Proc. of the Nineteenth Intl. Conf. on Computational Intelligence and Security (CIS 2023)
Editors Yuping Wang, Xiao-zhi Gao, Lalit Mohan Patnaik, David Camacho, and Hailin Liu

doi: 10.3850/978-981-18-8813-7_P233-cd

SODM:Stealing Object Detection Models via Diffusion Model

Yangming Zhang
School of Cyberspace Security, Hainan University, China. E-mail: zhangym121@hainanu.edu.cn

Suyu An
School of Cyberspace Security, Hainan University, China. E-mail: ansy@nipc.org.cn

Moxuan Zeng
School of Cyberspace Security, Hainan University, China. E-mail: zengmoxuan@hainanu.edu.cn

Yangzhong Wang
School of Cyberspace Security, Hainan University, China. E-mail: wangyangzhong@hainanu.edu.cn

Jun Niu
School of Computer Science and Technology, Xidian University, China. E-mail: niujun@stu.xidian.edu.cn

Yuqing Zhang*
School of Cyberspace Security, Hainan University; National Computer Network Intrusion Prevention Center, University of Chinese Academy of Sciences;School of Cyber Engineering, Xidian University, China. E-mail: zhangyq@nipc.org.cn

As deep neural networks (DNNs) demonstrate exceptional performance across multiple domains, Machine Learning as a Service (MLaaS) has gained popularity within cloud services. Deploying machine learning models in cloud services exposes them to the looming threat of model stealing attacks. Model stealing attacks primarily concentrate on image classification models in computer vision. However, research regarding model stealing attacks targeting another vital domain within computer vision, namely object detection models, still needs to be explored. We introduce an approach to steal object detection models, SODM. This method aims to expand the scope of attack scenarios in black-box settings, further relaxing attack assumptions and reducing the associated attack costs, all while achieving high-fidelity stealing of object detection models. Extensive experimental validations across various settings demonstrate our approach's excellence compared to other model stealing methods under relaxed attack assumptions. Before employing sample filtering, the fidelity of the substitute model reaches 93% of the victim model's accuracy. With the application of mutual information, we successfully reduce attack costs by 6.7% while maintaining the fidelity of the substitute model at 90%.

Keywords: Deep learning, privacy security, model stealing attacks, substitute models, diffusion models.

1. Introduction

With the widespread adoption of MLaaS in cloud services, models deployed in cloud services have become increasingly valuable. However, models deployed on cloud services are also vulnerable to a serious privacy threat known as Model Stealing Attacks (MSAs), also known as Model Extraction Attacks (MEAs) [Tramèr et al. (2016)]. This attack allows adversaries to access and observe the victim model's inputs and outputs, which are conducted in a black-box manner. Through this access and observation, attackers can construct a substitute model with similar functionality without the need for any knowledge about the internals of the victim model or its training data. Once the functionality of DNNs is stolen, attackers gain not only the substitute model for potential gains but also the ability to launch deeper white-box attacks.

Research efforts in MSAs have predominantly centered on image classification models in computer vision. However, limited attention has been devoted to MSAs targeting object detection mod-

els. Three factors contribute to the relative scarcity of research in this area. Firstly, suppose the public dataset is used as the query data for object detection models. In that case, the attack effect mainly depends on the distribution similarity between the public and training datasets. Furthermore, if synthetic data generated by generative models are employed as a substitute training set, the performance of the substitute model is substantially reliant on the quality and distribution of these synthetic examples. The prevailing generative models are Generative Adversarial Networks (GANs) [Truong et al. (2021)]. The GANs training is complicated, resulting in lower-quality generated images and a dearth of diversity. Finally, object detection models typically provide limited information, returning predicted locations and their corresponding hard labels. This restricted information necessitates a substantial volume of queries to the victim model.

