liu, Yan Wang, Yongjian Wu, Feiyue Huang, and Rongrong Ji. Black-box dissector: Towards erasing-based hard-label model stealing attack. In *European Conference on Computer Vision*, pages 192–208. Springer, 2022. 3, 5

[46] Zi Wang. Zero-shot knowledge distillation from a decision-based black-box model. In *International Conference on Machine Learning*, pages 10675–10685. PMLR, 2021. 1, 2, 3, 5

[47] Jianxiong Xiao, Krista A Ehinger, James Hays, Antonio Torralba, and Aude Oliva. Sun database: Exploring a large collection of scene categories. *International Journal of Computer Vision*, 119:3–22, 2016. 6

[48] 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 *NDSS*, 2020. 1

[49] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016. 2

[50] Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. You only propagate once: Accelerating adversarial training via maximal principle. *Advances in Neural Information Processing Systems*, 32, 2019. 3

[51] Jie Zhang, Chen Chen, and Lingjuan Lyu. Ideal: Query-efficient data-free learning from black-box models. In *The Eleventh International Conference on Learning Representations*, 2022. 1, 3

[52] Jie Zhang, Bo Li, Jianghe Xu, Shuang Wu, Shouhong Ding, Lei Zhang, and Chao Wu. Towards efficient data free black-box adversarial attack. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15115–15125, 2022. 1, 2, 3, 5

[53] Huang Zhizhong, Dai Mingliang, Zhang Yi, Zhang Junping, and Shan Hongming. Point, segment and count: A generalized framework for object counting. In *CVPR*, 2024. 1

[54] Wen Zhou, Xin Hou, Yongjun Chen, Mengyun Tang, Xiangqi Huang, Xiang Gan, and Yong Yang. Transferable adversarial perturbations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 452–467, 2018. 1

[55] Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. Freelb: Enhanced adversarial training for natural language understanding. *arXiv preprint arXiv:1909.11764*, 2019. 3