Protecting Object Detection Models From Model Extraction Attack via Feature Space Coverage

Zeyu Li[1] , **Yuwen Pu**[1*] , **Xuhong Zhang**[2] , **Yu Li**[3] , **Jinbao Li**[4] and **Shouling Ji**[1]
[1]College of Computer Science and Technology, Zhejiang University
[2]School of Software Technology, Zhejiang University
[3]School of Computer Science and Technology, Harbin Institute of Technology, ShenZhen
[4]School of Mathematics and Statistics, Qilu University of Technology
{zju_lzy, yw.pu,zhangxuhong}@zju.edu.cn, li.yu@hit.edu.cn, lijinb@sdas.org, sji@zju.edu.cn

Abstract

The model extraction attack is an attack pattern aimed at stealing well-trained machine learning models' functionality or privacy information. With the gradual popularization of AI-related technologies in daily life, various well-trained models are being deployed. As a result, these models are considered valuable assets and attractive to model extraction attackers. Currently, the academic community primarily focuses on defense for model extraction attacks in the context of classification, with little attention to the more commonly used task scenario of object detection. Therefore, we propose a detection framework targeting model extraction attacks against object detection models in this paper. The framework first locates suspicious users based on feature coverage in query traffic and uses an active verification module to confirm whether the identified suspicious users are attackers. Through experiments conducted in multiple task scenarios, we validate the effectiveness and detection efficiency of the proposed method.

1 Introduction

With the rapid development of Machine Learning as a Service (MLaaS), Well-trained models deployed on cloud service platforms have become attractive targets for attackers. These attackers construct query samples, access the victim model, and train substitute models based on the query results. Such model extraction attacks replicate the functionality and characteristics of the victim model, posing a significant threat to the copyright and privacy of the victim models. In 2023, ByteDance was exposed for using OpenAI's API to train its own large language model and employing data desensitization methods to conceal evidence, sparking a commercial dispute between the two companies. Such business disputes further drew attention from society regarding model copyright.

In recent years, various research [Jia *et al.*, 2021; Mazeika *et al.*, 2022] efforts have emerged in model extraction defense to mitigate the risks associated with these attacks on AI cloud services. Existing model extraction detection methods predominantly focus on classification models[Sadeghzadeh *et al.*, 2023], encoder models [Cong *et al.*, 2022], and generation models [Qiao *et al.*, 2023]. Recent studies [Li *et al.*, 2023; Liang *et al.*, 2022] have demonstrated the effectiveness of model extraction attacks against object detection models. However, to our knowledge, there is still no research focused on model extraction defense for widely used object detection models. To bridge this research gap, we explore defense countermeasures explicitly for object detection models and propose a detection method called OSD (Object detector Steal Detection).

Model extraction defenses primarily consist of proactive defense methods involving proactive perturbations [Mazeika *et al.*, 2022] and passive defense methods such as model watermark [Jia *et al.*, 2021] and model fingerprint [Huang *et al.*, 2023]. However, both methods have their limitations. They cannot achieve both preventing the attack from occurring in the query stage and ensuring the usability of benign users; therefore, they may not meet the defense requirements of real-world MLaaS platforms. To address this, we adopt a more traditional defense approach based on traffic monitoring, aiming to analyze the distribution of query traffic to identify attackers and disrupt the extraction process while minimizing the impact for benign users. Using such methods, We aim to identify attackers accurately and assure model owners that their victim models will not be extracted. According to our observation, model extraction attacks against object detection models primarily query the victim model with in-distribution data that tends to cover more functional spaces. Therefore, it can be identified by analyzing the query samples' coverage in the victim model's feature space. Motivated by this insight, we propose an abnormal query detection framework based on feature space coverage analysis.

Furthermore, we have observed that defense methods based on traffic monitoring in the object detection scenario face two challenges and requirements: 1. **Accuracy requirement**: Traffic monitor-based detection methods need high accuracy to minimize the impact on benign users. 2. **Efficiency requirement**: In the streaming data scenario, where large amounts of data flow continuously and defense needs to be individually deployed for each user, detection methods must exhibit high execution efficiency to ensure prompt query re-

[*] Corresponding author

sponse time. To meet the accuracy requirement, we design a proactive confirmation module that reduces the false positive rate. We also optimize the feature comparison process in traffic monitoring to minimize deployment costs. In summary, our work highlights the following contributions:

- To the best of our knowledge, we are the first to address model extraction defense for object detection models, identifying the unique distribution characteristics of query traffic in object detector extraction.
- Targeting the characteristics of extracting object detection models, we propose to detect abnormal queries by analyzing the feature space coverage of query samples.
- To improve detection accuracy, we introduce a proactive confirmation mechanism to reduce false positives for benign users.
- Through experiments conducted on various real-world datasets, we validate the effectiveness and efficiency of the proposed methods.

