D-DAE: Defense-Penetrating Model Extraction Attacks

Yanjiao Chen[1], Rui Guan[2], Xueluan Gong[1]*, Jianshuo Dong[3], and Meng Xue[1]
[1]School of Computer Science, Wuhan University, China
[2]School of Mathematics and Statistics, Wuhan University, China
[3]School of Cyber Science and Engineering, Wuhan University, China
*Corresponding author
{ruiguan, jianshuo.dong, xueluangong, xuemeng}@whu.edu.cn, chenyanjiao@zju.edu.cn

2023 IEEE Symposium on Security and Privacy (SP) | 978-1-6654-9336-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/SP46215.2023.10179406

***Abstract*—Recent studies show that machine learning models are vulnerable to *model extraction attacks*, where the adversary builds a substitute model that achieves almost the same performance of a black-box victim model simply via querying the victim model. To defend against such attacks, a series of methods have been proposed to disrupt the query results before returning them to potential attackers, greatly degrading the performance of existing model extraction attacks.**

In this paper, we make the first attempt to develop a defense-penetrating model extraction attack framework, named D-DAE, which aims to break disruption-based defenses. The linchpins of D-DAE are the design of two modules, i.e., *disruption detection* and *disruption recovery*, which can be integrated with generic model extraction attacks. More specifically, after obtaining query results from the victim model, the disruption detection module infers the defense mechanism adopted by the defender. We design a meta-learning-based disruption detection algorithm for learning the fundamental differences between the distributions of disrupted and undisrupted query results. The algorithm features a good generalization property even if we have no access to the original training dataset of the victim model. Given the detected defense mechanism, the disruption recovery module tries to restore a clean query result from the disrupted query result with well-designed generative models. Our extensive evaluations on MNIST, FashionMNIST, CIFAR-10, GTSRB, and ImageNette datasets demonstrate that D-DAE can enhance the substitute model accuracy of the existing model extraction attacks by as much as 82.24% in the face of 4 state-of-the-art defenses and combinations of multiple defenses. We also verify the effectiveness of D-DAE in penetrating unknown defenses in real-world APIs hosted by Microsoft Azure and Face++.

***Index Terms*—Model extraction attacks, disruption-based defenses, meta-learning.**

I. Introduction

Deep neural networks (DNNs) are being widely adopted in various areas. However, training well-performed DNN models is time-consuming, resource-consuming, and money-consuming, thus model owners are keen to protect the privacy of their DNN models.

Unfortunately, recent research has revealed that DNN models are vulnerable to *model extraction attacks* [40], where an attacker can establish a substitute model with almost the same functionalities as the target victim model via simply querying the victim model. The potential consequences of model extraction attacks may be serious. On the one hand, many service providers have launched cloud-based Machine Learning as a Service (MLaaS) to monetize their DNN models (e.g., Amazon AWS[1], Microsoft Azure[2], Google Cloud[3], and BigML[4]). Users pay service providers to query the model via Application Programming Interfaces (API). An adversary may steal the cloud-based model at a much lower cost than the expenditure of the service provider on data collection and training. By opening a competing MLaaS platform, the adversary violates intellectual property and causes financial losses to the service provider. On the other hand, model extraction attacks can serve as the springboard for other attacks. For instance, many adversarial example attacks require the gradient information of the victim model, which is not available in the black-box settings. In this case, the adversary can construct a substitute model via model extraction attacks and create adversarial examples based on the white-box substitute model. Due to the transferability property, these adversarial examples are also effective on the victim model.

Extensive research efforts have been made to prevent model extraction attacks. Mainstream defense strategies include detection [19], [22] and disruption [28], [31]. The intuition behind detection-based defense strategies is that malicious queries exhibit abnormal statistical distributions that are different from distributions of normal queries. Hypothesis tests can be conducted to identify out-of-distribution (OOD) queries. If a query sequence is considered anomalous, the defender may reject the user requests or disrupt the query results. Disruption-based defense strategies thwart the attempt of model extraction by disrupting all or the identified OOD query results. For instance, differential privacy has been used to add noises to query results with rigorous guaranteed privacy [46]. It is shown that with disruption-based defense, the accuracy of the substitute model drops from 87.3% to 35.7% [31]. Real-world MLaaS providers have adopted these defense strategies. For instance, Microsoft published a guidance on defending AI/ML-based products/services from model extraction attacks

D-Day is the codename for Allied invasion of Normandy in WWII.

[1]https://aws.amazon.com/cn/machine-learning/
[2]https://azure.microsoft.com/en-us/services/machine-learning/
[3]https://cloud.google.com/
[4]https://cloudacademy.com/blog/bigml-machine-learning/

DOI 10.1109/SP46215.2023.00025

by "return(ing) rounded confidence values" or obfuscating the prediction results [3].

To our knowledge, there are no existing works exploring the possibility of model extraction attacks against defended victim models. Inspired by this research gap, we propose a defense-penetrating model extraction framework, named D-DAE, which aims at wrecking disruption-based defenses. The main idea is to detect and recover disrupted query results. As shown in Fig. 1, generic model extraction attacks are usually conducted in three steps. To start with, the substitute model is initiated as empty or as a pre-trained model. Then, natural or synthetic samples are selected to query the victim model and obtain the query results. Finally, the substitute model is retrained based on the query results. The query and the retraining processes are iterated to reach ideal performance. Nevertheless, if the query results are disrupted by the defender, the retraining efforts may be futile. To address this problem, we design two general modules, i.e., *disruption detection* and *disruption recovery*, which can fit into generic model extraction attacks before the retraining phase.

To materialize our goal of defense-penetrating model extraction attacks, we are facing the following challenges.

- ***How to detect the specific defense mechanism adopted by the defender?***

In most cases, the specific defense mechanism adopted by the defender is unknown. In addition, the defender may choose to disrupt only suspicious queries but not all queries. To tackle this problem, we utilize meta-learning to differentiate the distributions of undisrupted query results and disrupted query results of different defenses. As we have no access to the original training dataset of the victim model, we leverage the generalization of meta-learning models to infer the intrinsic property of a particular defense mechanism. More specifically, we use many public datasets to train a pool of shadow models and apply each defense method to multiple shadow models. Then, for each defense method, we train a binary meta-classifier with the query results from unprotected shadow models and shadow models protected by this defense method. This trained meta-classifier is used to predict whether the query results are disrupted by the corresponding defense method. Finally, the prediction results of all meta-classifiers are combined to determine which defense method the defender adopts.

- ***How to recover the disrupted query results?***

Even if we detect which defense is adopted by the defender, the exact amount of noise added is uncertain since most defense methods randomize the noise to avoid being bypassed. It is also ineffective to subtract expected mean noise, which still yields inaccurate query results. To address this challenge, we develop a novel disruption recovery mechanism, which reconstructs clean query results using generative models. The generative model is carefully trained with disrupted query results as inputs and undisrupted query results as outputs for each defense method.

We have conducted extensive experiments to evaluate the enhancement of D-DAE under 4 types of defenses and hybrid defenses in terms of 3 state-of-the-art model extraction attacks. Experiment results on MNIST, FashionMNIST, CIFAR-10, GTSRB, and ImageNette have demonstrated that D-DAE can increase the performance of the substitute model by as much as 80% under the defense. We also verify the real-world impact of D-DAE on a Microsoft Azure API [2] and the Face++ API [1] with absolutely unknown defense strategies adopted by the service providers, which shows that D-DAE enhances existing attacks by as much as 18.93%. This indicates that D-DAE may help penetrate the defense built by the service providers, which could motivate future research efforts on more powerful defense strategies.

The contributions of this paper are summarized as follows.

- We have proposed a defense-penetrating model extraction attack framework D-DAE, as shown in Fig. 1, which can be integrated with generic model extraction attacks to resist disruption-based defenses.
- We have designed novel disruption detection and disruption recovery algorithms, which can differentiate the defense method adopted by the defender and restore clean query results for retraining the substitute model.
- We have conducted extensive experiments to verify that D-DAE can enhance the accuracy of substitute models under various defense scenarios.

The source code of D-DAE can be obtained at GitHub.

II. Preliminaries and Related Work

In this section, we give a brief summary of state-of-the-art model extraction attacks and defenses.

A. Model Extraction Attacks

Model extraction attacks aim to construct a substitute model ($\mathcal{F}_A$) that closely approximates the functionalities or the parameters of the target victim model ($\mathcal{F}_V$). In this sense, model extraction attacks are all black-box attacks, where the adversary is unaware of the victim model's structure, parameters, and hyperparameters. Nevertheless, the adversary is able to query the victim model, e.g., through an API, and obtain the results. The query result may be the confidence score or simply the class label. It is normally assumed that the adversary knows the classification task of the victim model, e.g., face recognition. Furthermore, the adversary may or may not have a subset of the original training dataset of the victim model but have full access to public datasets that may have a similar distribution of the original training dataset of the victim model.

There are two main metrics to evaluate the success of model extraction attacks: *accuracy* and *agreement*. *Accuracy* measures the fraction of correctly-labeled test samples by the substitute model. If the adversary's objective is to construct a well-performed classification model, high accuracy is desirable. For example, a malicious small-sized company may steal a valuable model trained with massive training data and computing resources by a large company and launch its

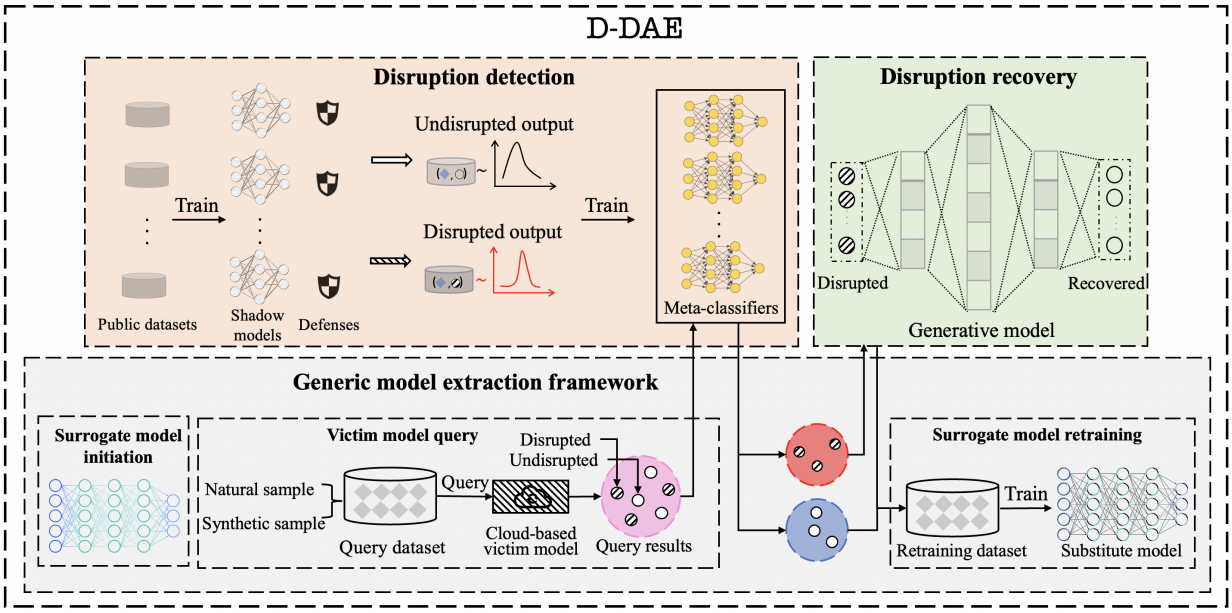


Fig. 1. The architecture of D-DAE.

own prediction service at a low cost. *Agreement* measures the fraction of test samples whose prediction label given by the substitute model is the same as that given by the victim model. A high agreement is ideal if the purpose of the adversary is to mount further attacks against the black-box victim model. For instance, one way to create adversarial examples for a black-box victim model is to construct a highly-similar substitute model. Due to transferability property, the adversarial examples of the (white-box) substitute model are also effective to the victim model.

Apart from the effectiveness of the extracted substitute model, another goal of model extraction algorithm design is to minimize the number of queries to the victim model. The intuition behind this goal is two-fold. The first is to reduce the query cost. The second and more important is to evade defense strategies adopted by the owner of the victim model. For example, as many defense algorithms detect whether a query sequence is malicious or not based on statistical deviations, it is easier to discover abnormalities in a long query sequence than a short query sequence.

As shown in Fig. 1, a generic model extraction attack consists of three steps: substitute model initiation, victim model query, and substitute model retraining.

- *Substitute model initiation*. The adversary first needs to determine the architecture of the substitute model, including the structure (e.g., LeNet [26], ResNet [17], VGG [37]), the number of layers, the number of neurons, and the activation function. Choosing the most appropriate architecture may require some expert knowledge. It has been shown that it is more effective to extract a victim model with a more complicated model, e.g., extracting a VGG with a ResNet [14]. The adversary may pre-train the substitute model with an available dataset or initiate an empty substitute model, and the former is proved to achieve better performance [30].
- *Victim model query*. Victim model query is key to model extraction attacks. There are three kinds of victim model query: natural sample query, synthetic sample query, and data-free query. *Natural sample query* selects representative natural samples from public datasets as query samples. *Synthetic sample query* constructs synthetic samples (usually adversarial examples) based on natural samples as query samples. *Data-free query* generates samples from noises using generative models as query samples.
- *Substitute model retraining*. After obtaining query results, the adversary retrains the substitute model based on the results. Victim model query and substitute model retraining are performed iteratively until the model converges or the number of queries hits the limit.

Next, we introduce the three query methods in more detail.

Natural sample query. The key in natural sample query is to choose the *best* natural samples from a set of candidate samples. A commonly-used selection approach is *active learning*, which adaptively samples public datasets for representative samples [5]. CopyCat [8] used semi-supervised learning to sift through natural non-labeled samples. KnockoffNet [30] used reinforcement learning, in which the reward function updates a hierarchical-structure learning strategy according to gradient bandit algorithm [39]. ActiveThief [32] used active learning subset selection strategies to iteratively pick informative samples.