The recent outstanding performance of diffusion models in image generation tasks has garnered significant attention, inspiring our work. In order to solve the difficulties in MSAs targeting object detection models, we introduce the SODM method for stealing the functionality of object detection models. SODM employs diffusion models instead of conventional GANs to generate a high-quality substitute training dataset. Compared to prior research, we further relax the assumptions of MSAs, rendering SODM more aligned with real-world attack scenarios. We incorporate mutual information to reduce the number of queries attackers make to the victim model. We apply two data augmentation strategies to the selected example data to delve deeper into the victim model's latent knowledge and internal information. Through an extensive series of experiments, our approach demonstrates substantial improvements in the accuracy of the substitute model and a successful reduction in attack costs under more permissive attack scenarios compared to other model stealing methods. Moreover, even if the structure of the substitute model is different from the victim model, our experimental results show that SODM can still obtain a high-fidelity substitute model. In summary, our contributions are as follows:

- Our approach relaxes the assumptions in MSAs against object detection models.
- While ensuring minimal variations in the fidelity of the substitute model, our method reduces the attacker's cost by 6.7%.
- Our experimental results demonstrate that our approach enhances the accuracy by 10% and the fidelity by 13% of the substitute model in lenient attack scenarios compared to state-of-the-art model-stealing methods.

2. Background and Related Work

2.1. *Model Stealing Attacks*

MSAs refer to malicious queries made by attackers on machine learning models to acquire corresponding predictions for the inputs from the victim model. This "input-prediction pair" is used to construct a substitute model training dataset. In doing so, attackers can create a substitute model with similar functionality, all without accessing the original training data of the victim model. When attackers do not know the model's internal information and training data, this approach allows them to obtain crucial insights into the model and even replicate its functionality.

2.2. *Diffusion Models*

As a deep generative model, the diffusion model [Ho et al. (2020)] operates based on a sequence of distinctive steps to shape new data representations from available training data. Stable Diffusion is a deep learning text-to-image generation model, which is a variant of the diffusion model known as the "Latent Diffusion Model" (LDM) [Rombach et al. (2022)]. Stable Diffusion is primarily employed for generating detailed, high-quality images based on textual descriptions. The model facilitates the generation of new images by utilizing prompt words that describe elements to be included or omitted in the generated images. This mechanism inspires our attack method.

2.3. *Existing MSAs*

Based on the diverse attack strategies employed by adversaries, we categorize MSAs into three distinct types.

2.3.1. *Side-Channel Attacks*

Side-channel MSAs leverage certain information leaked by the victim model during its execution, such as computation time [Hu et al. (2020)], power consumption [Zhang et al. (2021)], electromagnetic emissions [Maia et al. (2021)], cache behaviors [Weiss et al. (2023)], port traffic [Zhu et al. (2021)], and scientific plots [Zhang et al. (2023)]. Adversaries endeavor to gain insights into the internal mechanisms of the model by analyzing this side-channel information.

2.3.2. *Training Metamodel Attacks*

Training Metamodel attacks represent a distinct category of MSAs, where the metamodel itself serves as a classifier used to predict the classifications of the victim model [Zhang et al. (2023); Xiang et al. (2020)]. In this type of attack, the adversary first submits specific inputs, denoted as x, to the victim model and obtains corresponding outputs, denoted as y. Subsequently, the attacker proceeds to train a metamodel, denoted as f_m, which maps the target model's outputs y back to the original inputs x, i.e., $x = f_m(y)$. Through this approach, the trained metamodel can further predict the intrinsic attributes of the victim model from its given predictive outputs y. This attack leverages potential correlations between the output results of a classification model and its internal structure, enabling an effective stealing attack on the victim model.