2 Related Work

2.1 Object Detection

Object detection is one of the most classical machine learning tasks, aiming to recognize and detect objects of specific classes within a given image while annotating their positions using two-dimensional or three-dimensional bounding boxes and outputting their categories. Current mainstream object detection models are categorized into anchor-based and anchor-free detectors based on whether predefined anchor boxes are utilized in the model. In anchor-free object detection models, Fast-RCNN [Girshick, 2015] stands out as a representative two-stage detector, while YOLOv3 [Redmon and Farhadi, 2018] and YOLOv5 [Jocher *et al.*, 2021] are commonly used one-stage detectors known for their faster detection speed. Anchor-free detectors are not influenced by anchor box calculations, and representative models include CornerNet [Law and Deng, 2020], ExtremeNet [Zhou *et al.*, 2019], and AOPG [Cheng *et al.*, 2022].

2.2 Model Extraction Attack

Existing query-based model extraction attacks are designed for two settings: data-sufficient and data-free, based on the number of real samples the attacker collects. In the data-sufficient scenario, the attacker typically gathers an unlabeled dataset of the target task. CloudLeak [Yu *et al.*, 2020b] employs active learning to construct an efficient query dataset and fits the classification boundaries with adversarial examples. Chen et al. [Chen *et al.*, 2023] proposed a defense-penetration model extraction attack called D-DAE. In the data-limited scenario, the attacker has only a few or even no real samples and synthesizes samples using generation models like GAN [Goodfellow *et al.*, 2020]. MAZE [Kariyappa *et al.*, 2021] utilized GAN-generated samples to construct the query dataset and obtained approximate victim model gradients to train the generator. The further work [Sanyal *et al.*, 2022] explored data-free extraction in a hard-label setting. QUDA [Lin *et al.*, 2023] reduced the query budget by introducing a reinforcement learning-based I-FGSM algorithm. For object detectors, current works have conducted research in both data-free [Liang *et al.*, 2022; Shah *et al.*, 2023] and data-sufficient [Li *et al.*, 2023] scenarios.

2.3 Model Extraction Defense

Model extraction defense can be categorized into proactive and passive defense. Proactive defense methods introduce perturbations on query results to disrupt the attack. Passive defense methods identify suspicious users by watermark verification or traffic monitoring. For proactive defense, OOD Detector [Kariyappa and Qureshi, 2020] employed out-of-distribution detection to identify suspicious queries and added adaptive perturbations to the returned results. Other proactive methods include Prediction poisoning [Orekondy *et al.*, 2019] and HSYA [Mazeika *et al.*, 2022]. Passive defense methods can be further divided into verification-based and monitor-based. The former employs techniques such as model watermark or fingerprint to identify attackers. The latter identifies potential attackers during the attack query process through statistical analysis of query distributions. Entangled Watermarks [Jia *et al.*, 2021] addressed the issue of traditional backdoors being easily lost during the extraction by coupling task and watermark information during training. Peng et al. [Peng *et al.*, 2022] proposed a watermark generation method based on universal adversarial perturbation. There are currently existing model watermark methods for GANs [Qiao *et al.*, 2023] and encoders [Cong *et al.*, 2022]. In monitor-based defense methods, PARDA [Juuti *et al.*, 2019] utilized the distance distribution of query samples, and HODA [Sadeghzadeh *et al.*, 2023] analyzed the difficulty distribution. Some recent monitor-based research includes CIP [Zhang *et al.*, 2023] and AMAO [Jiang *et al.*, 2023].

3 Problem Description

3.1 Attack Objective

In the scenario of a model extraction attack against object detectors, the attacker's objective is to query the victim model through an API, build a query dataset, and obtain a substitute model similar to the victim model M_v. The attacker aims to replicate the functionality of the victim model or exploit its vulnerabilities for further attacks. It is worth noting that the samples used by the attacker for querying can be publicly collected by the attacker or generated by generation models.

3.2 Defense Objective

The model owners deploy their model M_v on a cloud service platform and charge users for accessing the model through queries. They also possess a victim dataset D_v for the training of M_v. The objective of the model owner is to accurately determine the identity of each querying user with as little historical query traffic as possible. If an attacker is identified, the defense mechanism aims to prevent the attack by blocking their access, thereby minimizing the loss.

3.3 Research Question

Based on the defense objectives of the model owner mentioned above, we summarize a set of research questions, and experiments will be conducted to address these questions:

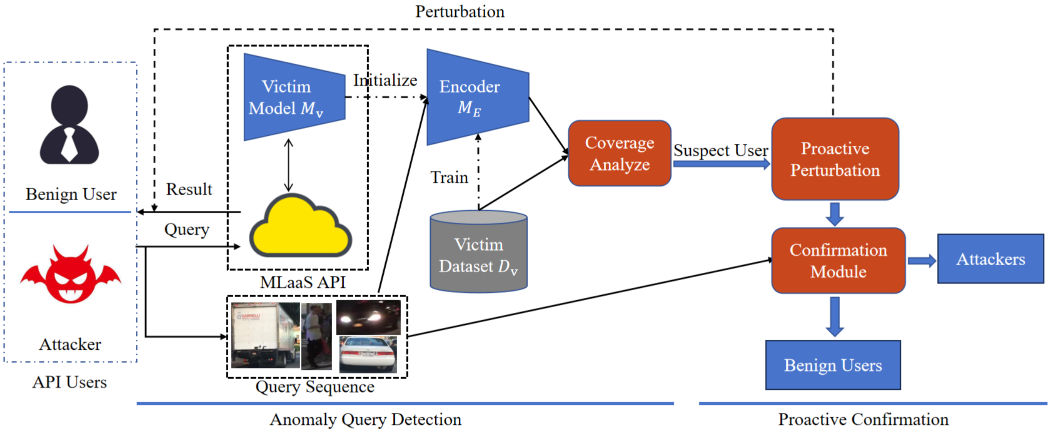


Figure 1: The overall framework of the OSD. OSD contains two phases: abnormal query detection and proactive confirmation. Abnormal query detection utilizes feature space coverage analysis to detect suspect users, and the proactive confirmation module further filters the benign users in the suspect users.

1. Detection Accuracy: Can OSD effectively detect malicious queries generated by attackers and accurately distinguish between benign users and attackers?
2. Detection Efficiency: Can OSD successfully detect attacks based on a small amount of query traffic?
3. Detection Cost: How much does it cost to deploy the detection method, primarily in terms of the computational time and storage cost required for detection operations?
4. Superiority: Does OSD demonstrate more significant defense effectiveness than other defense methods?

4 Detection Framework

This paper presents a two-stage model extraction defense method OSD based on traffic monitoring. The structure of the detection framework, shown in figure 1, consists of two main modules: the abnormal query detection module based on feature space coverage and the proactive confirmation module. The abnormal query detection module uses query traffic to statistically analyze the coverage of the victim model's feature space, enabling the identification of suspicious users. The proactive confirmation module perturbs the returned results for suspicious users and determines if they are attackers based on subsequent queries. The following two subsections provide details of these two modules.

4.1 Feature Space Coverage Based Anomaly Query Detection

We first propose an abnormal query detection method to detect suspicious users in query traffic. Unlike simpler models such as classifiers and encoders, object detectors primarily require post-processing operations to filter out low-confidence bounding boxes after the output. Thus, object detection models will only provide meaningful results for in-distribution query samples. Therefore, if an attacker wants to steal an object detection model, they must enrich their query dataset to include as many features as possible. Previous studies [Li *et al.*, 2023; Liang *et al.*, 2022] on model extraction also support this conclusion. Based on this observation, we design the abnormal query detection module in this section, which analyzes the increase in coverage of query samples in the victim model's feature space.

Contrastive Encoder Training

To assess the coverage of the feature space, we initially designate a set of embedding vectors as the anchor points, denoted as A. Subsequently, we calculate the similarity between the embeddings of query samples and A as the coverage. However, due to the imbalance in feature distribution density within the victim model's feature space, employing similarity calculation for coverage estimation may not effectively distinguish between feature distribution of malicious and benign queries, as there are significant variations in the amount of features covered by different anchor points. Therefore, We employ a contrastive learning approach to remap the feature space of M_v. Contrastive learning [Chen *et al.*, 2020] employs unsupervised learning to extract similar features for similar data, uniformly mapping all sample features onto a high-dimensional sphere while keeping similar features close. Since contrastive learning focuses more on abstract semantic-level features than instance details, it exhibits more robust generalization. Therefore, if a contrastive encoder is trained based on the victim model, it can also provide feature extraction and similarity comparison effects for user query samples. We extract the backbone of the victim model and connect it to a fully connected network, forming the encoder model M_E. By fixing the parameters of the backbone network and training the additional fully connected network with a contrastive loss, we fine-tuned M_E to serve as an encoder. As shown in algorithm1, We select anchor points A from the embedding of samples in the victim dataset D_v based on cosine similarity. In our experiments, we use approximately 20k anchor points.

After identifying the anchor points in the feature space, we can further calculate the coverage of the M_v feature space by comparing the similarity between the feature vectors of the victim model passed through M_E and the anchor points in the query traffic. Figure 2 shows the anchor points and sample features of both attack and benign queries in the feature

Algorithm 1 Feature space anchor A selection

```
Require: Contrastive encoder$M_e$, the training dataset of the victim model$D_v$, anchor similarity threshold $\theta_a$
Ensure: Feature space anchor$A$
$A \leftarrow \{\}$
for each $x \in D_v$ do
    if $A == \{\}$ then
        $A \leftarrow \{x\}$
    else
        $d \leftarrow M_e(x) \cdot A^T$
        if $max(d) < \theta_a$ then
            $A \leftarrow A \cup \{x\}$
        end if
    end if
end for
return $A$
```

space after dimensionality reduction using t-SNE. It can be observed that the anchor points are evenly distributed in the feature space, and the distribution of attack traffic samples is more similar to the distribution of anchor points, indicating higher coverage. In contrast, the distribution of benign query samples is more concentrated around common features, aligning with our initial conclusion.