Synthetic sample query. Synthetic sample query starts with a few natural samples possibly acquired from the training dataset of the victim model or the public dataset with similar (usually not the same) distribution to the original training dataset of the victim model. Then, the adversary crafts synthetic samples based on natural samples as query samples. Jacobian-based augmentation is commonly used in

existing works [40], [19], [33], [45], which leverages query feedback from the victim model to guide the sample synthesis process. The intuition is to compute the Jacobian matrix of the substitute model to craft adversarial examples, which are expected to be close to the decision boundary of the victim model. These adversarial examples are *hard* to learn and may solicit more information from the victim model through a query. Given a sample x, a synthetic sample is constructed as

$$x \leftarrow x - \lambda \bigtriangledown_x f(\mathcal{L}(x, F_A(x))), \tag{1}$$

where λ is the hyperparameter that adjusts the learning rate, $\mathcal{L}$ is the loss function, $\bigtriangledown$ is the gradient, and the function $f(\cdot)$ is designed differently in existing works.

Papernot *et al.* [33] first proposed to use Fast Gradient Sign Method (FSGM) [16], a simple and efficient Jacobian-based augmentation algorithm, to create adversarial examples. Jutti *et al.* [19] also used FSGM, and proposed to use *cross-validation search* to optimize hyperparameters in training. FeatureFool [45] used C&W attack [4] and Feature Adversary Attack [35] to generate adversarial samples and introduced triplet loss in the loss function to accelerate convergence. Note that natural samples and synthetic samples can be combined as query samples.

Data-free query. In some cases, the victim model may be trained with a rare training dataset where no similar public datasets exist, e.g., a breast cancer detection model based on Magnetic Resonance Imaging (MRI) samples. Inspired by the success of data-free knowledge distillation methods [18], researchers have proposed data-free model extraction attacks [41], [21], which use generative models to create query samples. However, a major disadvantage of data-free methods is the need for a large number of queries to obtain a well-performed substitute model.

The generated query samples are expected to maximize the prediction gap between the substitute model and the victim model such that the substitute model can learn more information from the query results of the generated samples.

$$\min_{F_A} \max_{G} \mathbb{E}_{z\sim\mathcal{N}(0,1)}[\mathcal{L}(F_V(G(z)), F_A(G(z)))], \tag{2}$$

where $z \sim \mathcal{N}(0,1)$ is the noise drawn from standard Gaussian distribution, $\mathcal{L}$ is the loss function, and $G(\cdot)$ is the generative model. According to (2), to update the parameters of the generative model based on gradient descent, we need $\bigtriangledown_{\theta_{F_A}}\mathcal{L}$, which can be computed by backpropagation, and $\bigtriangledown_{\theta_G}\mathcal{L}$, which can be approximated by *zero-order gradient estimation* [13]. MAZE [21] proposed the first data-free model extraction, which used KL-divergence as the loss function. DFME [41], another data-free model extraction method, is based on Generative Adversarial Networks (GAN) [15], using l_1 norm loss to prevent vanishing gradient problem.

B. Defenses Against Model Extraction Attacks

When defending model extraction attacks, the defender faces two conflicting goals: deteriorating malicious queries and boosting benign queries. More specifically, the defender wants to thwart the efforts of the adversary for model extraction at the minimum impact on the normal service to benign users. Some existing works target at *accuracy-preserving defenses* [28], which ensures the prediction accuracy for benign users and some focus on *accuracy-constrained defenses* [22], [20], [31], which make a trade-off between the victim model accuracy (capability) and the substitute model accuracy (privacy). Computational costs and compatibility with the victim model architecture are other factors to consider.

Existing defense methods follow two lines: detection and disruption. Detection-based methods try to judge whether a query sequence is malicious or not without distorting the query results. If a query sequence is regarded as malicious, the defender may choose to disrupt the query result or deny the service to the malicious user. If the detection accuracy is 100%, the detection-based methods are accuracy-preserving defenses. Disruption-based methods perturb the query results of all or some queries.

Detection-based defenses. The intuition behind detection-based defenses is that malicious and benign query samples have different distributions. For example, PRADA [19] assumes that benign queries fit normal distributions and adversarial queries do not. *Shapiro-Wilk test* [36] is used for normality test.

$$W(D) = \frac{\left(\sum_{i=1}^{n} a_i x_{(i)}\right)^2}{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}, D = \{x_1, \ldots x_n\}, \tag{3}$$

where $W(\cdot)$ is the *Shapiro-Wilk test* statistics, D is the dataset, a_i are constants, and $x_{(i)}$ is the i-th order in D.

Kariyappa *et al.* [22] proposed Adaptive Misinformation (AM) based on the assumption that existing attacks, no matter using synthetic or natural data, inevitably generate out-of-distribution (OOD) queries, which may be detected by an OOD detector and the corresponding query results are disrupted by a misinformation function. As an extension, Ensemble of Diverse Models (EDM) [20] used ensemble learning to recognize OOD queries and disrupted the query results.

Disruption-based Defenses. Disruption-based defenses disrupt the query results in different ways. Lee *et al.* [28] proposed Reverse Sigmoid (RS) as an additional layer attached to the victim model with random softmax probability that is difficult to inverse but preserves top-1 accuracy of the victim model. Prediction Poisoning (PP) [31] maximized the angle deviation between the original and the perturbed gradient as

$$\eta^* = \max_{\eta} \left\| \frac{\eta}{\|\eta\|_2} - \frac{\mu}{\|\mu\|_2} \right\|_2^2, \tag{4}$$

where η^* is the misinformation added to the prediction probability vector, $\eta = -\bigtriangledown_\theta \mathcal{L}(F_A(x;\theta), y)$ is the direction in which noise is added, $\mathcal{L}$ is the loss function, and μ is the gradient of the original probability vector.

Differential privacy (DP) is another commonly-used disruption method that provides a formal guarantee on privacy leakage under a privacy budget ϵ [10]. By disrupting weights

of deep learning models during training, DP can control privacy leakage in the inference stage and defend against model inversion attacks [12] as well as membership inference attacks [6]. Recently, DP has been applied to thwart model extraction attacks by obfuscating the prediction results to prevent attackers from extracting black-box models [44], [46]. Specifically, Zheng et al. [46] proposed to use boundary differential privacy (ϵ-BDP) to protect against model extraction attacks by perturbing the prediction near the decision boundary. Yan et al. [44] proposed a monitoring-based differential privacy mechanism to protect the victim model by dynamically adjusting the added noise based on the feedback of model extraction status.

III. Threat Model

We define the threat model in terms of the cans/knows and cannots/unknows of the adversary.

We assume that the adversary has the following capabilities.

- *Access to public datasets*. The adversary can collect datasets that are in-distribution or out-of-distribution of the original training dataset of the victim model. For example, if the victim model is trained with MNIST [27] for optical character recognition tasks, the public dataset of EMNIST Letters [7] is in-distribution and the public dataset ImageNet [9] is out-of-distribution.
- *Query the victim model*. The adversary can query the victim model, e.g., via API, and obtain the query results. We assume that the query results contain the confidence score, which is available in most MLaaS. The adversary has a query budget that limits the maximum number of queries that can be made. We assume that the owner of the victim model adopts disruption-based but not detection-based defense strategies. More specifically, the defender may disrupt some or all prediction results in an accuracy-preserving or accuracy-constrained manner.
- *Reconstruct defense strategies*. The adversary is capable of reconstructing a series of defense strategies, denoted by $\mathcal{T}_V = \{\mathcal{T}_A^1, \ldots, \mathcal{T}_A^K\}$, where $\mathcal{T}_A^k, k \in \{1, \ldots K\}$ is a defense strategy. We assume that the defense strategy adopted by the defender is in $\mathcal{T}_V$. We will show in experiments that D-DAE is effective even if the defense strategy adopted by the defender is not in $\mathcal{T}_V$.

We assume that the adversary has the following limitations.

- *No knowledge of the victim model*. The adversary does not have any knowledge of the victim model, including the architecture, parameters, and hyperparameters.
- *No original training dataset*. The adversary does not have access to original training samples of the original victim model, which are too difficult or too expensive to acquire.
- *No knowledge of the specific defense strategy*. The adversary does not know the exact defense strategy adopted by the defender, and the defender may choose a subset of query results to disrupt.

The adversary's goal is to establish a well-performed substitute model with a high agreement with the ground truth using a small number of queries to the victim model that is protected by an unknown defense strategy.

TABLE I
Parameters of D-DAE.

Parm.	Description
K	The number of defenses used to train the meta-classifier
$\mathcal{T}_V$	The set of K defense strategies use to train the meta-classifier
$\mathcal{T}_A^k$	The k-th defense strategy
M	The number of public datasets used to train shadow models
$\mathcal{D}_m$	The m-th public dataset
f_m	Unprotected shadow model trained on $\mathcal{D}_m$
f_m^k	Protected version of f_m disrupted by $\mathcal{T}_A^k$
$\mathcal{M}_k$	The meta-classifier to detect defense strategy $\mathcal{T}_A^k$
$\mathcal{F}_V$	The victim model
$\mathcal{F}_A$	The substitute model
T	The threshold for defense strategy detection
$\mathcal{G}_k$	The generator to recover query results disrupted by defense $\mathcal{T}_A^k$

IV. D-DAE: Detailed Construction

Compared with generic model extraction attacks, D-DAE develops two extra modules between the *victim model query* module and the *substitute model retraining* module in order to detect and restore disrupted query results, as shown in Fig. 1. The first module is *disruption detection*, which leverages meta-classifiers to identify the defense strategy adopted by the defender. The second module is *disruption recovery*, which attempts to recover correct query results from disrupted query results. These two modules can fit into generic model extraction attack frameworks. In this section, we focus on the detailed description of the *disruption detection* module and the *disruption recovery* module.

A. Disruption Detection

Our key idea is to identify the defense strategies adopted by the defender by exploiting distinctive distributions of disrupted query results of different defense strategies. We first build a pool of unprotected classification models $\{f_1, f_2, \cdots, f_M\}$[5] on different public datasets $\{\mathcal{D}_1, \mathcal{D}_2, \cdots, \mathcal{D}_M\}$ with diverse architectures and optimization procedures. It is ideal that these shadow models have similar architecture to the victim model, which is, however, infeasible in the black-box setting. Therefore, we diversify the architectures of the shadow models to achieve good generalization. Then, we apply defenses to these shadow models. Since we have K defense methods and M sets of shadow models, we can apply a certain defense method to multiple shadow models or apply multiple defense methods to the same shadow models. The objective is to obtain sufficient pairs of protected and unprotected shadow models for each defense.

Let f_m^k denote the counterpart version of f_m protected by the $\mathcal{T}_A^k$. It is expected that $f_m(x_m), x_m \in \mathcal{D}_m$ embody the distribution of undisrupted query results, and $f_m^k(x_m), x_m \in \mathcal{D}_m$ embody the distribution of query results using defense strategy $\mathcal{T}_A^k$. Based on this intuition, we train K binary meta-classifier

[5]We use different architectures and optimization procedures to train multiple models on each public dataset. Therefore, f_m represents a set of unprotected models trained on $\mathcal{D}_m$.

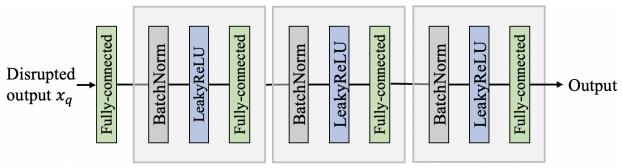


Fig. 2. The architecture of the generator in D-DAE.

Algorithm 1 Defense-penetrating model extraction attack framework of D-DAE

Require: Interface of the victim model $\mathcal{F}_V$, public datasets $\mathcal{D}_1, \cdots, \mathcal{D}_M$, the set of defense strategies $\mathcal{T}_A^1, \cdots, \mathcal{T}_A^K$.
Ensure: Substitute model $\mathcal{F}_A$.

// Preparation.
Train shadow models $f_1, \ldots, f_M$ with the public datasets $\mathcal{D}_1, \cdots, \mathcal{D}_M$.
for $k \in [1, K]$ **do**
  Apply $\mathcal{T}_A^k$ to $f_m, m \in [1, M]$.
  Train a binary meta-classifier $\mathcal{M}_k$ with the inputs $(x_m, f_m^k(x_m)), \forall m$ as positive samples and $(x_m, f_m(x_m)), \forall m$ as negative samples.
  Train a generative model $\mathcal{G}_k$ with $f_m^k(x_m), \forall m$ as inputs and $f_m(x_m), \forall m$ as outputs.
end for
// Substitute model initiation.
Initiate an empty or a pre-trained substitute model F_A.
while F_A is not converged & the query budget is not zero **do**
  // Victim model query.
  Generate a query sample x_q.
  Query the victim model F_V and obtain the query result $F_V(x_q)$.
  // Disrupt detection.
  Feed $(x_q, \mathcal{F}_V(x_q))$ into $\mathcal{M}_1, \cdots, \mathcal{M}^K$.
  if There exist non-zero prediction results **then**
    Select the prediction with the highest confidence score, denoted by k^*.
    // Disruption recovery.
    Feed $F_V(x_q)$ into $\mathcal{G}_{k^*}$ to obtain $F_V(x_q) \leftarrow \mathcal{G}_{k^*}(F_V(x_q))$.
  end if
  // Substitute model retraining.
  Use x_q and the restored $F_V(x_q)$ to retrain the substitute model F_A.
end while

$\mathcal{M}_1, \cdots, \mathcal{M}_K$, where $\mathcal{M}_k$ is used to detect whether the query results are protected by the k-th defense strategy. We train $\mathcal{M}_k$ with input $(x_m, f_m^k(x_m)), \forall m$ as positive samples (i.e., the label is 1) and $(x_m, f_m(x_m)), \forall m$ as negative samples (i.e., the label is 0). Since the meta-classifier $\mathcal{M}_k$ is trained with samples regarding different classification tasks, we expect $\mathcal{M}_k$ to learn the intrinsic differences in distributions of disrupted and undisrupted query results by defense $\mathcal{T}_A^k$ regardless of the specific classification task. In this way, the meta-classifiers can generalize to the victim model whose training dataset is unknown to the adversary.