2.3.3. *Training Substitute Model Attacks*

Many researchers have widely adopted the strategy of training a substitute model for MSAs. The fundamental idea behind this approach is to utilize the labeled data generated by the victim model to train a substitute model. This substitute model can be regarded as a simulator of the victim model, primarily aiming to emulate the behavioral characteristics of the victim model. In this approach, attackers gather a substantial amount of query inputs, denoted as x, and the corresponding outputs from the victim model are represented as y. This dataset is then used to train a substitute model, denoted as f_s, where $y = f_s(x)$. In this attack, the victim model is regarded as a label generator, and the labels it generates are employed to train the substitute model. While the substitute model can share the same architecture as the target model, it is not mandatory. As demonstrated by several studies such as [Pal et al. (2020); Wen et al. (2021)], existing research has shown that even when the substitute model employs a different architecture, it can still achieve impressive attack performances. Our approach draws inspiration from this attack method and extends its application to object detection models.

3. Methods

3.1. *Problem Formulation*

Victim Model. The victim object detection model, denoted as F_V, represents the target attackers aim to steal. Typically, the victim model is trained on a specific training dataset D_V, which uniquely determines the number of objects of detection categories, denoted as N_C. Providers of these models usually offer users the capability to query these models through APIs, making them black-box for users.

Stealing the functionality. The victim object detection model provides an API for user access. The attacker's goal is to obtain a substitute object detection model F_S that has similar functionality with the victim object detection model F_V, i.e., $F_S \approx F_V$. The attacker can only gather feedback on examples through queries made to the victim model's API to construct the substitute model training dataset D_S. Therefore, the performance of MSAs largely depends on the distributional differences between D_S and D_V. The act of attackers stealing the functionality of the object detection model can be described as:

$$\ell_{goal} = arg \min_{\theta_S} \mathbb{E}_{x \sim D_S}[\mathcal{L}(F_V(x), F_S(x))] \quad (1)$$

where θ_S represents the parameters of the substitute model F_S. Since the attacker cannot access the true annotations $\mathcal{T}_{true}$ of the victim model's training set, we employ the pseudo-labels generated by the victim model's outputs as the ground truth for training the substitute model. $\mathcal{L}$ denotes the loss function between the pseudo-labels generated by the victim model's outputs and the substitute model's outputs.

Attackers' background knowledge. Detailed information about the internal structure, training dataset, and included categories of the victim model is entirely hidden from the attackers. They cannot access this information, and their only recourse is to interact with and observe the victim model through the API it provides.

3.2. *Methods of Feedback Information Improvement*

3.2.1. *Adversarial examples*

In image classification, adversarial examples can mislead models into making incorrect predictions. However, most mainstream models employ non-maximum suppression algorithms in object detection. Even if adversarial examples successfully attack the best prediction box, the model may still choose a suboptimal one near the best one. These characteristics make adversarial attacks against object detection models challenging to execute. Nonetheless, we can precisely leverage these traits to delve deeper into the latent knowledge within the victim model. Existing research [Goodfellow et al. (2014)] has demonstrated the transferability of adversarial examples. To minimize access to the victim model during the adversarial example generation process, we can construct an attack model, denoted as F_A, to perform white-box adversarial attacks NAA [Zhang et al. (2022)]. The objective function for generating adversarial examples in this model is as follows:

$$arg\max_{x_A} \mathcal{L}(F_A(x_N), F_A(x_A)) \tag{2}$$

where x_N represents normal examples, x_A represents adversarial examples generated through adversarial attacks, and $\mathcal{L}$ denotes the loss function of the attack model F_A.

3.2.2. *Random erasing*

Zhong et al.[Zhong et al. (2020)] introduced a novel data augmentation method for Convolutional Neural Networks (CNNs) known as the random erasing strategy. Leveraging the characteristics of this approach, we apply random erasing to normal examples. Subsequently, we send these erased examples to the victim model. If partial critical regions are erased, the victim model's predictions for the erased examples may differ from those for the original examples. This assists us in gaining insights into the decision boundaries of the victim model.