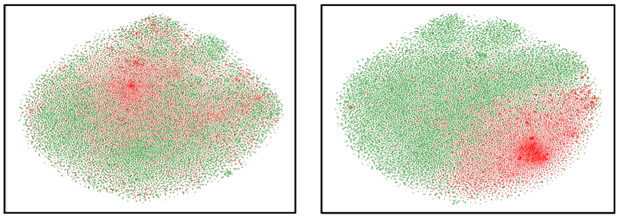
Object Feature Similarity and Coverage Calculation

For each user utilizing the services of the victim model, we extract the objects from each query sample and process them using M_E, adding it to a query sequence buffer Q. We use a one-dimensional one-hot vector C, with a length equal to the number of anchor points, to represent the coverage status of the anchor points. When the buffer Q is full, we update C based on the samples in the buffer and the anchor points in the feature space. The specific calculations are as follows:

$$C_i = \begin{cases} 0 & max(sim(A_i, M_E(Q))) > \theta_c \\ 1 & otherwise \end{cases} \tag{1}$$

Where $sim()$ denotes feature vector similarity. The feature space coverage is calculated based on the updated C:

$$Cov(q, A) = \sum_{i=0}^{c_A} \frac{C_i}{c_A} \tag{2}$$



(a) attack samples

(b) benign samples

Figure 2: Distribution of attack (a) and benign (b) embeddings after t-SNE. The green points are anchors and red points are query samples

Where q represents all samples in the current user's query traffic, $Cov_(q, A)$ is the feature coverage of the query samples, A represents anchor in the victim model's feature space, and c_A is the number of anchor points.

In calculating feature space coverage, a straightforward similarity comparison method is directly calculating feature vectors' cosine similarity. As query samples are continuously received as streaming data and OSD is separately deployed for each user, the similarity calculation must balance accuracy and efficiency to meet the efficiency requirement. Therefore, we also consider reducing the dimensionality to focus on more representative features and reduce the time and space complexity. We initially considered the widely used PCA and t-SNE dimension reduction methods, primarily Incremental PCA (IPCA). StreamingPCA [Fujiwara *et al.*, 2019] is an incremental PCA reduction method designed for streaming data. This method ensures that the dimensionality results after each update of original data maintain the consistency of historical results through geometric transformations. To minimize storage consumption, we make improvements based on StreamingPCA, discarding redundant historical embedding data during each update stage, and only the embedding and representation data from recent rounds need to be stored. This improved method further reduces the computational and storage costs required for deployment, which we denote as IPCA*. After dimension reduction using the above methods, we directly obtain low-dimensional representations of sample features, using Euclidean distance to calculate the similarity between feature vectors. The table 1 compares the resources consumed by the various feature similarity comparison methods. As shown in the table, IPCA* only requires representation storage, which is cheaper than the embedding storage cost, and consumes less time than t-SNE.

As the overall coverage of the feature space for abnormal queries increases more rapidly with the number of queries, we identify suspect users based on the growth rate of feature space coverage. After each epoch of feature coverage computation, we calculate the growth rate of feature coverage as:

$$v = \frac{Cov(q, A)}{n} \tag{3}$$

Where v represents the growth rate of feature coverage, and n is the total number of queries by the current user. When v exceeds a certain threshold θ_v, the current user is classified as a suspicious user.

Model	Cosine similarity	t-SNE	IPCA	IPCA*
Raw Sample storage	✗	✗	✗	✗
Embedding storage	✗	✓	✓	✗
Time cost	✗	✓	✗	✗
Representation storage	✗	✗	✗	✓

Table 1: Storage cost comparison between feature similarity calculation methods

4.2 Proactive Confirmation Module

While the feature space coverage-based detection can identify suspicious users in query traffic, it has a certain false-positive rate, which may lead to incorrect access blocks for some benign users. To prevent this, we introduce an additional module that further verifies user identity through a proactive confirmation module. Since model extraction attackers train substitute models based on the information returned by the victim model, if perturbations are added to the returned information, disrupting the substitute model training, the distribution of subsequent query sequences may change, distinguishing them from benign users.

In existing model extraction defense methods for classification models, perturbations are generally added to labels or confidence scores. In contrast, in object detection, model owners have more perturbation choices. Currently, FPN (Feature Pyramid Network) [Lin *et al.*, 2017] is often used to enhance detection capabilities for objects of various sizes. FPN extracts features at different scales, decoupling feature extraction for objects in different sizes. Therefore, we can disrupt the substitute model's training for extracting features in corresponding regions by applying random perturbations to the detection results for objects in a certain size range. As object models process the large regions with the highest level feature map, we choose to perturb objects in size larger than a threshold θ_l, denoted as o_l. The decrease in the detection performance of substitute models for o_l will decrease the proportion of o_l in subsequent queries. The victim model owners can confirm the attacker's identity by detecting the changes in the ratio of different-sized objects in subsequent query samples. For the i-th large object bounding box $o_{L,i}$ for sample x, the specific perturbation is applied as follows:

$$\begin{cases} o_{l,i,x} = o_{l,i,x} + rand(-\alpha, \alpha) * o_{l,i,w} \\ o_{l,i,y} = o_{l,i,y} + rand(-\alpha, \alpha) * o_{l,i,h} \end{cases} \tag{4}$$

Where $o_{l,i,x}$ and $o_{l,i,y}$ represent the x,y coordinates of $o_{l,i}$'s top-left and bottom-right point. $o_{l,i,w}$ and $o_{l,i,h}$ denotes the width and height of $o_{l,i}$. α is the perturbation magnitude, and $rand()$ is a random function within the specified range.