The trained meta-classifiers are utilized to identify the defense strategy of the victim model. Let $x_q \in \mathcal{D}_q$ denote a query sample from the query dataset, and $\mathcal{F}_V(x_q)$ is the query result. We feed $(x_q, \mathcal{F}_V(x_q))$ into meta-classifiers $\mathcal{M}_1, \cdots, \mathcal{M}_K$ and get the prediction results. If all prediction results are below a threshold T (e.g., 0.5), we regard the sample as undisrupted; otherwise, we compare the confidence score of the prediction results that are above threshold T and choose the one with the highest confidence score as the predicted defense strategy. According to our experiment, in most cases, the meta-classifier of the true defense strategy will yield a confidence score of more than 90%, which indicates that we can well differentiate the adopted defense strategy with high confidence.

B. Disruption Recovery

If it is predicted that the query result of x_q is undisrupted, the query result will be directly used to retrain the substitute model. Otherwise, we restore the clean query result from the disrupted query result based on the detected defense strategy. Note that defense detection may be performed at different time granularity (e.g., just once, periodically, or per query). In the most exact case, if the defender randomly alternates among multiple defenses, the attacker may conduct defense detection on each query result and then recover each disrupted query result depending on the detection results.

To recover the query result protected by a certain defense strategy $\mathcal{T}_A^k$, we build a generative model $\mathcal{G}_k$ that takes the disrupted query result as input $\mathcal{F}_V(x_q)$ and outputs the undisrupted query result $\mathcal{F}(x_q)$. To train each generative model $\mathcal{G}_k$, we leverage the shadow models prepared for disruption detection in the previous step. More specifically, each training sample has the form of $\big(f_m^k(x_m), f_m(x_m)\big), \forall m$, where $f_m^k(x_m)$ is the protected query result by defense strategy $\mathcal{T}_A^k$ for shadow model f_m and $f_m(x_m)$ is the corresponding clean query result. In this way, $\mathcal{G}_k$ learns a mapping from disrupted results to undisrupted ones such that $\mathcal{G}_k(f_m^k)$ has the same distribution as f_m.

There are two major families of generative models, i.e., Variational Autoencoders (VAEs) [23] and Generative Adversarial Networks (GANs) [15]. Autoencoders are usually used for dimensionality reduction, i.e., compress the data samples into informative representations such that the dimension of the output is smaller than that of the input, and GANs are usually used for inverse transformation, i.e., generate samples of a more complex distribution based on random noises such that the dimension of the output is larger than that of the input. Nevertheless, in our disruption discovery, the output clean query result has the same dimensionality as the input disrupted query result. Therefore, neither VAE nor GAN is suitable to realize the goal of disruption discovery.

As shown in Fig. 2, we build a generator with carefully-designed architecture explained as follows. The generator $\mathcal{G}$ is formed by multiple blocks that consist of BatchNorm, LeakyReLU, and fully-connected layers. In each block, the

BatchNorm layer first normalizes the feature vectors to avoid problems of convergence difficulty and mode collapse. Then, LeakyReLU is used instead of ReLU as the activation function to mitigate vanishing gradient problem [29]. Finally, the fully-connected layer produces refined query results with the same dimension as the disrupted query results. Our proposed generator is lightweight with all necessary components to recover the clean query results.

We train the generator to minimize the discrepancy between the recovered query result $\mathcal{G}_k(f_m^k(x_m))$ and the real clean query result $f_m(x_m)$, characterized by a loss function. Commonly-used loss functions include MSE, MAE, and KLD.

Mean Squares Error (MSE) Loss. MSE is a simple loss function that has fast convergence.

$$\mathcal{L}_{MSE} = \frac{1}{|\sum_m \mathcal{D}_m|} \sum_{x_m \in \mathcal{D}_m, \forall m} [\mathcal{G}_k(f_m^k(x_m)) - f_m(x_m)]^2. \tag{5}$$

Mean Absolute Errors (MAE) Loss. MAE has a slower and less significant gradient decay than MSE, but the derivative of MAE is not unique at point 0, which may result in non-convergence.

$$\mathcal{L}_{MAE} = \frac{1}{|\sum_m \mathcal{D}_m|} \sum_{x_m \in \mathcal{D}_m, \forall m} |\mathcal{G}_k(f_m^k(x_m)) - f_m(x_m)|. \tag{6}$$

Kullback-Leibler Divergence (KLD) Loss. KLD measures the similarity between two probability distributions and is heavily used in knowledge distillation [18]. LD loss may suffer from *vanishing gradients* [11].

$$\mathcal{L}_{KLD} = \frac{1}{|\sum_m \mathcal{D}_m|} \sum_{x_m \in \mathcal{D}_m, \forall m} f_m(x_m) \log \frac{f_m(x_m)}{\mathcal{G}_k(f_m^k(x_m))}. \tag{7}$$

We summarize the framework of D-DAE in Algorithm 1 and the parameters of D-DAE in Table I.

V. Experiment Setup

A. Datasets and Victim Models

We conduct experiments on five commonly-used datasets, i.e., MNIST [27], FashionMNIST [42], CIFAR-10 [24], GTSRB [38], and ImageNette[6] [25]. We utilize LeNet [26], VGG16-bn [37], and Resnet-34 [17] to train victim models on these datasets, respectively. The details of datasets and victim models are listed in Table II.

B. State-of-the-Art Attacks

We consider three state-of-the-art model extraction attacks to be enhanced by D-DAE.

JBDA [33]. JBDA uses Jacobian-based augmentation to generate synthetic samples based on a small set of unlabeled *seeds* that are assumed to have a similar distribution to the training samples of the victim model. In our implementation, the size of the seed set is 100 for MNIST and FashionMNIST, and 50 for CIFAR-10, GTSRB, and ImageNette. Other parameters are consistent with those in the original paper.

[6]ImageNette is a representative subset of ImageNet. The original ImageNet contains more than 20,000 categories and more than 14 million images, which cannot be processed by our capacity-limited servers. The baselines, i.e., JBDN [33], KnockoffNet [30], and ActiveThief [32], were all evaluated on lower-resolution datasets than ImageNette. Interested readers can test D-DAE on ImageNet using our released code.

TABLE II
Datasets and victim models.

Dataset	# classes	# samples	structure	model accuracy
MNIST	10	60,000	LeNet	99.40%
FashionMNIST	10	60,000	LeNet	92.04%
CIFAR-10	10	60,000	VGG16-bn	93.43%
GTSRB	43	39,209	VGG16-bn	93.90%
ImageNette	10	9,469	ResNet34	93.90%

KnockoffNet [39]. KnockoffNet uses reinforcement learning to select natural samples from a different but similar dataset from the training dataset of the victim model to query the victim model. For the MNIST, FashionMNIST, CIFAR-10, GTSRB, and ImageNette victim model, we use EMNISTLetters, EMNIST, CIFAR-100, GTSRB, and ImageNette as the query datasets, respectively. Note that the training samples and the querying samples from GTSRB and ImageNette have no overlap.

ActiveThief [32]. ActiveThief used active learning to select natural samples from public datasets. The query dataset begins with 500 initial seed samples and an additional 500 query samples are selected by active learning in each iteration until the query budget is hit. For each iteration, we use the k-center strategy to select query samples. The selection of the public datasets is the same as KnockoffNet.

C. Implementation of D-DAE

Note that D-DAE can be used in conjunction with any existing model extraction attack. We materialize D-DAE in the above three model extraction attack frameworks to obtain their enhanced version D-JBDA, D-KnockoffNet, and D-ActiveThief.

Shadow model. In the experiments, we adopt the CNN model structure for the shadow models. In each shadow model, there are four convolutional blocks after the input layer, and each block consists of 1 convolutional layer (kernel size=$3*3$, stride=1) followed by ReLU and maxpool layers. Filters in convolutional layers are decided by the input image size. The output of the maxpool layers is passed through a fully connected layer, a dropout layer, and a softmax layer to obtain the prediction results.

Meta-classifier. To achieve good generalization of the meta-classifiers to accurately detect the specific defense method adopted by the defender, we need to train a large number of shadow models. The default number of shadow models in the experiments is 256 for MNIST, 1,024 for FashionMNIST, 2,048 for CIFAR-10, GTSRB, and ImageNette. We use a two-layer fully-connected neural network [43] to train each meta-classifier. We use the Adam optimizer with default hyperparameters (lr=1e-3) to train each meta-classifier for 30 epochs.

Generative Model. The generative model consists of three blocks, each of which has a fully-connected layer,

TABLE III
ACCURACY OF THE SUBSTITUTE MODEL FOR MNIST AND FASHIONMNIST. THE HIGHER ACCURACY IS HIGHLIGHTED IN BOLD.

Dataset	Budget	Defense	PP (ϵ)			RS (β)				AM (τ)	EDM	Hybrid		Unknown	
			0.5	0.99	1.1	0.1	0.3	0.5	0.8	0.99		PP/RS	EDM+RD	Un-EDM	Un-RD
MNIST	0.5k	KnockoffNet	79.01%	61.04%	56.97%	**85.10%**	82.43%	78.73%	69.36%	74.39%	32.70%	77.83%	**64.10%**	32.70%	81.37%
		D-KnockoffNet	**80.36%**	**66.22%**	**58.00%**	81.18%	**82.83%**	**80.66%**	**76.54%**	**77.75%**	**33.10%**	**80.39%**	**64.10%**	**32.88%**	**84.35%**
		JBDA	26.24%	26.50%	18.62%	72.67%	67.72%	64.39%	60.41%	85.67%	12.91%	77.24%	55.11%	12.91%	55.70%
		D-JBDA	**73.14%**	**52.88%**	**45.51%**	**85.04%**	**81.14%**	**74.89%**	**61.33%**	**87.84%**	**15.72%**	**78.87%**	**59.33%**	**17.48%**	**64.77%**
		ActiveThief	37.44%	10.30%	10.09%	74.90%	72.79%	70.62%	68.36%	76.30%	14.59%	16.25%	24.51%	14.59%	43.95%
		D-ActiveThief	**43.00%**	**40.10%**	**40.20%**	**85.20%**	**79.40%**	**76.20%**	**77.40%**	**84.70%**	**28.05%**	**26.37%**	**43.70%**	**17.78%**	**56.27%**
	5k	KnockoffNet	90.66%	73.81%	67.52%	95.75%	93.57%	91.22%	86.43%	76.90%	45.90%	87.55%	62.60%	45.90%	93.28%
		D-KnockoffNet	**93.83%**	**80.29%**	**72.82%**	**96.66%**	**95.88%**	**94.88%**	**91.74%**	**81.35%**	**51.31%**	**91.39%**	**63.20%**	**49.83%**	**97.37%**
		JBDA	50.19%	26.56%	24.94%	75.82%	67.29%	64.75%	59.03%	96.04%	7.73%	78.12%	18.72%	7.73%	62.16%
		D-JBDA	**69.19%**	**45.58%**	**43.26%**	**93.40%**	**90.70%**	**85.37%**	**82.33%**	**97.74%**	**11.07%**	**80.88%**	**19.89%**	**8.78%**	**68.16%**
		ActiveThief	**90.73%**	72.93%	69.84%	82.04%	79.26%	75.20%	73.87%	96.40%	44.27%	92.49%	39.10%	44.27%	84.34%
		D-ActiveThief	86.30%	**83.00%**	**82.80%**	**88.60%**	**88.00%**	**87.80%**	**89.80%**	**98.10%**	**74.73%**	**94.95%**	**61.70%**	**56.33%**	**91.87%**
	20k	KnockoffNet	94.29%	73.97%	69.05%	97.79%	97.05%	96.57%	95.04%	80.58%	45.20%	92.79%	61.60%	45.20%	96.19%
		D-KnockoffNet	**97.33%**	**81.84%**	**75.45%**	**97.92%**	**97.69%**	**97.26%**	**96.62%**	**83.29%**	**49.69%**	**97.18%**	**61.90%**	**51.54%**	**98.11%**
		JBDA	41.42%	30.14%	24.32%	73.71%	61.36%	55.99%	56.37%	94.25%	12.18%	79.78%	12.10%	12.18%	57.67%
		D-JBDA	**65.85%**	**66.08%**	**55.98%**	**90.90%**	**87.10%**	**84.48%**	**78.59%**	**94.58%**	**12.65%**	**80.98%**	**12.56%**	**13.02%**	**60.73%**
		ActiveThief	87.82%	69.92%	67.02%	80.34%	77.30%	74.88%	71.25%	96.80%	47.64%	93.61%	44.32%	47.64%	93.14%
		D-ActiveThief	**93.60%**	**92.70%**	**88.30%**	**95.60%**	**94.40%**	**93.70%**	**89.20%**	**98.30%**	**77.19%**	**97.24%**	**62.31%**	**61.13%**	**97.36%**
	60k	KnockoffNet	96.01%	74.84%	68.61%	98.05%	97.98%	97.63%	96.88%	86.02%	46.20%	93.03%	60.31%	46.20%	97.35%
		D-KnockoffNet	**98.04%**	**85.10%**	**79.04%**	**98.34%**	**98.19%**	**98.07%**	**97.14%**	**89.94%**	**55.50%**	**97.84%**	**63.11%**	**52.72%**	**98.41%**
		JBDA	23.90%	16.11%	12.08%	75.11%	68.25%	69.51%	63.42%	92.38%	10.37%	67.53%	10.70%	10.37%	62.97%
		D-JBDA	**67.68%**	**61.75%**	**56.62%**	**88.60%**	**84.10%**	**82.28%**	**75.67%**	**92.55%**	**11.66%**	**85.37%**	**11.11%**	**10.55%**	**64.01%**