3.3. *Method of Attack Costs Reduction*

We aim to reduce the number of queries to the victim model by computing mutual information values to filter more representative examples. The original substitute training set is D_S. The filtered training set, obtained through mutual information value selection, is represented as D'_S, where $D'_S \subset D_S$. For any two examples $X \in D_S$ and $Y \in D_S$, we compute their mutual information value MIV. According to the definition of mutual information [Cover (1999)], the mutual information value calculation for examples X and Y is as follows:

$$MIV(X,Y) = \sum_{x\in X}\sum_{y\in Y} P(x,y) log \frac{P(x,y)}{P(x)P(y)} \tag{3}$$

where $P(x)$ represents the marginal probability distribution function of image X, indicating the probability of X taking the value x. $P(y)$, Y and y are similar to $P(x)$, X and x. $P(x,y)$ stands for the joint probability distribution function of image X and image Y simultaneously taking values x and y. For each pair of samples (X,Y) in the original substitute training set D_S, where $X \in D_S$, $Y \in D_S$, and $X \neq Y$, we calculate their mutual information value using Equation 3.

3.4. *Overall Attack Framework*

As shown in Figure 1, we construct a substitute model training dataset using the diffusion model in the first stage. To do this, we explore and collect a catalog of object categories detectable by the victim model. This involves gathering online images of various objects and submitting them to the victim model for prediction results. We set a threshold, denoted as λ, for the confidence score in the predictions made by the victim model. Categories surpassing this threshold are added to the catalog. The quality of images generated by the diffusion model heavily relies on the prompt

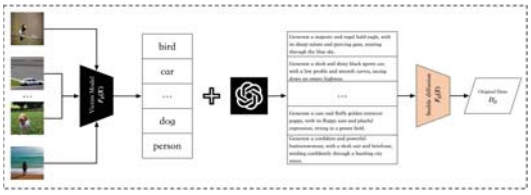

Fig. 1. The process of generating synthetic data by Stable Diffusion

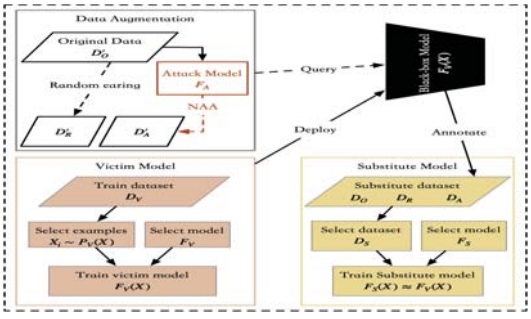

Fig. 2. Overall framework of SODM

words. Attackers can generate a series of prompt words corresponding to the catalog of categories using ChatGPT based on the victim model's category directory. With these prompt words, the diffusion model can generate the original unlabeled dataset D'_O.

Furthermore, data augmentation is applied to the original substitute training dataset D'_O. As illustrated in Figure 2, two strategies for data augmentation are employed on the original synthetic examples to fully exploit the victim model's potential information and decision boundaries. One strategy involves data augmentation through random erasing, resulting in the dataset D'_R. The other strategy employs the method for generating highly transferable adversarial examples (NAA), resulting in the dataset D'_A. In this manner, we obtain three unlabeled candidate substitute datasets.

Finally, we construct VOC-formatted training datasets, denoted as D_S, and train the substitute model based on this dataset. The attacker provides the unlabeled datasets, D'_O, D'_R, and D'_A, to the victim model. Following the annotations provided by the victim model, annotated substitute datasets, D_O, D_R, and D_A, are created in the standard