For model extraction attacks in data-sufficient setting [Yu *et al.*, 2020b], attackers can train a substitute model with a certain detection ability based on previously stolen information and utilize it to select query samples. Defenders can perturb larger objects in the returned information to interfere with training the substitute model's relevant network modules for larger objects. This reduces the substitute model's attention to larger objects, decreasing the proportion of larger objects in subsequent query samples. The approach may also be practical against data-free model extraction methods in scenarios, as the proactive perturbations cause the generator trained by attackers to conflict more with both the victim model and the substitute model, resulting in poorer generation performance for larger objects, which also leads to the victim model detecting fewer large object samples in subsequent query sequences. Therefore, we can effectively distinguish attackers by detecting the ratio of larger-sized objects in the overall samples. In the detection process, the proactive confirmation module perturbs N_p query samples from suspicious users, counting the proportion of o_L. N_p is calculated as follows:

$$N_p = \eta \cdot N_f \tag{5}$$

Where N_f is the number of samples that suspicious users have sent in the abnormal queries detection stage, and η is a hyperparameter. Since the model training of attackers has a certain lag, we only consider the last 20% of samples for calculation. The discrimination metric is calculated as:

$$P = \frac{(O'_l/O')}{(O_l/O)} \tag{6}$$

Where P represents a discrimination metric, O and O' represent the number of objects in all samples in two stages, O_l and O'_l represent the number of objects larger than θ_l in all samples in two stages. If P is less than the detection threshold θ_p, indicating it meets the attacker criteria, the user is identified as an attacker, and his query permission is restricted. Otherwise, the user is classified as a benign user.

5 Experiments

We design a series of experiments to address the research questions raised in the third chapter. In this section, we first introduce various settings used in the experiments, including the task scenarios, datasets, and model architectures. Subsequently, we present the experimental results regarding the research questions, followed by a discussion and analysis.

5.1 Experiment Setting

In this subsection, we briefly describe the experimental setting, including the datasets and model architectures used in the experiments. For dataset selection, we aim to evaluate the generalizability of OSD across three distinct object detection tasks: general object detection, self-driving, and aerial image detection. For each scenario, several datasets are chosen, with one dataset designated as the training dataset for the victim model, another serving as the initial sample set for the attacker, and the remaining datasets utilized to simulate queries from benign users. In the three scenarios, we employ the COCO [Lin *et al.*, 2014], nuImages [Motional, 2020], and AI-TOD [Wang *et al.*, 2021] datasets as the training datasets for the victim models. The VOC [Everingham *et al.*, 2015], BDD100K [Yu *et al.*, 2020a], and DOTA [Xia *et al.*, 2018] datasets are chosen to simulate the data available to the attacker. Additionally, the Caltech-101 [Li *et al.*, 2022] and LVIS [Gupta *et al.*, 2019] datasets, the KITTI [Geiger *et al.*, 2013] and TuSimple [Tusimple, 2022] datasets, the vhr-10 [Cheng *et al.*, 2016] and RSOD [Long *et al.*, 2017] datasets are used to simulate query data from benign users for the three tasks. Concerning model architectures, we primarily utilized YOLO series models and the Fast-RCNN [Girshick, 2015] model to represent commonly used types of object detection models. The substitute model architectures employed by the attacker were primarily the YOLOv3 [Redmon and Farhadi, 2018] and YOLOv5 [Jocher *et al.*, 2021] models.

Model	nuImages				COCO				AI-TOD			
	cosine		IPCA*		cosine		IPCA*		cosine		IPCA*	
	ACC	FPR	ACC	FPR	ACC	FPR	ACC	FPR	ACC	FPR	ACC	FPR
YOLOv3	0.893	0.1	0.893	0.05	0.857	0.15	0.929	0.10	0.857	0.00	0.929	0.00
YOLOv5	0.964	0.05	0.964	0.05	0.929	0.10	1.000	0.00	0.964	0.05	0.964	0.00
YOLOv7	0.929	0.05	1.000	0.00	0.893	0.05	0.929	0.05	0.929	0.00	0.893	0.05
Fast-RCNN	0.857	0.15	0.857	0.10	0.893	0.05	0.857	0.05	0.857	0.10	0.929	0.05

Table 2: Detection accuracy evaluation using cosine similarity and IPCA*

5.2 Detection Accuracy

In this subsection, we conduct experiments to evaluate the effectiveness of OSD. In this subsection, we conduct 12 independent experiments for each of the three scenarios and four victim model architectures, using two feature similarity comparison methods: cosine similarity and IPCA*. In each independent experiment targeting malicious users, we employ two substitute model architectures and two extraction methods: Cloudleak [Yu *et al.*, 2020b] and MEAOD [Li *et al.*, 2023], each performed four times, resulting in eight attack traffic samples. We use two datasets corresponding to the respective scenarios for benign users and randomly sample 10 times to obtain 20 benign samples. As shown in the table 2, we use OSD to evaluate the detection accuracy and false positive rate. The results indicate that OSD achieves good defensive performance with a low false positive rate, ensuring that benign users' access to the victim model service is unaffected. Comparing the two feature similarity comparison methods, the dimension-reduced method exhibits better detection performance due to its focus on essential features.