Dataset	Budget	Defense	PP (ϵ)			RS (β)				AM (τ)	EDM	Hybrid		Unknown	
			0.5	0.99	1.1	0.1	0.3	0.5	0.8	0.99		PP/RS	EDM+RD	Un-EDM	Un-RD
FashionMNIST	0.5k	KnockoffNet	35.57%	29.51%	23.41%	36.70%	35.41%	36.23%	31.63%	24.73%	12.50%	30.38%	36.32%	12.50%	34.35%
		D-KnockoffNet	**40.06%**	**31.87%**	**30.35%**	**40.00%**	**37.38%**	**37.10%**	**34.30%**	**32.56%**	**21.00%**	**40.81%**	**38.60%**	**16.62%**	**36.33%**
		JBDA	19.49%	13.66%	19.01%	19.26%	21.95%	18.28%	19.44%	32.31%	17.35%	23.96%	34.40%	17.35%	18.65%
		D-JBDA	**30.87%**	**30.82%**	**31.10%**	**34.27%**	**29.11%**	**33.57%**	**24.78%**	**33.42%**	**26.84%**	**27.81%**	**43.60%**	**27.55%**	**28.09%**
		ActiveThief	19.36%	14.65%	11.02%	10.00%	10.00%	10.00%	6.90%	42.10%	10.31%	10.45%	10.91%	10.31%	15.43%
		D-ActiveThief	**22.37%**	**18.70%**	**17.80%**	**16.70%**	**17.13%**	**16.30%**	**18.35%**	**48.70%**	**16.64%**	**16.64%**	**18.93%**	**11.97%**	**16.61%**
	5k	KnockoffNet	39.36%	34.78%	34.29%	38.72%	40.67%	36.66%	32.90%	26.96%	15.70%	36.68%	39.34%	15.70%	50.35%
		D-KnockoffNet	**49.52%**	**44.59%**	**42.15%**	**48.80%**	**46.14%**	**45.16%**	**41.64%**	**37.92%**	**18.77%**	**44.29%**	**41.22%**	**16.16%**	**54.54%**
		JBDA	13.54%	6.71%	10.50%	18.93%	24.88%	18.08%	14.83%	48.40%	19.51%	23.34%	39.14%	19.51%	28.47%
		D-JBDA	**41.30%**	**21.54%**	**18.23%**	**37.55%**	**35.93%**	**37.98%**	**32.37%**	**49.40%**	**23.99%**	**35.73%**	**40.70%**	**25.85%**	**31.97%**
		ActiveThief	40.62%	36.82%	32.17%	21.90%	16.60%	21.10%	12.90%	54.60%	11.23%	37.61%	9.70%	11.23%	37.04%
		D-ActiveThief	**45.38%**	**44.92%**	**44.06%**	**32.52%**	**36.12%**	**34.07%**	**32.95%**	**58.60%**	**13.74%**	**38.60%**	**21.70%**	**15.08%**	**39.89%**
	20k	KnockoffNet	48.89%	35.93%	36.97%	48.63%	50.39%	46.88%	36.34%	38.03%	12.90%	39.91%	40.71%	12.90%	50.86%
		D-KnockoffNet	**58.07%**	**53.01%**	**53.49%**	**61.71%**	**58.56%**	**57.41%**	**52.01%**	**47.35%**	**15.80%**	**54.78%**	**41.50%**	**14.15%**	**58.71%**
		JBDA	12.79%	9.41%	3.99%	16.75%	25.05%	19.21%	14.78%	42.90%	17.59%	29.78%	35.80%	17.59%	24.72%
		D-JBDA	**42.69%**	**22.15%**	**24.89%**	**30.45%**	**30.18%**	**32.42%**	**34.67%**	**44.08%**	**22.43%**	**39.86%**	**38.10%**	**26.92%**	**35.23%**
		ActiveThief	41.81%	34.54%	34.27%	26.10%	21.79%	13.35%	13.69%	55.90%	17.85%	39.86%	12.22%	17.85%	54.21%
		D-ActiveThief	**46.97%**	**45.02%**	**45.19%**	**34.26%**	**37.96%**	**35.34%**	**34.38%**	**57.02%**	**19.34%**	**46.73%**	**20.11%**	**19.15%**	**56.74%**
	60k	KnockoffNet	52.30%	39.01%	35.09%	57.38%	54.50%	56.40%	41.26%	40.64%	11.80%	43.58%	38.33%	11.80%	59.52%
		D-KnockoffNet	**62.90%**	**57.28%**	**57.27%**	**67.21%**	**62.44%**	**57.67%**	**56.24%**	**41.22%**	**17.31%**	**57.95%**	**39.20%**	**15.48%**	**65.13%**
		JBDA	14.02%	6.39%	5.82%	21.80%	23.54%	22.84%	14.59%	42.71%	15.33%	13.87%	35.70%	15.33%	32.14%
		D-JBDA	**30.16%**	**27.61%**	**31.15%**	**28.89%**	**29.18%**	**30.37%**	**32.88%**	**44.80%**	**20.14%**	**15.20%**	**40.21%**	**22.84%**	**34.41%**

BatchNorm1d, LeakyReLU, and a softmax layer to match the required output dimension. The final activation function is the hyperbolic tangent function. We use the SGD optimizer (lr=0.01) with momentum 0.9 to train each generative model for 20 epochs.

Substitute model. We initialize the substitute model with the same architecture as the victim model following existing works [32]. To train substitute models, we use the SGD optimizer (lr=0.1) with momentum 0.5 for 30 epochs (LeNet) and 100 epochs (VGG16-bn, ResNet-34) with a learning rate decayed by a factor of 0.1 every 60 epochs.

D. State-of-the-Art Defenses

We mainly consider four state-of-the-art disruption-based defense methods, i.e., PP [31], RS [28], AM [22], and EDM [20]. The four methods disrupt query results in different ways. In RS, the top-1 prediction of the victim model never changes, so it is an accuracy-preserving defense. In PP, the prediction accuracy of the victim model decreases as more noise is added to the query results, making it an accuracy-constraint defense. Both AM and EDM are combinations of detection- and disruption-based defenses. In AM, the defender adaptively sends misleading predictions to the identified OOD queries. In EDM, the defender produces different predictions for OOD queries with different members of the ensemble learning. We also evaluate prediction truncation methods, e.g., top-K, random noise (RN), and rounding [40]. Although prediction truncation methods are not designed for defending against model extraction attacks, it is shown that they can protect model privacy [31].

Prediction Poisoning (PP). PP [31] perturbs the query results based on a hyperparameter ϵ, which denotes the magnitude of perturbation. We evaluate the defense at $\epsilon = 0.5, 0.99, 1.1$.

Reverse Sigmoid (RS). RS [28] introduces ambiguity among non-max probabilities. We evaluate RS at various hy-

TABLE IV
ACCURACY OF THE SUBSTITUTE MODEL FOR CIFAR-10 AND GTSRB. THE HIGHER ACCURACY IS HIGHLIGHTED IN BOLD.

Dataset	Budget	Defense	PP (ϵ)			RS (β)				AM (τ)	EDM	Hybrid		Unknown	
			0.5	0.99	1.1	0.1	0.3	0.5	0.8	0.99		PP/RS	EDM+RD	Un-EDM	Un-RD
CIFAR-10	0.5k	KnockoffNet	22.47%	17.88%	16.68%	25.14%	24.24%	22.91%	21.39%	24.95%	35.50%	19.15%	31.53%	35.50%	24.31%
		D-KnockoffNet	**23.66%**	**20.19%**	**30.35%**	**25.28%**	**28.39%**	**23.61%**	**22.77%**	**26.57%**	**42.86%**	**21.33%**	**32.40%**	**41.48%**	**26.81%**
		JBDA	12.24%	10.00%	12.41%	15.06%	14.37%	12.63%	11.49%	13.58%	17.58%	20.55%	28.40%	17.58%	10.66%
		D-JBDA	**13.23%**	**14.04%**	**14.36%**	**16.24%**	**23.68%**	**14.24%**	**13.27%**	**16.49%**	**20.04%**	**23.57%**	**31.70%**	**19.47%**	**14.93%**
		ActiveThief	10.00%	10.14%	10.00%	10.00%	10.00%	10.20%	10.30%	**10.00%**	10.38%	12.15%	27.11%	10.38%	15.14%
		D-ActiveThief	**13.30%**	**12.88%**	**12.93%**	**12.64%**	**12.58%**	**11.97%**	**11.04%**	**10.00%**	**30.78%**	**15.13%**	**29.70%**	**21.47%**	**15.68%**
	5k	KnockoffNet	27.09%	23.26%	17.39%	36.53%	34.88%	32.49%	31.42%	37.45%	55.50%	32.25%	47.40%	55.50%	32.46%
		D-KnockoffNet	**34.38%**	**27.95%**	**42.15%**	**37.41%**	**35.74%**	**34.42%**	**32.58%**	**39.10%**	**61.98%**	**36.14%**	**48.20%**	**59.78%**	**40.30%**
		JBDA	12.54%	10.83%	14.69%	17.30%	11.87%	11.87%	16.53%	13.99%	12.19%	10.02%	11.50%	12.19%	10.90%
		D-JBDA	**13.10%**	**12.75%**	**15.87%**	**24.26%**	**14.18%**	**12.67%**	**31.77%**	**31.67%**	**14.97%**	**11.63%**	**13.70%**	**14.13%**	**13.56%**
		ActiveThief	18.00%	13.21%	13.00%	37.20%	33.10%	30.60%	28.00%	40.59%	46.14%	22.13%	44.51%	46.14%	19.97%
		D-ActiveThief	**35.61%**	**32.08%**	**31.90%**	**39.48%**	**35.96%**	**34.87%**	**32.05%**	**41.38%**	**46.85%**	**32.87%**	**45.74%**	**46.40%**	**21.29%**
	20k	KnockoffNet	32.94%	22.94%	20.50%	**48.58%**	43.58%	42.01%	39.12%	46.88%	67.50%	40.53%	59.20%	67.50%	51.46%
		D-KnockoffNet	**43.14%**	**34.43%**	**53.49%**	46.27%	**44.23%**	**49.96%**	**40.49%**	**50.65%**	**71.24%**	**41.46%**	**62.00%**	**67.75%**	**53.39%**
		JBDA	13.15%	12.42%	12.97%	12.49%	14.94%	15.98%	11.76%	17.22%	13.94%	11.41%	10.00%	13.94%	12.48%
		D-JBDA and	**14.00%**	**16.41%**	**13.27%**	**12.57%**	**30.83%**	**32.53%**	**12.42%**	**32.69%**	**17.46%**	**12.24%**	**10.50%**	**16.00%**	**15.73%**
		ActiveThief	33.00%	24.21%	20.00%	36.70%	34.10%	30.50%	29.80%	42.70%	55.84%	32.73%	54.00%	55.84%	30.54%
		D-ActiveThief	**38.72%**	**27.06%**	**29.54%**	**40.74%**	**37.63%**	**35.02%**	**33.69%**	**45.24%**	**56.48%**	**36.26%**	**57.34%**	**56.73%**	**31.36%**
	60k	KnockoffNet	45.26%	22.38%	15.11%	**57.98%**	**53.34%**	50.60%	47.04%	60.48%	71.50%	49.19%	65.20%	71.50%	61.77%
		D-KnockoffNet	**50.68%**	**40.34%**	**57.27%**	53.97%	51.65%	**57.22%**	**48.35%**	**68.97%**	**75.86%**	**51.30%**	**68.99%**	**73.05%**	**63.16%**
		JBDA	11.93%	11.30%	12.71%	12.77%	13.16%	12.01%	12.26%	15.88%	13.09%	10.09%	12.61%	13.09%	12.01%
		D-JBDA	**15.64%**	**12.98%**	**13.04%**	**13.26%**	**13.58%**	**15.16%**	**14.68%**	**32.59%**	**13.45%**	**12.43%**	**13.33%**	**15.40%**	**15.37%**

Dataset	Budget	Defense	PP (ϵ)			RS (β)				AM (τ)	EDM	Hybrid		Unknown	
			0.5	0.99	1.1	0.1	0.3	0.5	0.8	0.99		PP/RS	EDM+RD	Un-EDM	Un-RD
GTSRB	0.5k	KnockoffNet	21.52%	17.90%	16.73%	16.73%	16.84%	10.93%	5.57%	24.73%	13.25%	38.27%	6.50%	13.25%	48.80%
		D-KnockoffNet	**47.71%**	**42.69%**	**39.28%**	**31.18%**	**32.83%**	**30.66%**	**26.54%**	**32.87%**	**14.97%**	**46.48%**	**7.11%**	**14.26%**	**50.06%**
		JBDA	11.82%	3.30%	2.98%	32.46%	27.03%	24.51%	18.05%	5.94%	**5.70%**	5.22%	8.40%	**5.70%**	4.62%
		D-JBDA	**12.97%**	**5.70%**	**4.53%**	**37.84%**	**29.63%**	**27.58%**	**20.73%**	**9.46%**	**5.70%**	**11.41%**	**9.41%**	**5.70%**	**5.82%**
		ActiveThief	6.91%	0.71%	0.71%	27.81%	19.46%	13.78%	7.21%	30.44%	8.63%	9.32%	7.60%	8.63%	10.35%
		D-ActiveThief	**8.77%**	**8.54%**	**8.76%**	**28.47%**	**22.17%**	**23.54%**	**24.59%**	**35.57%**	**8.97%**	**12.25%**	**8.30%**	**8.97%**	**17.63%**
	5k	KnockoffNet	68.73%	63.46%	59.10%	61.47%	48.27%	38.72%	19.09%	65.29%	31.82%	89.59%	13.70%	31.82%	85.93%
		D-KnockoffNet	**80.41%**	**71.53%**	**66.82%**	**76.66%**	**65.88%**	**64.88%**	**31.74%**	**73.81%**	**40.40%**	**89.91%**	**14.55%**	**37.26%**	**92.88%**
		JBDA	1.38%	0.48%	0.36%	25.90%	6.17%	7.17%	7.63%	4.69%	6.52%	5.46%	7.50%	6.52%	4.29%
		D-JBDA	**5.96%**	**4.05%**	**5.02%**	**26.94%**	**9.35%**	**12.85%**	**10.98%**	**5.46%**	**6.68%**	**5.70%**	**9.10%**	**7.04%**	**5.47%**
		ActiveThief	83.20%	28.27%	29.56%	60.69%	49.58%	40.23%	29.56%	62.56%	10.48%	42.42%	18.11%	10.48%	88.37%
		D-ActiveThief	**84.46%**	**52.02%**	**31.88%**	**62.07%**	**60.32%**	**59.29%**	**59.30%**	**68.36%**	**10.59%**	**60.51%**	**19.10%**	**10.71%**	**92.58%**
	10k	KnockoffNet	77.57%	71.36%	68.20%	71.90%	59.64%	50.34%	26.54%	70.10%	34.27%	90.51%	25.92%	34.27%	94.58%
		D-KnockoffNet	**86.73%**	**82.64%**	**80.61%**	**87.92%**	**79.69%**	**67.26%**	**46.62%**	**88.65%**	**47.35%**	**91.23%**	**27.20%**	**39.46%**	**95.34%**
		JBDA	2.58%	1.19%	0.92%	20.28%	7.39%	5.80%	8.62%	5.23%	9.28%	5.48%	6.10%	9.28%	5.46%
		D-JBDA	**4.05%**	**6.57%**	**5.76%**	**25.89%**	**16.43%**	**15.44%**	**13.96%**	**5.46%**	**10.22%**	**7.50%**	**9.90%**	**9.94%**	**5.67%**
		ActiveThief	76.62%	64.66%	56.37%	76.62%	64.66%	56.37%	40.59%	73.02%	14.78%	65.19%	21.71%	14.78%	90.03%
		D-ActiveThief	**77.52%**	**74.30%**	**66.78%**	**85.97%**	**80.09%**	**74.42%**	**65.37%**	**78.29%**	**15.19%**	**73.14%**	**22.11%**	**15.27%**	**93.25%**
	30k	KnockoffNet	80.81%	74.41%	71.73%	82.71%	74.82%	67.81%	35.39%	82.97%	48.53%	95.34%	30.51%	48.53%	96.33%
		D-KnockoffNet	**92.16%**	**86.92%**	**78.40%**	**88.34%**	**82.19%**	**78.07%**	**57.14%**	**87.03%**	**53.31%**	**96.82%**	**34.00%**	**54.16%**	**96.80%**
		JBDA	23.90%	16.11%	12.08%	13.85%	14.14%	5.85%	10.01%	**5.46%**	5.46%	10.43%	19.01%	5.48%	5.23%
		D-JBDA	**27.65%**	**20.96%**	**18.80%**	**16.75%**	**14.73%**	**15.44%**	**14.17%**	**5.46%**	**9.61%**	**15.09%**	**20.10%**	**7.23%**	**5.46%**

perparameter settings with $\beta = 0.1, 0.3, 0.5, 0.8$ while keeping the dataset-specific hyperparameter γ fixed at 0.2 for MNIST, 0.4 for FashionMNIST, 0.1 for CIFAR-10, 0.2 for GTSRB, and 0.2 for ImageNette.