Algorithm 1: Stealing Object Detection Models

Input: Victim model F_V, attack model F_A, substitute model F_S, Stable diffusion F_G, query budget $\mathcal{Q}$, internet images $D_I = \{i_1, i_2, \ldots, i_n\}$, object categories D_C, learning rate η, substitute model iterations e, input prompts D_P, threshold λ
Output: Trained F_S

initialization: $D_C \leftarrow \{\}, \lambda \leftarrow 0.8, e \leftarrow 15, \eta \leftarrow 0.005$.
for $k = 1 \ldots n$ **do**
  if $F_V(i_k) > \lambda$ **then**
    $D_C = D_C \cup \{i_k\}$
  end
end
employing chatGpt to generate prompts $D_P = \{p_1, p_2, \ldots, p_m\}$ based on D_C
for $j = 1 \ldots m$ **do**
  `// generating unlabeled synthetic data` D'_O `via Stable Diffusion`
  $D'_O = D'_O \cup \{F_G(p_j)\}$
end
`// explore the internal knowledge of victim models`
erase samples in D'_O by random erasing to generate unlabeled D'_R
execute NAA attack in D'_O to generate unlabeled D'_A according to Eq.2
while $\mathcal{Q} > 0$ **do**
  `// receive the pseudo-labels of victim model`
  $D_O, D_R, D_A \leftarrow F_V(D'_O, D'_R, D'_A)$
  `// attacker has a finite query budget, Q denotes query budget`
  update remaining query budget $\mathcal{Q}$
end
Select a best dataset from $\{D_O, D_R, D_A, D_O \cup D_R, D_O \cup D_A, D_R \cup D_A, D_O \cup D_R \cup D_A\}$ as substitute dataset D_S
`// reduce attack costs`
if *is_mutualinfo is True* **then**
  filter examples in D_S according to Eq.3
end
for $j = 1 \ldots e$ **do**
  $x \sim D_S$
  compute: $F_S(x)$,$\mathcal{L}(x) = \mathcal{L}(F_V(x), F_S(x))$,$\nabla_{\theta_S} \mathcal{L}(x)$
  `// update` θ_S `according to Eq.1`
  $\theta_S = \theta_S - \eta \nabla_{\theta_S} \mathcal{L}(x)$
end

VOC2012 dataset format. These datasets serve as candidate training sets for the substitute model. Simultaneously, the attacker selects an appropriate model architecture for the substitute model and conducts training. Our complete attack procedure is presented clearly in Algorithm 1.

4. Experiments

4.1. *Experimental Setup*

Dataset: In our experiments, the victim model utilizes the standard VOC2012 dataset, which consists of 20 target object categories. We select the training set of VOC2012, comprising 5,717 examples, as the training dataset for the victim model. The victim and substitute models are tested on the VOC2012 test set containing 5,823 examples.

Model architectures: We have chosen RetinaNet as the architecture for the victim model, denoted as F_V. It utilizes ResNet-50 as the backbone and has been pre-trained on the Microsoft COCO dataset. For the structure of the substitute model F_S, we have selected three different architectures:

Table 1. Experiments of Different Substitute Models Architectures and Training Data Groups.