5.3 Detection Efficiency

In this subsection, we evaluate the detection efficiency of OSD by analyzing the number of historical query samples required to identify an attacker. We employ the IPCA* feature comparison and record the query counts involved in detecting an attacker and the performance of the attacker's substitute model when detected. As shown in the table 3 and figure 3, OSD achieves early detection of attackers for all three task scenarios. Simultaneously, the substitute models trained by the attackers can not replicate the functionalities of the victim models sufficiently, failing in the extraction attempts.

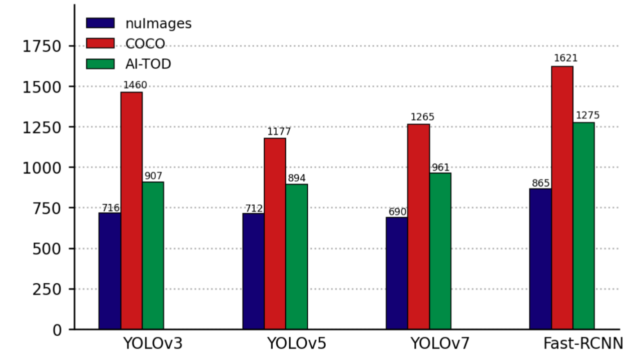


Figure 3: The number of query before detected

5.4 Detection Cost

In model extraction defense based on traffic-monitor, the computation speed of the detection method faces significant challenges due to the potentially high query speed from users. In this subsection, we conduct a statistical analysis of the detection computational costs under different victim model architectures and various feature similarity calculation methods. Throughout the experiments, we utilize an experiment platform equipped with two Nvidia GeForce RTX 3090Ti GPUs for the inference of the victim model and OSD deployment. In terms of computation time, we query 1000 times in three scenarios. As shown in table 4, the results include the time consumption statistics and the percentage increase in computation time. Compared to the native API, queries with deployed detection cause a relatively small increase in computation time, with the increment generally not exceeding 20%. Regarding storage costs, we observe memory consumption in multiple experiments because there are no tools to analyze the additional memory accurately. The additional memory consumption caused by deployment does not exceed 5%.

5.5 Comparison with Other Defense Methods

To validate the difficulty of migrating existing defense methods to the object detection model domain and to show the superiority of OSD over current works, in this subsection, we compare two representative defense mechanisms, namely OOD detector [Kariyappa and Qureshi, 2020] and HODA [Sadeghzadeh *et al.*, 2023], as they are transferred to the object detection domain. As shown in the table5, the defense methods designed for classification models do not perform effectively when transferred to the object detection model domain. Moreover, as most existing methods are not optimized

Model	nuImages	COCO	AI-TOD
YOLOv3	0.086(11.5%) 0.035	0.055(8.42%) 0.026	0.074(14.3%) 0.031
YOLOv5	0.083(11.5%) 0.025	0.043(6.71%) 0.023	0.061(11.8%) 0.027
YOLOv7	0.092(11.6%) 0.040	0.075(10.8%) 0.039	0.064(12.0%) 0.033
Fast-RCNN	0.076(12.3%) 0.021	0.046(8.30%) 0.017	0.037(7.5%) 0.019

Table 3: The performance of the substitute model when detected. The two rows for each model architecture correspond to metric: mAP50, mAP50:95

Model	nuImages			COCO		
	original(s)	defense(s)	δ	original(s)	defense(s)	δ
YOLOv3	159	181	14.6%	136	159	16.9%
YOLOv5	134	157	17.2%	115	141	22.6%
YOLOv7	115	136	18.3%	110	127	15.5%
Fast-RCNN	351	385	9.7%	298	329	10.4%

Table 4: Computation time cost of OSD deployment

Model		nuImages		COCO	
		ACC	FPR	ACC	FPR
YOLOv3	OOD	0.392	0.65	0.571	0.40
	HODA	0.429	0.70	0.679	0.45
	OSD	**0.893**	**0.05**	**0.929**	**0.10**
YOLOv5	OOD	0.429	0.55	0.464	0.5
	HODA	0.464	0.75	0.643	0.4
	OSD	**0.964**	**0.05**	**1.000**	**0.00**

Table 5: Defense effect comparison with other monitor-based defense methods

for streaming data scenarios, OSD demonstrates superiority over such existing methods.

5.6 Ablation Study

This section conducts ablation experiments to validate the effectiveness of two separate modules in OSD. The table 6 presents the experiment results using only the first stage for anomaly query detection. It can be observed that abnormal query detection has a high detection accuracy for attackers but results in a higher false positive rate for benign users. Compared with the results in previous subsections, it is evident that the abnormal query detection detects most of the attackers while the proactive confirmation module can further reduce the false positive rate for benign users.