Adaptive Misinformation (AM). AM [22] utilizes an OOD detector to distinguish between benign and adversarial queries, and only disrupts identified OOD queries. Hyperparameter τ balances security and accuracy, and we set $\tau = 0.99$ in our experiments. AM needs an OOD dataset to train the outlier detector. We use KMNIST as the OOD dataset for MNIST and FashionMNIST, CIFAR-100 for CIFAR-10, LISA for GTSRB, and Mini-Imagenet for ImageNette.

Ensemble of Diverse Models (EDM). EDM [20] trains an ensemble of diverse models to achieve high accuracy on in-distribution data and diversity in out-of-distribution data. In the inference phase, a hash function is used to select a member of the ensemble to respond to the query. In our implementation, the parameter λ_D and the ensemble size are the same as in the original paper. The auxiliary OOD datasets are selected in the same way as for AM.

Top-K. Top-K leaves the top-K largest elements in the prediction vector unchanged and replaces the others by zero.

Rounding (RD). Rounding retains several digits (default=3) after the decimal point for each element in the prediction vector.

Random Noise (RN). RN adds Gaussian noise $z \in \mathcal{N}(0, \sigma^2)$ to the prediction vector with $\sigma^2 = 0.2$ as the default setting.

Combination of multiple defenses (hybrid defense). In addition to single defenses, we also consider two combinations of multiple defenses, referred to as hybrid defenses. More specifically, each hybrid defense is a combination of two defenses. In D-DAE, we perform defense detection for each query and then leverage the corresponding generative model to recover the query.

PP/RS. The defender alternates randomly between PP and RS for each query.

EDM+RD. The defender performs Rounding after EDM.

TABLE V
ACCURACY OF THE SUBSTITUTE MODEL FOR IMAGENETTE. THE HIGHER ACCURACY IS HIGHLIGHTED IN BOLD.

Dataset	Budget	Defense	PP (ϵ)			RS (β)				EDM	Hybrid		Unknown	
			0.5	0.99	1.1	0.1	0.3	0.5	0.8		PP/RS	EDM+RD	Un-EDM	Un-RD
ImageNette	0.1k	KnockoffNet	71.76%	52.30%	54.73%	**90.77%**	82.97%	72.30%	44.07%	45.40%	84.09%	50.71%	45.40%	13.65%
		D-KnockoffNet	**79.56%**	**70.13%**	**69.00%**	87.81%	**84.29%**	**79.17%**	**68.20%**	**88.24%**	**88.24%**	**51.20%**	**56.66%**	**18.11%**
		JBDA	34.83%	17.30%	15.94%	12.10%	12.45%	11.97%	11.24%	10.32%	28.21%	66.00%	10.32%	13.80%
		D-JBDA	**87.94%**	**33.44%**	**30.45%**	**94.34%**	**88.60%**	**70.67%**	**31.54%**	**14.50%**	**30.45%**	**66.77%**	**12.74%**	**15.70%**
		ActiveThief	52.01%	33.50%	20.10%	86.90%	74.00%	57.33%	31.10%	22.73%	82.38%	51.00%	22.73%	70.52%
		D-ActiveThief	**55.94%**	**51.93%**	**40.99%**	**87.11%**	**78.87%**	**59.90%**	**53.83%**	**36.42%**	**87.11%**	**54.73%**	**28.57%**	**71.11%**
	1k	KnockoffNet	91.58%	71.80%	70.63%	96.39%	96.12%	95.58%	90.11%	58.30%	94.37%	64.94%	58.30%	17.26%
		D-KnockoffNet	**95.22%**	**87.04%**	**82.93%**	**97.74%**	**97.60%**	**97.02%**	**95.29%**	**60.08%**	**96.71%**	**67.70%**	**62.37%**	**22.26%**
		JBDA	13.11%	10.94%	11.32%	15.20%	14.89%	14.79%	11.34%	28.36%	18.92%	15.70%	28.36%	15.08%
		D-JBDA	**19.94%**	**13.54%**	**12.88%**	**21.18%**	**17.38%**	**17.84%**	**15.21%**	**28.50%**	**23.35%**	**16.72%**	**28.62%**	**16.46%**
		ActiveThief	89.62%	53.55%	57.40%	96.40%	95.00%	91.91%	76.51%	33.40%	85.21%	84.94%	33.40%	84.60%
		D-ActiveThief	**95.35%**	**85.21%**	**84.94%**	**97.20%**	**95.69%**	**95.19%**	**93.13%**	**49.07%**	**89.55%**	**90.67%**	**42.08%**	**85.22%**
	5k	KnockoffNet	93.41%	76.60%	57.56%	97.36%	97.56%	**97.52%**	96.24%	51.75%	96.12%	66.73%	51.75%	95.50%
		D-KnockoffNet	**97.09%**	**85.88%**	**83.12%**	**97.97%**	**97.67%**	96.98%	**96.62%**	**52.75%**	**97.02%**	**69.90%**	**52.25%**	**96.70%**
		JBDA	13.42%	11.21%	11.68%	16.73%	14.02%	13.20%	11.48%	14.78%	16.40%	10.22%	14.78%	15.34%
		D-JBDA	**14.97%**	**12.22%**	**12.22%**	**22.89%**	**15.42%**	**13.73%**	**13.30%**	**24.13%**	**27.69%**	**10.40%**	**25.02%**	**27.58%**
		ActiveThief	92.73%	54.34%	58.91%	88.06%	86.99%	79.46%	77.29%	33.94%	93.28%	69.70%	33.94%	95.62%
		D-ActiveThief	**97.46%**	**86.63%**	**85.79%**	**98.05%**	**97.62%**	**97.52%**	**97.31%**	**34.63%**	**96.15%**	**71.80%**	**34.21%**	**96.90%**
	10k	KnockoffNet	94.80%	84.94%	58.45%	97.21%	97.21%	97.25%	97.02%	51.05%	96.43%	72.66%	51.05%	96.28%
		D-KnockoffNet	**96.89%**	**87.27%**	**81.96%**	**97.74%**	**97.68%**	**97.43%**	**97.07%**	**56.09%**	**97.33%**	**74.50%**	**54.96%**	**96.90%**
		JBDA	8.99%	10.28%	10.28%	19.38%	17.20%	17.01%	15.96%	15.52%	15.24%	9.81%	15.52%	19.13%
		D-JBDA	**13.19%**	**11.87%**	**15.52%**	**21.10%**	**18.00%**	**17.73%**	**16.87%**	**33.01%**	**17.38%**	**11.80%**	**27.35%**	**27.44%**

Unknown defenses. To evaluate whether D-DAE is also effective for unknown defenses, we design the following attack scenarios.

Un-EDM. The attacker trains D-DAE using defenses PP, RS, AM, RD, and Top-K, and tests D-DAE against a model protected by EDM.

Un-RD. The attacker trains D-DAE using defenses PP, RS, AM, EDM, and Top-K, and tests D-DAE against a model protected by RD.

E. Real-world APIs

We consider two real-world APIs hosted by Microsoft Azure [2] and Face++ [1] with unknown defenses.

Microsoft Traffic Recognition API. We train a model on the GTSRB dataset through Microsoft Custom Vision and deploy it on Microsoft Azure as the black-box victim model.

Face++ Emotion Recognition API. The Face++ Emotion Recognition API does not allow users to train models on their own datasets, but only provides an interface for users to query, making it an entirely black-box victim model. Given a query image, the API returns the probability of the image reflecting emotions of happiness, fear, surprise, anger, disgust, neutrality, and sadness. As the original training and testing datasets of the API are unknown, we use RAF [34] to query the API. More specifically, we use 10,064 samples for query and 2,674 samples to test the performance of the substitute model.

VI. EVALUATION RESULTS

A. Enhancement of Existing Attacks

Table III, Table IV, and Table V show the experiment results of applying D-DAE to existing model extraction attacks under the four disruption-based defenses, i.e., PP [31], RS [28], AM [22], and EDM [20] with various hyperparameters. Note that for the query budget higher than 20k, we did not have the results of ActiveThief since it takes extremely long for the algorithm to converge. The detailed analysis of experimental results is as follows. Due to page limitation, the results of agreement are show in Table XI, Table XII, and Table XIII in the Appendix. Note that the agreement between the surrogate model and the protected victim model is not necessarily high since some accuracy-constraint defense strategies (e.g., PP) reduce the accuracy of the original victim model by altering the top-1 prediction results.

1) Single defense: For all but 7 cases of single defense listed, our proposed attack framework enhances existing generic model extraction attacks. It is shown that the defenses indeed lower the accuracy of the substitute model compared with the accuracy of the victim model in Table II, especially for earlier attack JBDA. Under the disruption defense methods, the enhancement of D-DAE over generic model extraction attacks can be as large as 82.24%. We can observe that when the query budget increases, the performance of existing methods may not improve as expected, possibly because the query results are disrupted and mislead the training of surrogate models. In contrast, in most cases, the corresponding D-DAE models have better performance as the query budget increases, since disrupted query results are recovered and contribute to the training of the surrogate model. In only a small number of cases when the disruption is mild (e.g., RS with $\beta = 0.1$ and PP with $\epsilon = 0.5$), D-DAE has a lower accuracy than the corresponding generic model extraction methods. This is probably because the defensive intensity is relatively small, thus the differences in the distributions of disrupted and undisrupted query results are similar, which affects the effectiveness of the meta-classifiers and the generative models of D-DAE. Note that the AM defense has not been applied to ImageNette in the original paper, and we have found that we cannot obtain a victim model with acceptable prediction accuracy for ImageNette. We speculate that ImageNette is a high-resolution dataset, making the victim model *overfit* on *OOD dataset* in AM. Therefore, we did not implement AM on ImageNette.

TABLE VI
PERFORMANCE OF D-DAE AGAINST REAL-WORLD APIS ON MICROSOFT AZURE AND FACE++.

Microsoft Azure				Face++			
Budget	Attack	Accuracy	Agreement	Budget	Attack	Accuracy	Agreement
0.5k	KnockoffNet	19.56%	28.70×	0.5k	KnockoffNet	37.42%	43.51×
	D-KnockoffNet	**20.70%**	**45.72**×		D-KnockoffNet	**37.59%**	**43.83**×
	JBDA	7.31%	20.43×		JBDA	38.20%	33.44×
	D-JBDA	**26.24%**	**36.49**×		D-JBDA	**39.05%**	**38.94**×
	ActiveThief	12.90%	27.45×		ActiveThief	39.41%	33.90×
	D-ActiveThief	**26.04%**	**40.51**×		D-ActiveThief	**41.66%**	**41.70**×
1k	KnockoffNet	28.31%	38.90×	1k	KnockoffNet	39.57%	47.59×
	D-KnockoffNet	**31.93%**	**47.92**×		D-KnockoffNet	**40.36%**	**48.06**×
	JBDA	6.91%	6.40×		JBDA	**38.89%**	33.74×
	D-JBDA	**8.06%**	**9.19**×		D-JBDA	38.59%	**38.49**×
	ActiveThief	24.09%	37.81×		ActiveThief	38.62%	34.29×
	D-ActiveThief	**31.95%**	**46.83**×		D-ActiveThief	**48.92%**	**49.40**×
5k	KnockoffNet	49.80%	49.98×	3k	KnockoffNet	45.57%	55.35×
	D-KnockoffNet	**53.20%**	**50.06**×		D-KnockoffNet	**45.67%**	**56.31**×
	JBDA	8.00%	5.89×		JBDA	39.02%	33.77×
	D-JBDA	**10.58%**	**11.36**×		D-JBDA	**42.34%**	**41.52**×
	ActiveThief	34.11%	50.34×		ActiveThief	39.63%	35.46×
	D-ActiveThief	**37.92%**	**52.64**×		D-ActiveThief	**56.32%**	**55.68**×
10k	KnockoffNet	50.09%	53.22×	5k	KnockoffNet	47.85%	56.24×
	D-KnockoffNet	**58.98%**	**54.37**×		D-KnockoffNet	**48.35%**	**56.43**×
	JBDA	6.90%	5.35×		JBDA	38.62%	33.57×
	D-JBDA	**10.23%**	**8.89**×		D-JBDA	**43.29%**	**42.53**×
	ActiveThief	37.08%	52.95×		ActiveThief	41.46%	37.71×
	D-ActiveThief	**41.72%**	**57.23**×		D-ActiveThief	**58.12%**	**56.73**×
30k	KnockoffNet	54.27%	56.21×	10k	KnockoffNet	49.67%	58.21×
	D-KnockoffNet	**64.30%**	**58.00**×		D-KnockoffNet	**50.04%**	**58.59**×
	JBDA	8.57%	5.70×		JBDA	31.23%	28.91×
	D-JBDA	**9.49%**	**9.98**×		D-JBDA	**41.98%**	**41.55**×
	ActiveThief	40.29%	56.68×		ActiveThief	46.81%	41.30×
	D-ActiveThief	**42.67%**	**57.55**×		D-ActiveThief	**60.01%**	**59.08**×

TABLE VII
THE IMPACT OF THE NUMBER OF OF SHADOW MODEL ON THE ACCURACY OF META-CLASSIFIERS.