Model	Structure	Dataset	Queries	mAP	mAP_0.5	mAP_0.75	mAP_s	mAP_m	mAP_l
F_V	Retinanet_Resnet50	VOC2012	5717	0.537	0.769	0.582	0.203	0.395	0.598
F_S	Retinanet_Resnet50	D_O	9633	0.430	0.664	0.465	0.184	0.336	0.476
		D_R	9460	0.480	0.720	0.526	0.209	0.363	0.533
		D_A	9155	**0.483(0.90×)**	0.727(0.95×)	0.522(0.90×)	0.209	**0.369**	**0.536**
		$D_O \cup D_R$	14295	0.481	0.720	0.525	0.209	0.366	0.532
		$D_O \cup D_A$	13964	0.483	0.728	0.523	0.204	0.367	0.536
		$D_R \cup D_A$	8993	0.481	0.727	0.523	0.205	0.371	0.533
		$D_O \cup D_R \cup D_A$	18626	0.481	0.722	0.520	0.204	0.360	0.533
	SSD_Resnet50	D_O	9633	0.335	0.557	0.356	0.049	0.187	0.400
		D_R	9460	0.358	0.582	0.384	0.059	0.191	0.429
		D_A	9155	0.352	0.577	0.379	0.046	0.183	0.424
		$D_O \cup D_R$	14295	0.362	0.592	0.388	0.057	0.200	0.431
		$D_O \cup D_A$	13964	0.361	0.586	0.385	0.056	0.191	0.433
		$D_R \cup D_A$	8993	0.352	0.579	0.376	0.051	0.184	0.424
		$D_O \cup D_R \cup D_A$	18626	**0.367(0.68×)**	**0.593(0.77×)**	**0.399(0.69×)**}	0.058	**0.206**	**0.438**
	Faster-RCNN_Resnet50	D_O	9633	0.427	0.705	0.454	0.189	0.356	0.466
		D_R	9460	0.434	0.708	0.463	0.215	0.355	0.477
		D_A	9155	0.429	0.717	0.452	0.189	0.354	0.469
		$D_O \cup D_R$	14295	0.428	0.702	0.458	0.169	0.353	0.468
		$D_O \cup D_A$	13964	0.436	0.714	0.460	0.188	0.353	0.478
		$D_R \cup D_A$	8993	0.429	0.716	0.463	0.203	0.364	0.468
		$D_O \cup D_R \cup D_A$	18626	**0.441(0.82×)**	**0.722(0.94×)**	**0.471(0.81×)**	**0.216**	0.359	**0.483**

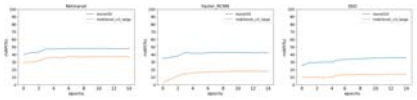

Fig. 3. Experiments of different backbones

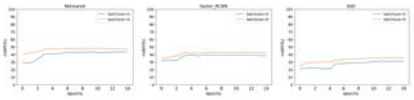

Fig. 4. Experiments of different hyperparameters

RetinaNet, Faster R-CNN, and SSD. To investigate the impact of the backbone on the model stealing attack, we have employed two different backbones, namely ResNet-50 and MobileNet-v3-large, for these three substitute models. As for the attack model F_A, we use the publicly available Mask-RCNN model on PyTorch.

Evaluation metrics: Microsoft COCO evaluation metrics are widely adopted for assessing the performance of object detection models. Therefore, we have chosen these metrics as the evaluation criteria in our experiments. For all the metrics mentioned in the experimental results, higher values indicate better performance in the model stealing attack.

4.2. *Ablation Studies*

In this section, we conduct ablation studies to analyze four factors that impact the accuracy and fidelity of the substitute model.

4.2.1. *Architecture of substitute models*

We select three substitute models with different architectures for the model stealing attack and observe their attack performances. These three substitute models have structures corresponding to RetinaNet, Faster R-CNN, and SSD, all utilizing ResNet-50 as their backbone. As shown in Table 1, the results indicate that when the substitute model's structure is RetinaNet, it achieves the best attack performance. This aligns with our expectations: the closer the structure of the substitute model matches that of the victim model, the higher the fidelity of the substitute model. As seen in Table 1, when the substitute model's structure is Faster R-CNN, its attack performance is close to that of the RetinaNet-based substitute model. Despite the significant structural differences between the Faster R-CNN model and the victim model, the fidelity of the substitute model obtained closely resembles that of the RetinaNet-based substitute model. This suggests that even when attackers use substitute models with structures different from the victim model, they can also achieve outstanding model stealing attack performances.

4.2.2. *Different training data groups*

To identify the optimal substitute model dataset for achieving the best attack performance, we conduct model stealing attack experiments on three different datasets: D_O, D_R, and D_A, as well

Table 2. Comparison Results with Other Methods.