5.7 Adaptive Attack

Adaptive attackers who understand the details of the deployed defense method may exploit the vulnerability of our defense framework to launch adaptive attacks and bypass detection mechanisms, posing a more significant threat. Against the detection framework based on feature space coverage, attackers can mitigate the increase in feature space coverage by adding background images that do not contain the foreground objects or by using query samples that only include objects of common categories. In this subsection, we test the detection effectiveness of our defense method against adaptive attacks. As shown in the table 7, attackers can partially bypass the defense after applying adaptive attacks, but there is still strong defense efficacy, which validates OSD's ability to defend against adaptive attacks.

6 Limitation

While OSD has been experimentally validated for high accuracy and detection efficiency, some limitations remain: a) The current research on model extraction attacks against object detection needs to be expanded. Therefore, this paper can only verify the defensive effects on a relatively narrow set of attack methods, potentially limiting its generalizability; b) Although blocking an attacker's access privileges can effectively disrupt the attack process, OSD assumes that the same attacker can only launch attacks through a single account and might be bypassed if the attacker gains control of multiple accounts simultaneously; c) Due to the disturbance imposed on larger-sized objects by the proactive confirmation module, it may still impact the user experience of benign users entering the second phase. We acknowledge these limitations and plan to address them in future work.

7 Conclusion

This paper addresses the research gap in model extraction defense for object detection, proposing a model extraction detection method OSD based on feature space coverage. OSD initially relies on feature coverage records to identify suspicious query traffic within query flows. Subsequently, active confirmation is performed on suspicious users to further verify the attacker's identity. This method achieves efficient and accurate model extraction detection through experimental validation, demonstrating its adaptability to the detection accuracy requirements of model extraction defense in MLaaS scenarios. Moreover, it requires a low computation cost, meeting streaming data's high query speed demands.

Model	nuImages		COCO		AI-TOD	
	ACC	FPR	ACC	FPR	ACC	FPR
YOLOv3	0.929	0.10	0.893	0.15	0.929	0.10
YOLOv5	0.893	0.15	0.857	0.20	0.893	0.15
YOLOv7	0.964	0.05	0.929	0.10	0.929	0.10
Fast-RCNN	0.785	0.250	0.821	0.20	0.857	0.15

Table 6: The ablation study results

Model	nuImages		COCO	
	ACC	FPR	ACC	FPR
YOLOv3	0.857	0.05	0.893	0.10
YOLOv5	0.853	0.05	0.857	0.10
YOLOv7	0.929	0.00	0.929	0.05
Fast-RCNN	0.785	0.10	0.714	0.05

Table 7: Adaptive attack result against OSD

Acknowledgements

This work was partly supported by the National Key Research and Development Program of China under No. 2022YFB3102100, the National Natural Science Foundation of China (Grant No. 62306093), and the Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies (Grant No. 2022B1212010005).

References

[Chen *et al.*, 2020] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

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

[Cheng *et al.*, 2016] Gong Cheng, Peicheng Zhou, and Junwei Han. Learning rotation-invariant convolutional neural networks for object detection in vhr optical remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 54(12):7405–7415, 2016.

[Cheng *et al.*, 2022] Gong Cheng, Jiabao Wang, Ke Li, Xingxing Xie, Chunbo Lang, Yanqing Yao, and Junwei Han. Anchor-free oriented proposal generator for object detection. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–11, 2022.

[Cong *et al.*, 2022] Tianshuo Cong, Xinlei He, and Yang Zhang. Sslguard: A watermarking scheme for self-supervised learning pre-trained encoders. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 579–593, 2022.

[Everingham *et al.*, 2015] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111:98–136, 2015.

[Fujiwara *et al.*, 2019] Takanori Fujiwara, Jia-Kai Chou, Shilpika Shilpika, Panpan Xu, Liu Ren, and Kwan-Liu Ma. An incremental dimensionality reduction method for visualizing streaming multidimensional data. *IEEE transactions on visualization and computer graphics*, 26(1):418–428, 2019.

[Geiger *et al.*, 2013] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[Girshick, 2015] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[Goodfellow *et al.*, 2020] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[Gupta *et al.*, 2019] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5356–5364, 2019.

[Huang *et al.*, 2023] Ziheng Huang, Boheng Li, Yan Cai, Run Wang, Shangwei Guo, Liming Fang, Jing Chen, and Lina Wang. What can discriminator do? towards box-free ownership verification of generative adversarial networks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5009–5019, 2023.

[Jia *et al.*, 2021] Hengrui Jia, Christopher A Choquette-Choo, Varun Chandrasekaran, and Nicolas Papernot. Entangled watermarks as a defense against model extraction. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 1937–1954, 2021.