	% of training samples	256	512	1,024	2,048
MNIST	0.5%	77.22%	84.39%	85.02%	89.78%
	1.0%	74.86%	78.18%	83.99%	90.54%
	2.0%	90.00%	89.90%	88.28%	89.00%
	3.0%	89.00%	89.62%	90.00%	91.00%
FashionMNIST	0.5%	73.94%	80.57%	80.86%	86.42%
	1.0%	73.69%	82.70%	86.13%	87.81%
	2.0%	76.55%	83.65%	85.26%	89.61%
	3.0%	75.68%	76.88%	87.41%	88.85%
CIFAR-10	0.5%	64.87%	65.49%	65.59%	66.02%
	1.0%	62.29%	63.96%	64.45%	64.97%
	2.0%	58.94%	60.99%	63.24%	64.84%
	3.0%	54.36%	57.11%	60.04%	60.71%
GTSRB	0.5%	65.92%	66.96%	68.78%	71.97%
	1.0%	61.81%	63.53%	64.27%	66.09%
	2.0%	78.39%	90.60%	92.74%	93.98%
	3.0%	84.14%	92.12%	93.04%	93.81%
ImageNette	0.5%	65.31%	64.65%	67.34%	62.76%
	1.0%	61.81%	63.53%	64.27%	66.09%
	2.0%	65.82%	65.49%	69.31%	71.55%
	3.0%	66.38%	64.25%	68.57%	73.69%

2) Hybrid defense: When the defender uses hybrid defenses, e.g., PP/RS or EDM+RD, the defense power is generally higher than that of single defenses. In this case, we perform per-query disruption detection as the defender may randomly alternate between different defenses. Hybrid defenses pose a great challenge to model extraction attacks, yet D-DAE still enhances generic methods. It is shown that the disruption detection module in D-DAE can differentiate different defenses adopted by the defender with relatively high accuracy.

3) Unknown defense: As shown in Table III, IV, and V, we can see that the substitute models stolen by D-DAE have much higher accuracy than those stolen by KnockoffNet, JBDA, and ActiveThief for all datasets and all levels of budgets. Take MNIST dataset as an example, the results in Un-RD show that the substitute model stolen by D-KnockoffNet has an accuracy of 84.35% (0.5k budget) and 97.37% (5k budget), while the substitute model stolen by KnockoffNet has an accuracy of 81.37% (0.5k budget) and 93.28% (5k budget). These results verify the effectiveness of D-DAE against unknown defenses.

4) Impact of defense hyperparameters: In PP, ϵ denotes the magnitude of added noise, so does β in RS. The larger these hyperparameters, the stronger the defense. We find that D-DAE achieves better performance than baselines even under the strongest defense. PP can effectively degrade the performance of baselines with a small ϵ. As ϵ increases from 0.5 to 1.1, the accuracy of almost all substitute models decreases,

TABLE VIII
AUC OF META-CLASSIFIERS IN D-DAE. $\epsilon = 0.5$ FOR PREDICTION POISONING (PP). $\beta = 0.1, \gamma = 0.2$ FOR REVERSE SIGMOID(RS). $\sigma^2 = 0.2$ FOR RANDOM NOISE (RN). 3 DIGITS FOR TOP-K. 3 DIGITS FOR ROUNDING. $\tau = 0.99$ FOR AM.

	PP	RS	RN	Top-K	RD	AM	EDM
MNIST	91.0%	98.41%	100%	100%	100%	87.59%	100%
FashionMNIST	89.61%	93.66%	100%	100%	100%	84.38%	100%
GTSRB	93.98%	95.05%	99.60%	100%	100%	74.95%	89.54%
CIFAR-10	64.84%	74.48%	100%	100%	100%	68.27%	100%
ImageNette	71.55%	74.89%	98.86%	100%	100%	91.42%	98.57%

since larger noises are added to the prediction results. At low query budgets, the baselines are more susceptible to the defense. In RS, the accuracy of the substitute model drops less when the defense magnitude β increases than when ϵ increases in PP. This is mainly because RS is an accuracy-preserving defense such that the noise-adding operation is more conservative than that in PP. For D-DAE, the impact of the RS defense is much smaller.

5) Real-world APIs: Table VI demonstrates the performance of D-DAE against real-world APIs of Azure and Face++ with absolutely unknown defense strategies. It is shown that even if the defense method may not be in the set of known defense strategies, D-DAE can still enhance model extraction attacks by as high as 18.93%.

B. The Effectiveness of Disruption Detection

In this part, we examine the accuracy of the meta-classifier in detecting the defenses adopted by the defender. After training the meta-classifiers, we test their prediction accuracy with another 256 shadow models protected by different defenses. We set the hyperparameters in test shadow models as $\epsilon = 0.5$ for PP, $\beta = 0.1, \gamma = 0.2$ for RS, $\sigma^2 = 0.2$ for RN, keeping top-3 digits for Top-K, and retaining 3 digits for Rounding. We set the threshold T as 0.5.

From Table VIII we can see that, for simple datasets (e.g., MNIST, GTSRB, and FashionMNIST), the corresponding meta-classifiers can achieve AUC of more than 90%. For complex datasets (e.g., ImageNette and CIFAR-10), the performance of the meta-classifiers decreases slightly. Top-K and Rounding may be distinguished by looking at the number of digits in confidence scores. Surprisingly, RN can be detected with an accuracy of nearly 100% possibly because the added noise follow certain distributions that can be detected. However, when it comes to more advanced defenses, such as PP, RS, AM and EDM, the confidence score of the meta-classifiers decreases, as shown in Table X. We also explore the impact of the shadow dataset size on the performance of meta-classifiers. As shown in Table VII, in general, as the size of shadow datasets increases, the accuracy will increase.

C. Effectiveness of Disruption Recovery

1) Impact of Loss Function: We evaluate the impact of loss functions on the performance of the generative model. The shadow models are protected by PP ($\epsilon = 0.5$). We evaluate the effectiveness of disruption recovery using three metrics, (1) the Manhattan distance D_{l_1} between the generated recovered output $\mathcal{G}_k(f_s^k(x_s))$ and the original undisrupted output $f_s(x_s)$, (2) the Euclidean distance D_{l_2} between $\mathcal{G}_k(f_s^k(x_s))$ and $f_s(x_s)$, and (3) the value of loss function $\mathcal{L}$. As shown in Fig. 5 (Appendix), MAE and MSE achieve a significantly closer distance between the generated output and the original output compared to KLD on all four datasets. This may be because the KLD loss suffers from vanishing gradients, as explained in Section IV-B. Note that we do not display the results of GTSRB here, because the output of the GTSRB classification model has 43 dimensions, making the Manhantan distance and the Euclidean distance much higher compared with other 10-class datasets.

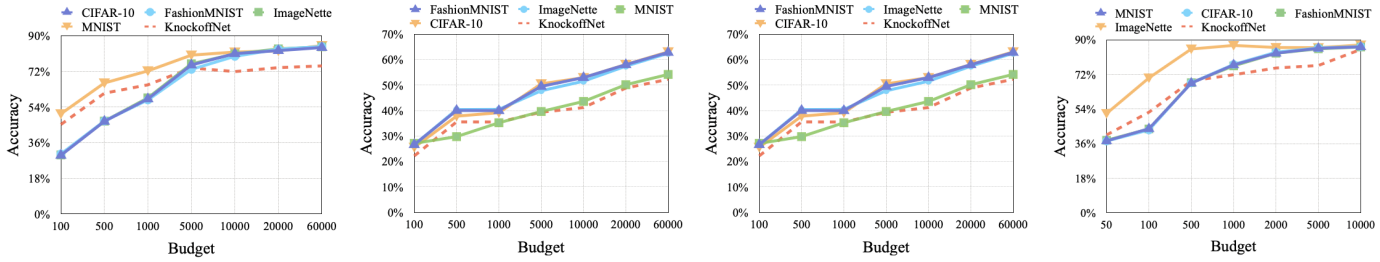
2) Generative Model Transferability: We train the generative model in D-DAE with datasets that are quite different from the dataset of the victim model. When the victim model is protected by PP ($\epsilon = 0.99$), the accuracy of the substitute model under D-DAE with a transferred generative model is shown in Fig. 3. Note that the output of the GTSRB generative model is 43-dimensional, which does not match the 10-dimensional generative models of MNIST, FashionMNIST, CIFAR-10, and ImageNette. We can observe that the generative models show a certain degree of transferability to other datasets. With a maximum budget of 60k, there is no significant difference in the performance of different generative models.

D. Different Victim-Surrogate Model Architectures

In the previous experiments, we train the surrogate model using the same architecture as the victim model. Now we study the influence of the architectural choice of surrogate models on the attack performance. When the victim model architectures are VGG16-bn for CIFAR-10 and Resnet-34 for ImageNette, we choose different architectures for the surrogate model. For CIFAR-10, we choose Alexnet, Resnet-18, Resnet-34, Resnet-50, and Densenet-161 for the surrogate model, and for ImageNette, we choose Alexnet, Resnet-18, Resnet-34, Resnet-50, VGG16-bn, and VGG19-bn for the surrogate model. As shown in Fig. 4 in the Appendix, even if there is a mismatch between the victim model architecture and the surrogate model architecture, as long as the surrogate model architecture is sufficiently complex, the attacker can successfully steal the victim model.

E. Cost Analysis

The overhead of D-DAE can be divided into two parts: 1) offline training of shadow models, meta-classifiers, and generators, and 2) online query detection and recovery based on the trained meta-classifiers and generators. The offline training process induces a one-shot cost. We present the training time regarding the five datasets empirically in Table IX. If the defender adopts a new defense method that comes to the knowledge of the attacker. The attacker only needs to train the corresponding shadow models, meta-classifiers, and generators offline for the defense. Note that the trained meta-classifiers and generators can be reused to extract different



(a) $\mathcal{D}_V$: MNIST (b) $\mathcal{D}_V$: FashionMNIST (d) $\mathcal{D}_V$: CIFAR-10 (e) $\mathcal{D}_V$: ImageNette

Fig. 3. The test accuracy of substitute models when using transferred generative model v.s. task-specific generative model to recover query results disrupted by PP ($\epsilon = 0.99$). KnockoffNets (dotted line) is the baseline.

TABLE IX
OFFLINE TRAINING COST OF D-DAE (IN SECONDS).

Dataset	Shadow Model	Meta-Classifier	Generator
MNIST	10	205	14,952
FashionMNIST	10	284	16,859
CIFAR-10	28	238	20,365
GTSRB	52	250	29,482
ImageNette	46	1,582	27,024

models for multiple times. If the defender adopts an unknown defense method, it is shown that D-DAE has a certain degree of transferability, possibly because some new defense methods are the enhancement of conventional defense methods. The online query detection and recovery process only load the meta-classifiers and generators for inference, which adds slight overhead to the generic model extraction framework that D-DAE is embedded in.

VII. DISCUSSIONS

In this section, we discuss the limitations of D-DAE.

A. Detection- & DP-based Defenses

We only focus on penetrating disruption-based defenses against model extraction attacks by recovering disrupted prediction results. As detection-based defenses check the query samples for abnormality and D-DAE only modifies the query results but not query samples, whether the attack can be detected by detection-based defenses relies mainly on the generic model extraction framework but little on D-DAE. For example, it is shown that JBDA is vulnerable to the detection-based defense PRADA [19] so that it is likely that D-JBDA is also detectable by PRADA. For certain generic model extraction frameworks, D-DAE indirectly affects the effectiveness of detection-based defenses as some generic model extraction frameworks use the query results as feedback for query sample generation. If the detection-based defense completely rejects to return the results of suspicious queries, we cannot acquire any information to retrain the substitute model, but this may greatly affect the experience of normal users who are wrongly detected as attackers. If the detection-based defense instead returns disrupted query results, our attack is unaffected since we can recover clean query results from disrupted query results. To bypass detection-based defenses, we need to engineer the *victim model query* module in generic model extraction frameworks such that the generated query sample distribution is similar to the normal query distribution, e.g., by injecting dummy query samples. This is our future work.

We have introduced two works [44], [46], which, as far as we know, are the only studies on DP-based model extraction defenses. However, the codes of these two works have not been open-sourced yet. As the code of D-DAE is open-sourced, interested readers may evaluate the effectiveness of D-DAE against these two works once their codes are available and against any other future defenses for model extraction attacks.

B. Generalizability

The meta-classifier learns from the outputs of defended and undefended shadow models to differentiate disrupted and undisrupted query results. Therefore, the diversity of shadow models is influential on the generalizability of the meta-classifier. A shadow model with similar architecture to the victim model is more likely to produce prediction results whose distribution is closer to that of the victim model. However, we assume that the victim model is unknown to the attacker. Thus, the attacker needs to build various shadow models with different possible architectures to enable the meta-classifier to identify all kinds of disrupted/undisrupted query results. Increasing the diversity of shadow models will improve the generalizability of the meta-classifier, but will lead to higher overheads. One of the possible solutions is to leverage meta-learning to enhance the generalizability of the meta-classifier with lower costs, which is our future direction.

VIII. CONCLUSION

We propose a defense-penetrating model extraction framework, named D-DAE, to construct a well-performed substitute model even if disruption-based defenses protect the victim model. We detect the specific defense method adopted by the defender with meta-classifiers and recover disrupted query results with a generative model. Extensive experiment results have confirmed that D-DAE can greatly improve the accuracy of the substitute model under defenses.

IX. ACKNOWLEDGEMENT

We thank the anonymous shepherd and reviewers for their valuable comments. Yanjiao's research is partially supported by the National Natural Science Foundation of China No. 61972296.

REFERENCES

[1] Face++. https://www.faceplusplus.com/, 2022.