Methods	Data Sources	Queries	mAP	mAP_0.5	mAP_0.75	mAP_s	mAP_m	mAP_l
Knockoff	public dataset(100%)	35780	0.073(0.14×)	0.207(0.27×)	0.031(0.05×)	0.027	0.062	0.082
Imitated Detectors	public dataset(25%)	12000	0.376(0.70×)	0.612(0.80×)	0.376(0.65×)	0.141	0.298	0.413
SODM	synthetic data(100%)	8993	**0.429(0.80×)**	**0.716(0.93×)**	**0.463(0.80×)**	**0.203**	**0.364**	**0.468**
SODM-MI	synthetic data(100%)	8390	0.416(0.77×)	0.689(0.90×)	0.439(0.75×)	0.187	0.343	0.455

as their unions. Results in Table 1 reveal that a larger training dataset only sometimes leads to better performances for substitute models with three distinct structures. Taking the example of the RetinaNet-ResNet50 structured substitute model, although the model extraction performance is similar for substitute model training datasets $D_O \cup D_A$ and D_A, the former implies more query attempts, translating to higher attack costs.

4.2.3. *Backbone of substitute models*

We conduct MSAs for the three distinct substitute model structures using two different backbones for each substitute model. As illustrated in Figure 3, when the backbone of the three substitute models is changed to MobileNet-v3-large, i.e., not belonging to the ResNet family like the victim model's backbone (ResNet-50), the fidelity of all three substitute models significantly decreases compared to when their backbone is ResNet-50.

4.2.4. *Hyperparameters of substitute models*

Selecting appropriate hyperparameters for the substitute model is crucial for successfully executing MSAs. In theory, the closer the hyperparameters of the substitute model align with those of the victim model, the better the effectiveness of the model stealing attack should be. However, we discover this is not always true in our experiments. As illustrated in Figure 4, regardless of the model architecture, setting the batch size of the substitute model to 4, matching that of the victim model, results in lower fidelity than when the batch size of the substitute model is set to 8.

4.3. *Comparison Results*

As shown in Table 2, we have compared our method and various model stealing attack approaches. It's worth noting that our selection of RetinaNet-ResNet50 as the substitute model architecture represents a relatively stringent choice. Therefore, in our experiments comparing our approach with other methods, we opt for the Faster R-CNN as our substitute model structure. The methods we compare in our experiments include Knockoff [Orekondy et al. (2019)], Imitated Detectors [Liang et al. (2022)], our method (SODM), and our method combined with mutual information (SODM-MI). We report the values of six standard COCO evaluation metrics for each attack method. To provide a clearer comparison of the performance of our method SODM, we calculate the fidelity for each attack method. The fidelity represents the evaluation metrics of the substitute models obtained through various model extraction attacks divided by the evaluation metrics of the victim model on the victim model's test dataset.

Both the Knockoff and Imitated Detectors attack methods require access to parts of the training data distribution of the victim model, which can be considered as gray-box threat models from a threat perspective. In contrast, our work outperforms two gray-box attack schemes across all six metrics without prior knowledge of the victim model's training data distribution. Specifically, SODM achieves a mAP_0.5 of 71.6% (with a fidelity of 93% to the victim model). SODM-MI shows only slight decreases in evaluation metrics, with a 6.7% reduction in query cost.

5. Conclusion

In this paper, we propose an attack method for stealing object detection models called SODM. This method is designed for black-box attack scenarios and achieves a highly accurate model stealing attack of object detection models while relaxing attack assumptions and reducing attack costs. We use the diffusion model to replace traditional GANs for constructing a high-quality and uniformly distributed substitute model training dataset. In order to explore their internal knowledge, we have employed adversarial examples and random erasing schemes. Simultaneously, we

utilize mutual information to filter the generated examples to reduce attack costs. Our method demonstrates a more tolerant attack assumption through extensive experimental validations than other model stealing methods. With the application of mutual information, we have reduced attack costs by 6.7%, resulting in substitute model fidelity reaching 93% and 90% compared to the victim model before and after the reduction, respectively. We believe that our attack method is viable in practical application scenarios, and new defense mechanisms should be developed to counteract this potential threat.