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

[Jocher *et al.*, 2021] Glenn Jocher, Alex Stoken, Jirka Borovec, Ayush Chaurasia, Liu Changyu, Adam Hogan, Jan Hajek, Laurentiu Diaconu, Yonghye Kwon, Yann Defretin, et al. ultralytics/yolov5: v5. 0-yolov5-p6 1280 models, aws, supervise. ly and youtube integrations. *Zenodo*, 2021.

[Juuti *et al.*, 2019] 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.

[Kariyappa and Qureshi, 2020] Sanjay Kariyappa and Moinuddin K Qureshi. Defending against model stealing attacks with adaptive misinformation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2020.

[Kariyappa *et al.*, 2021] Sanjay Kariyappa, Atul Prakash, and Moinuddin K. Qureshi. MAZE: data-free model stealing attack using zeroth-order gradient estimation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 13814–13823. Computer Vision Foundation / IEEE, 2021.

[Law and Deng, 2020] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. *Int. J. Comput. Vis.*, 128(3):642–656, 2020.

[Li *et al.*, 2022] Fei-Fei Li, Marco Andreeto, Marc'Aurelio Ranzato, and Pietro Perona. Caltech 101, 2022.

[Li *et al.*, 2023] Zeyu Li, Chenghui Shi, Yuwen Pu, Xuhong Zhang, Yu Li, Jinbao Li, and Shouling Ji. Meaod: Model extraction attack against object detectors. *arXiv preprint arXiv:2312.14677*, 2023.

[Liang *et al.*, 2022] Siyuan Liang, Aishan Liu, Jiawei Liang, Longkang Li, Yang Bai, and Xiaochun Cao. Imitated detectors: Stealing knowledge of black-box object detectors. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 4839–4847, 2022.

[Lin *et al.*, 2014] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[Lin *et al.*, 2017] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[Lin *et al.*, 2023] Zijun Lin, Ke Xu, Chengfang Fang, Huadi Zheng, Aneez Ahmed Jaheezuddin, and Jie Shi. Quda: Query-limited data-free model extraction. In *Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security*, pages 913–924, 2023.

[Long *et al.*, 2017] Yang Long, Yiping Gong, Zhifeng Xiao, and Qing Liu. Accurate object localization in remote sensing images based on convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 55(5):2486–2498, 2017.

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

[Motional, 2020] Motional. nuimages dataset. https://www.nuscenes.org/nuimages, 2020. Accessed: 2022-10-01.

[Orekondy *et al.*, 2019] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Prediction poisoning: Towards defenses against dnn model stealing attacks. *arXiv preprint arXiv:1906.10908*, 2019.

[Peng *et al.*, 2022] Zirui Peng, Shaofeng Li, Guoxing Chen, Cheng Zhang, Haojin Zhu, and Minhui Xue. Fingerprinting deep neural networks globally via universal adversarial perturbations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13430–13439, 2022.

[Qiao *et al.*, 2023] Tong Qiao, Yuyan Ma, Ning Zheng, Hanzhou Wu, Yanli Chen, Ming Xu, and Xiangyang Luo. A novel model watermarking for protecting generative adversarial network. *Computers & Security*, 127:103102, 2023.

[Redmon and Farhadi, 2018] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement, 2018.

[Sadeghzadeh *et al.*, 2023] Amir Mahdi Sadeghzadeh, Amir Mohammad Sobhanian, Faezeh Dehghan, and Rasool Jalili. Hoda: Hardness-oriented detection of model extraction attacks. *IEEE Transactions on Information Forensics and Security*, 2023.

[Sanyal *et al.*, 2022] 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.

[Shah *et al.*, 2023] Harshit Shah, G Aravindhan, Pavan Kulkarni, Yuvaraj Govindarajulu, and Manojkumar Parmar. Data-free model extraction attacks in the context of object detection. In *International Conference on Computer Vision Systems*, pages 391–402. Springer, 2023.

[Tusimple, 2022] Tusimple. Tusimple-benchmark. https://github.com/TuSimple/tusimple-benchmark, 2022. Accessed: 2023-11-01.

[Wang *et al.*, 2021] Jinwang Wang, Wen Yang, Haowen Guo, Ruixiang Zhang, and Gui-Song Xia. Tiny object detection in aerial images. In *2020 25th international conference on pattern recognition (ICPR)*, pages 3791–3798. IEEE, 2021.

[Xia *et al.*, 2018] Gui-Song Xia, Xiang Bai, Jian Ding, Zhen Zhu, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang. Dota: A large-scale dataset for object detection in aerial images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3974–3983, 2018.

[Yu *et al.*, 2020a] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2636–2645, 2020.

[Yu *et al.*, 2020b] 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.

[Zhang *et al.*, 2023] Haitian Zhang, Guang Hua, Xinya Wang, Hao Jiang, and Wen Yang. Categorical inference poisoning: Verifiable defense against black-box dnn model stealing without constraining surrogate data and query times. *IEEE Transactions on Information Forensics and Security*, 18:1473–1486, 2023.

[Zhou *et al.*, 2019] Xingyi Zhou, Jiacheng Zhuo, and Philipp Krahenbuhl. Bottom-up object detection by grouping extreme and center points. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 850–859, 2019.