[2] Microsoft Azure. https://azure.microsoft.com/en-us/, 2022.

[3] Emre Kiciman Andrew Marshall, Jugal Parikh and Ram Shankar Siva Kumar. Threat Modeling AI/ML Systems and Dependencies. https://docs.microsoft.com/en-us/security/engineering/threat-modeling-aiml5-model-stealing, 2022.

[4] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy*, pages 39–57, 2017.

[5] Varun Chandrasekaran, Kamalika Chaudhuri, Irene Giacomelli, Somesh Jha, and Songbai Yan. Exploring connections between active learning and model extraction. In *USENIX Security Symposium*, pages 1309–1326, 2020.

[6] Junjie Chen, Wendy Hui Wang, and Xinghua Shi. Differential privacy protection against membership inference attack on machine learning for genomic data. In *BIOCOMPUTING 2021: Proceedings of the Pacific Symposium*, pages 26–37. World Scientific, 2020.

[7] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *International Joint Conference on Neural Networks*, pages 2921–2926. IEEE, 2017.

[8] Jacson Rodrigues Correia-Silva, Rodrigo F Berriel, Claudine Badue, Alberto F de Souza, and Thiago Oliveira-Santos. Copycat cnn: Stealing knowledge by persuading confession with random non-labeled data. In *International Joint Conference on Neural Networks*, pages 1–8. IEEE, 2018.

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

[10] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference*, pages 265–284. Springer, 2006.

[11] Gongfan Fang, Jie Song, Chengchao Shen, Xinchao Wang, Da Chen, and Mingli Song. Data-free adversarial distillation. *arXiv preprint arXiv:1912.11006*, 2019.

[12] Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. In *USENIX Security Symposium*, pages 17–32. USENIX Association, 2014.

[13] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

[14] Xueluan Gong, Qian Wang, Yanjiao Chen, Wang Yang, and Xinchang Jiang. Model extraction attacks and defenses on cloud-based machine learning models. *IEEE Communications Magazine*, 58(12):83–89, 2020.

[15] 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.

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

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

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

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

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

[21] Sanjay Kariyappa, Atul Prakash, and Moinuddin K Qureshi. Maze: Data-free model stealing attack using zeroth-order gradient estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13814–13823, 2021.

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

[23] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2014.

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

[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 25:1097–1105, 2012.

[26] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.

[27] Yann LeCun, Corinna Cortes, and Chris Burges. Mnist handwritten digit database, 2010.

[28] Taesung Lee, Benjamin Edwards, Ian Molloy, and Dong Su. Defending against machine learning model stealing attacks using deceptive perturbations. In *IEEE Security and Privacy Workshops*, 2019.

[29] Andrew L. Maas. Rectifier nonlinearities improve neural network acoustic models. 2013.

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

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

[32] Soham Pal, Yash Gupta, Aditya Shukla, Aditya Kanade, Shirish Shevade, and Vinod Ganapathy. Activethief: Model extraction using active learning and unannotated public data. In *AAAI Conference on Artificial Intelligence*, volume 34, pages 865–872, 2020.

[33] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *ACM on Asia Conference on Computer and Communications Security*, pages 506–519, 2017.

[34] RAF-DB. Real-world Affective Faces Database. http://www.whdeng.cn/raf/model1.html, 2022.

[35] Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J Fleet. Adversarial manipulation of deep representations. In *International Conference on Learning Representations*, 2016.

[36] Samuel Sanford Shapiro and Martin B Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4):591–611, 1965.

[37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[38] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: A multi-class classification competition. In *International Joint Conference on Neural Networks*, pages 1453–1460. IEEE, 2011.

[39] Richard S Sutton, Andrew G Barto, et al. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

[40] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction apis. In *USENIX Security Symposium*, pages 601–618, 2016.

[41] Jean-Baptiste Truong, Pratyush Maini, Robert J Walls, and Nicolas Papernot. Data-free model extraction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4771–4780, 2021.

[42] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[43] Xiaojun Xu, Qi Wang, Huichen Li, Nikita Borisov, Carl A Gunter, and Bo Li. Detecting ai trojans using meta neural analysis. In *IEEE Symposium on Security and Privacy*, pages 103–120, 2021.

[44] Haonan Yan, Xiaoguang Li, Hui Li, Jiamin Li, Wenhai Sun, and Fenghua Li. Monitoring-based differential privacy mechanism against query flooding-based model extraction attack. *IEEE Transactions on Dependable and Secure Computing*, 2021.

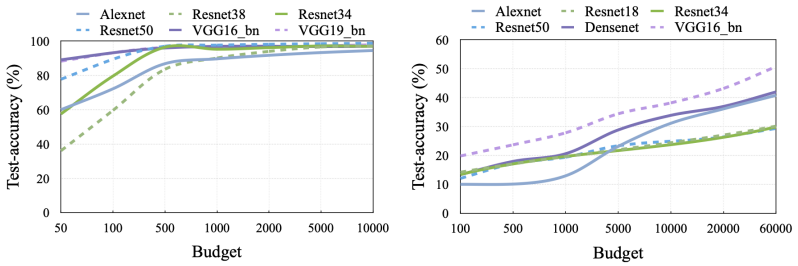
[45] Honggang Yu, Kaichen Yang, Teng Zhang, Yun-Yun Tsai, Tsung-Yi Ho, and Yier Jin. Cloudleak: Large-scale deep learning models stealing through adversarial examples. In *Annual Network and Distributed System Security Symposium*. The Internet Society, 2020.

[46] Huadi Zheng, Qingqing Ye, Haibo Hu, Chengfang Fang, and Jie Shi. Bdpl: A boundary differentially private layer against machine learning model extraction attacks. In *European Symposium on Research in Computer Security*, pages 66–83. Springer, 2019.

X. Appendix

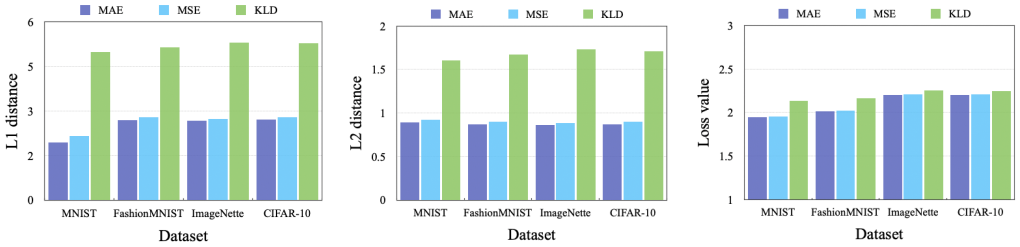
TABLE X
CONFIDENCE SCORE OF META-CLASSIFIERS (IN ROW) FOR DIFFERENT DEFENSES (IN LINE).

Dataset	Defense	PP	RS	RN	Top-K	RD	AM	EDM
MNIST	PP	**92.58%**	62.38%	59.92%	50.59%	50.00%	49.26%	57.38%
	RS	72.50%	**99.49%**	87.85%	50.00%	50.00%	52.30%	79.60%
	RN	81.64%	96.29%	**99.61%**	50.45%	50.00%	52.03%	79.65%
	Top-K	53.71%	66.60%	58.70%	**90.86%**	50.00%	50.00%	54.26%
	RD	29.57%	44.53%	49.41%	0.16%	**100.00%**	20.04%	49.73%
	AM	50.00%	50.00%	50.00%	50.12%	50.00%	**90.00%**	50.04%
	EDM	73.63%	84.14%	79.30%	50.04%	50.00%	50.08%	**99.77%**
Dataset	Defense	PP	RS	RN	Top-K	RD	AM	EDM
FashionMNIST	PP	**90.98%**	52.77%	65.51%	50.00%	50.00%	50.00%	69.34%
	RS	50.00%	**76.17%**	62.34%	50.08%	50.00%	50.00%	48.36%
	RN	50.00%	54.18%	**68.36%**	50.00%	50.00%	50.00%	62.62%
	Top-K	50.00%	52.11%	59.22%	**90.08%**	50.00%	50.00%	59.69%
	RD	0.00%	1.45%	10.55%	0.00%	**99.80%**	0.00%	11.56%
	AM	50.00%	50.00%	50.00%	50.20%	50.00%	**87.95%**	50.43%
	EDM	50.00%	51.41%	65.82%	50.04%	50.00%	50.00%	**73.44%**
Dataset	Defense	PP	RS	RN	Top-K	RD	AM	EDM
CIFAR-10	PP	**90.00%**	50.00%	50.08%	50.00%	50.00%	50.00%	50.12%
	RS	50.00%	**60.39%**	50.04%	50.00%	50.00%	50.00%	50.08%
	RN	50.00%	50.00%	**87.69%**	50.00%	50.00%	50.04%	69.82%
	Top-K	50.00%	50.00%	50.12%	**70.96%**	50.00%	50.00%	50.27%
	RD	0.00%	0.00%	0.00%	0.00%	**100.00%**	15.47%	0.27%
	AM	50.00%	50.00%	50.04%	50.00%	50.00%	**79.04%**	50.00%
	EDM	50.00%	50.00%	50.04%	50.00%	50.00%	50.00%	**61.17%**
Dataset	Defense	PP	RS	RN	Top-K	RD	AM	EDM
GTSRB	PP	**76.76%**	58.59%	56.21%	61.64%	50.00%	50.23%	57.03%
	RS	59.88%	**96.48%**	86.56%	54.73%	50.00%	50.04%	66.99%
	RN	71.60%	85.12%	**96.72%**	54.41%	50.00%	50.16%	74.92%
	Top-K	50.00%	50.00%	50.12%	**78.27%**	50.00%	50.00%	50.27%
	RD	18.01%	44.80%	46.72%	4.77%	**100.00%**	10.12%	44.45%
	AM	54.26%	57.81%	52.30%	63.79%	50.00%	**72.96%**	56.05%
	EDM	69.84%	85.43%	85.2%	54.10%	50.00%	50.24%	**97.85%**
Dataset	Defense	PP	RS	RN	Top-K	RD	AM	EDM
ImageNette	PP	**70.00%**	51.21%	52.15%	50.55%	50.00%	50.00%	55.27%
	RS	50.00%	**55.34%**	53.79%	50.82%	50.00%	50.00%	47.54%
	RN	50.00%	51.68%	**54.88%**	50.82%	50.00%	50.00%	47.81%
	Top-K	50.00%	50.86%	52.46%	**73.94%**	50.00%	50.00%	55.86%
	RD	0.00%	1.09%	4.18%	1.95%	**99.69%**	9.49%	6.99%
	AM	50.00%	50.04%	49.92%	50.12%	50.00%	**67.25%**	50.90%
	EDM	50.00%	51.91%	53.52%	50.82%	50.00%	50.00%	**57.62%**



(a) ImageNette (b) CIFAR-10

Fig. 4. Impact of the architecture choice for surrogate models on the attack performance.



(a) l_1 distance (b) l_2 distance (c) Value of the loss function

Fig. 5. The impact of loss functions on the generative model. We display the l_1/l_2 distance between the generated recovered query results and the corresponding original undisrupted query results and the value of the loss function after 20 epochs of training.

TABLE XI
AGREEMENT BETWEEN THE SUBSTITUTE MODEL AND DEFENSED VICTIM MODEL FOR MNIST, FASHIONMNIST, AND CIFAR-10.

Dataset	Budget	Defense	PP (ϵ)			RS (β)				AM (τ)	EDM	Hybrid	
			0.5	0.99	1.1	0.1	0.3	0.5	0.8	0.99		PP/RS	EDM+RD
MNIST	0.5k	KnockoffNet	0.17×	1.54 ×	8.40×	69.07×	59.18×	52.24×	43.03×	47.59×	37.65×	37.30×	12.05×
		D-KnockoffNet	2.14×	4.43×	12.63×	84.38×	82.82×	80.66×	76.81×	78.04×	38.25×	72.12×	17.81×
		JBDA	20.00×	16.61×	12.75×	11.86×	68.24×	26.64×	19.78×	79.63×	41.75×	65.81×	9.93×
		D-JBDA	23.20×	21.55×	21.01×	78.78×	78.56×	79.10×	74.30×	78.39×	40.94×	76.38×	28.05×
		ActiveThief	3.30×	9.90×	1.70×	35.40×	22.60×	15.50×	13.00×	35.84×	30.35×	22.69×	13.90×
		D-ActiveThief	9.80×	17.90×	81.90×	36.30×	37.40×	36.10×	35.40×	41.52×	41.80×	28.58×	21.16×
	5k	KnockoffNet	0.04×	3.01×	11.79×	86.41×	81.51×	77.01×	65.31×	74.20×	38.22×	57.59×	20.70×
		D-KnockoffNet	0.06×	8.00×	12.63×	96.52×	95.83×	94.90×	91.83×	83.94×	42.73×	63.40×	22.50×
		JBDA	68.14×	31.15×	20.96×	93.77×	85.90×	85.90×	82.50×	94.75×	34.82×	33.95×	4.46×
		D-JBDA	70.16×	37.88×	24.64×	96.79×	89.17×	87.30×	86.34×	98.04×	41.00×	46.09×	10.30×
		ActiveThief	88.90×	19.15×	3.30×	96.40×	95.40×	93.80×	91.70×	95.06×	53.15×	82.62×	5.59×
		D-ActiveThief	85.30×	84.20×	88.70×	98.80×	96.30×	94.50×	92.30×	98.70×	60.45×	90.07×	8.08×
	20k	KnockoffNet	0.01×	2.75×	6.08×	95.69×	91.66×	88.40×	78.44×	79.73×	56.29×	71.74×	23.16×
		D-KnockoffNet	2.45×	9.22×	15.27×	98.07×	97.78×	97.39×	96.81×	84.05×	56.16×	77.10×	30.11×
		JBDA	64.51×	43.73×	23.06×	91.07×	84.65×	84.65×	78.81×	90.84×	16.55×	63.05×	1.50×
		D-JBDA	74.17×	55.80×	33.33×	89.97×	89.08×	91.25×	81.96×	87.29×	25.40×	85.86×	7.73×
		ActiveThief	66.67×	12.21×	10.10×	96.30×	94.70×	94.30×	91.70×	95.16×	51.97×	84.88×	12.95×
		D-ActiveThief	77.22×	67.67×	77.98×	98.90×	95.60×	95.10×	93.60×	98.54×	56.35×	87.47×	10.89×
	60k	KnockoffNet	0.03×	2.03×	5.21×	97.67×	96.31×	93.16×	86.44×	83.05×	59.37×	72.91×	25.50×
		D-KnockoffNet	2.85×	9.64×	18.42×	98.60×	98.33×	98.23×	97.46×	87.93×	59.37×	81.60×	25.50×
		JBDA	65.34×	37.64×	22.39×	89.26×	84.38×	82.55×	77.29×	88.93×	15.47×	47.03×	3.40×
		D-JBDA	66.31×	34.54×	31.63×	89.47×	89.47×	80.03×	80.49×	90.02×	16.29×	68.30×	13.05×