Acknowledgement

This work was supported by the National Key Research and Development Program(2023YFB3106400, 2023QY1202), the National Natural Science Foundation of China(U1836210), and the Key Research and Development Science and Technology of Hainan Province(GHYF2022010).

References

Cover, T. M. (1999). *Elements of information theory*. John Wiley & Sons.

Goodfellow, I. J., J. Shlens, and C. Szegedy (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Ho, J., A. Jain, and P. Abbeel (2020). Denoising diffusion probabilistic models. *Advances in neural information processing systems 33*, 6840–6851.

Hu, X., L. Liang, S. Li, L. Deng, P. Zuo, Y. Ji, X. Xie, Y. Ding, C. Liu, T. Sherwood, et al. (2020). Deepsniffer: A dnn model extraction framework based on learning architectural hints. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 385–399.

Liang, S., A. Liu, J. Liang, L. Li, Y. Bai, and X. Cao (2022). Imitated detectors: Stealing knowledge of black-box object detectors. In *Proceedings of the 30th ACM International Conference on Multimedia*, pp. 4839–4847.

Maia, H. T., C. Xiao, D. Li, E. Grinspun, and C. Zheng (2021). Can one hear the shape of a neural network?: Snooping the gpu via magnetic side channel. *arXiv preprint arXiv:2109.07395*.

Orekondy, T., B. Schiele, and M. Fritz (2019). Knockoff nets: Stealing functionality of black-box models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4954–4963.

Pal, S., Y. Gupta, A. Shukla, A. Kanade, S. Shevade, and V. Ganapathy (2020). Activethief: Model extraction using active learning and unannotated public data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Volume 34, pp. 865–872.

Rombach, R., A. Blattmann, D. Lorenz, P. Esser, and B. Ommer (2022). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695.

Tramèr, F., F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart (2016). Stealing machine learning models via prediction {APIs}. In *25th USENIX security symposium (USENIX Security 16)*, pp. 601–618.

Truong, J.-B., P. Maini, R. J. Walls, and N. Papernot (2021). Data-free model extraction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4771–4780.

Weiss, J. O., T. Alves, and S. Kundu (2023). Ezclone: Improving dnn model extraction attack via shape distillation from gpu execution profiles. *arXiv preprint arXiv:2304.03388*.

Wen, T., H. Hu, and H. Zheng (2021). An extraction attack on image recognition model using vae-kdtree model. In *International Workshop on Advanced Imaging Technology (IWAIT) 2021*, Volume 11766, pp. 128–131. SPIE.

Xiang, Y., Z. Chen, Z. Chen, Z. Fang, H. Hao, J. Chen, Y. Liu, Z. Wu, Q. Xuan, and X. Yang (2020). Open dnn box by power side-channel attack. *IEEE Transactions on Circuits and Systems II: Express Briefs 67*(11), 2717–2721.

Zhang, B., X. He, Y. Shen, T. Wang, and Y. Zhang (2023). A plot is worth a thousand words: Model information stealing attacks via scientific plots. *arXiv preprint arXiv:2302.11982*.

Zhang, J., W. Wu, J.-t. Huang, Y. Huang, W. Wang, Y. Su, and M. R. Lyu (2022). Improving adversarial transferability via neuron attribution-based attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14993–15002.

Zhang, Y., R. Yasaei, H. Chen, Z. Li, and M. A. Al Faruque (2021). Stealing neural network structure through remote fpga side-channel analysis. *IEEE Transactions on Information Forensics and Security 16*, 4377–4388.

Zhong, Z., L. Zheng, G. Kang, S. Li, and Y. Yang (2020). Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, Volume 34, pp. 13001–13008.

Zhu, Y., Y. Cheng, H. Zhou, and Y. Lu (2021). Hermes attack: Steal {DNN} models with lossless inference accuracy. In *30th USENIX Security Symposium (USENIX Security 21)*.