Dataset	Budget	Defense	PP (ϵ)			RS (β)				AM (τ)	EDM	Hybrid	
			0.5	0.99	1.1	0.1	0.3	0.5	0.8	0.99		PP/RS	EDM+RD
FashionMNIST	0.5k	KnockoffNet	8.89×	8.13×	4.79×	29.86×	27.99×	26.67×	21.76×	20.14×	28.79×	16.94×	4.40×
		D-KnockoffNet	13.96×	9.60×	7.06×	40.30×	37.63×	37.44×	34.40×	33.86×	35.24×	28.12×	11.77×
		JBDA	23.30×	16.23×	12.42×	31.26×	31.59×	29.50×	26.11×	32.04×	23.30×	33.23×	10.66×
		D-JBDA	26.67×	18.97×	19.56×	34.81×	32.90×	34.14×	25.37×	34.95×	26.63×	34.02×	7.83×
		ActiveThief	10.17×	10.82×	10.48×	10.57×	10.49×	9.95×	9.26×	46.27×	32.22×	21.62×	2.40×
		D-ActiveThief	20.56×	20.11×	19.87×	24.75×	25.08×	25.26×	25.03×	49.75×	35.13×	22.65×	23.82×
	5k	KnockoffNet	2.27×	5.90×	4.69×	33.97×	32.16×	31.70×	30.01×	21.36×	21.41×	6.54×	3.33×
		D-KnockoffNet	11.10×	13.00×	16.93×	49.41×	46.37×	45.75×	41.92×	38.69×	31.04×	10.33×	10.70×
		JBDA	33.18×	16.11×	13.93×	38.02×	35.47×	30.76×	32.64×	44.29×	32.13×	21.39×	7.73×
		D-JBDA	30.78×	29.21×	30.10×	41.19×	42.54×	42.51×	33.18×	43.46×	33.01×	37.02×	10.10×
		ActiveThief	9.92×	11.30×	11.13×	23.97×	21.60×	24.39×	20.68×	61.08×	38.31×	13.03×	1.86×
		D-ActiveThief	12.34×	11.30×	12.33×	34.96×	33.49×	32.70×	30.17×	66.33×	44.82×	27.65×	11.44×
	20k	KnockoffNet	2.80×	3.52×	4.03×	39.63×	33.55×	32.44×	30.82×	34.85×	40.50×	11.33×	3.64×
		D-KnockoffNet	11.22×	19.10×	18.58×	62.52×	59.00×	57.98×	52.76×	48.04×	44.43×	22.11×	3.60×
		JBDA	36.61×	14.79×	15.52×	30.74×	30.70×	32.80×	35.08×	41.97×	22.64×	25.41×	4.44×
		D-JBDA	30.52×	29.81×	30.13×	48.08×	40.87×	31.31×	40.09×	43.26×	31.70×	40.76×	10.93×
		ActiveThief	7.87×	6.67×	3.78×	28.76×	24.08×	23.69×	21.49×	60.41×	38.26×	17.94×	3.32×
		D-ActiveThief	34.00×	48.67×	47.98×	40.27×	39.65×	41.76×	45.34×	68.98×	31.12×	37.84×	7.87×
	60k	KnockoffNet	0.87×	2.18×	2.96×	46.95×	35.63×	33.17×	31.22×	33.79×	41.93×	25.68×	1.98×
		D-KnockoffNet	1.36×	8.46×	26.57×	69.04×	63.80×	61.37×	56.79×	50.25×	31.50×	41.94×	3.04×
		JBDA	27.32×	17.17×	17.22×	29.15×	29.65×	30.76×	33.73×	42.09×	22.76×	28.90×	2.21×
		D-JBDA	29.42×	20.02×	21.02×	34.47×	29.89×	41.97×	34.75×	67.23×	29.82×	30.59×	10.44×

Dataset	Budget	Defense	PP (ϵ)			RS (β)				AM (τ)	EDM	Hybrid	
			0.5	0.99	1.1	0.1	0.3	0.5	0.8	0.99		PP/RS	EDM+RD
CIFAR-10	0.5k	KnockoffNet	5.91×	11.23×	22.31×	22.37×	18.20×	18.87×	18.11×	22.40×	34.13×	14.76×	15.51×
		D-KnockoffNet	7.35×	14.08×	6.41×	25.23×	23.67×	23.48×	23.00×	22.60×	37.71×	20.44×	10.34×
		JBDA	9.80×	13.40×	35.90×	14.10×	13.60×	12.40×	10.00×	11.70×	14.64×	11.86×	13.44×
		D-JBDA	10.80×	14.60×	83.30×	14.20×	16.20×	13.50×	12.30×	13.50×	12.97×	16.84×	16.11×
		ActiveThief	3.86×	6.38×	10.05×	10.00×	9.80×	9.80×	9.80×	9.70×	14.57×	8.77×	18.58×
		D-ActiveThief	9.82×	17.26×	49.63×	11.70×	10.90×	11.30×	11.10×	9.50×	30.67×	10.75×	23.55×
	5k	KnockoffNet	4.49×	12.04×	14.70×	38.53×	35.41×	30.60×	22.78×	36.80×	51.41×	13.17×	10.48×
		D-KnockoffNet	5.46×	13.39×	6.00×	36.03×	35.67×	34.52×	32.86×	34.10×	11.16×	22.70×	10.24×
		JBDA	10.00×	15.70×	34.70×	17.10×	16.40×	10.00×	10.00×	13.90×	11.51×	16.03×	5.66×
		D-JBDA	10.40×	14.50×	83.00×	18.20×	14.10×	9.80×	13.00×	12.00×	13.47×	20.80×	3.44×
		ActiveThief	12.70×	11.43×	13.28×	32.70×	31.50×	27.60×	23.80×	39.40×	44.57×	25.81×	13.86×
		D-ActiveThief	10.37×	14.29×	68.35×	32.60×	29.40×	28.70×	28.20×	38.80×	45.08×	28.71×	15.10×
	20k	KnockoffNet	2.52×	6.12×	11.86×	57.75×	50.82×	46.80×	36.37×	44.20×	61.51×	53.25×	13.79×
		D-KnockoffNet	6.94×	15.72×	7.83×	56.65×	54.01×	50.54×	40.88×	42.00×	51.56×	56.73×	16.26×
		JBDA	12.40 ×	11.80×	37.30×	10.00×	12.00×	12.40×	12.30×	16.40×	12.79×	11.96×	7.66×
		D-JBDA	11.30×	14.70×	58.30×	10.00×	13.00×	10.40×	12.50×	12.10×	10.07×	19.59×	2.33×
		ActiveThief	17.36×	15.86×	14.25×	32.40×	32.20×	29.10×	24.30×	40.63×	52.93×	37.86×	14.29×
		D-ActiveThief	13.94×	20.04×	70.31×	34.20×	30.10×	30.00×	29.40×	47.29×	52.51×	39.12×	22.71×
	60k	KnockoffNet	1.80×	5.85×	15.77×	67.37×	60.38×	56.97×	45.33×	57.80×	65.25×	42.75×	13.79×
		D-KnockoffNet	8.30×	16.28×	9.93×	68.08×	62.00×	57.88×	48.59×	61.36×	13.99×	46.91×	63.74×
		JBDA	4.70×	12.60×	39.20×	10.70×	10.80×	11.30×	12.20×	13.00×	11.94×	5.13×	4.58×
		D-JBDA	5.80×	13.60×	84.50×	11.60×	12.00×	11.60×	13.40×	13.40×	11.90×	11.13×	11.35×

TABLE XII
AGREEMENT BETWEEN THE SUBSTITUTE MODEL AND DEFENSED VICTIM MODEL FOR GTSRB.

Dataset	Budget	Defense	PP (ϵ)			RS (β)				AM (τ)	EDM	Hybrid	
			0.5	0.99	1.1	0.1	0.3	0.5	0.8	0.99		PP/RS	EDM+RD
GTSRB	0.5k	KnockoffNet	0.72×	4.99×	28.73×	7.74×	3.74×	3.12×	2.48×	30.96×	34.30×	1.21×	0.63×
		D-KnockoffNet	4.40×	4.70×	30.40×	26.80×	17.20×	11.20×	11.90×	41.39×	32.40×	18.22×	0.44×
		JBDA	15.87×	19.75×	27.50×	40.95×	38.29×	36.97×	33.26×	9.73×	25.90×	22.80×	1.10×
		D-JBDA	10.96×	9.75×	10.37×	48.58×	47.39×	45.31×	46.07×	11.29×	28.40×	32.23×	4.10×
		ActiveThief	5.70×	2.20×	97.60×	14.10×	9.60×	11.20×	10.40×	42.63×	26.80×	11.62×	0.30×
		D-ActiveThief	7.60×	9.10×	97.30×	15.20×	15.20×	14.60×	8.50×	47.28×	25.80×	15.45×	1.20×
	5k	KnockoffNet	0.70×	4.20×	16.65×	28.09×	13.00×	6.52×	2.16×	69.26×	39.20×	18.84×	0.96×
		D-KnockoffNet	14.50×	17.20×	47.20×	61.80×	48.50×	38.70×	21.30×	77.32×	54.30×	61.00×	0.92×
		JBDA	6.49×	8.36×	27.94×	31.69×	15.38×	18.40×	19.25×	8.21×	2.20×	25.57×	2.30×
		D-JBDA	25.76×	34.29×	36.17×	37.63×	21.80×	23.27×	21.80×	12.39×	2.70×	28.33×	3.22×
		ActiveThief	67.40×	49.50×	96.80×	43.60×	30.30×	42.30×	34.50×	70.60×	28.55×	60.62×	1.10×
		D-ActiveThief	70.20×	52.60×	98.10×	49.60×	48.50×	48.40×	49.20×	77.35×	28.10×	82.96×	0.99×
	10k	KnockoffNet	0.75×	5.05×	24.07×	37.30×	18.03×	9.55×	1.68×	75.26×	59.02×	6.38×	2.75×
		D-KnockoffNet	17.30×	23.20×	57.30×	71.90×	59.80×	50.20×	41.20×	81.37×	57.50×	70.07×	1.64×
		JBDA	5.08×	7.36×	21.98×	30.96×	14.28×	13.97×	10.25×	12.33×	6.80×	28.94×	1.70×
		D-JBDA	21.87×	24.36×	25.03×	38.86×	39.54×	37.64×	37.16×	15.75×	26.20×	33.44×	0.40×
		ActiveThief	86.70×	89.20×	97.60×	55.20×	20.40×	52.80×	42.70×	80.04×	28.60×	56.07×	0.92×
		D-ActiveThief	78.40×	74.70×	97.90×	67.90×	67.60×	67.40×	67.50×	83.26×	30.60×	86.10×	0.66×
	30k	KnockoffNet	0.64×	3.73×	16.77×	53.65×	26.88×	16.53×	1.33×	89.92×	64.42×	20.82×	1.20×
		D-KnockoffNet	20.95×	29.48×	68.39×	82.50×	74.50×	67.80×	52.00×	92.35×	26.10×	27.67×	1.01×
		JBDA	6.38×	7.04×	47.25×	28.06×	30.47×	10.37×	19.49×	9.64×	6.10×	20.22×	0.98×
		D-JBDA	29.74×	30.29×	28.46×	31.46×	33.83×	32.58×	29.96×	12.55×	2.10×	36.76×	6.66×

TABLE XIII
AGREEMENT BETWEEN THE SUBSTITUTE MODEL AND DEFENSED VICTIM MODEL FOR IMAGENETTE.

Dataset	Budget	Defense	PP (ϵ)			RS (β)				EDM	Hybrid	
			0.5	0.99	1.1	0.1	0.3	0.5	0.8		PP/RS	EDM+RD
ImageNette	0.1k	KnockoffNet	10.38×	19.36×	63.69×	75.06×	64.28×	49.94×	36.47×	46.00×	50.36×	7.60×
		D-KnockoffNet	8.39×	31.76×	87.39×	78.50×	71.90×	58.60×	70.10×	53.20×	55.99×	7.63×
		JBDA	79.83×	10.04×	17.37×	91.20×	85.90×	68.40×	30.30×	25.40×	82.09×	7.72×
		D-JBDA	79.67×	10.05×	17.34×	91.16×	85.95×	68.50×	30.33×	27.06×	89.20×	8.91×
		ActiveThief	5.80×	22.10×	50.40×	86.10×	72.30×	57.00×	31.80×	36.00×	40.24×	7.10×
		D-ActiveThief	6.10×	24.00×	83.30×	42.40×	46.20×	29.10×	24.40×	43.90×	41.48×	12.33×
	1k	KnockoffNet	18.31×	23.85×	50.29×	89.47×	80.25×	73.26×	69.85×	53.50×	29.92×	7.01×
		D-KnockoffNet	14.87×	35.96×	88.20×	92.30×	90.70×	87.40×	91.10×	56.94×	32.77×	8.43×
		JBDA	21.00×	19.40×	2.60×	28.80×	30.00×	30.30×	22.70×	27.00×	26.30×	9.10×
		D-JBDA	8.40×	12.80×	16.70×	21.30×	17.30×	17.70×	15.10×	33.43×	33.84×	10.50×
		ActiveThief	12.30×	24.70×	78.60×	93.60×	91.80×	92.30×	93.40×	44.30×	77.51×	7.01×
		D-ActiveThief	9.80×	23.60×	78.30×	84.30×	26.70×	19.90×	9.20×	52.70×	84.48×	14.72×
	5k	KnockoffNet	21.35×	28.46×	30.85×	89.63×	84.39×	83.45×	80.83×	49.80×	71.84×	7.44×
		D-KnockoffNet	15.38×	37.88×	90.84×	92.40×	92.70×	92.30×	92.50×	50.62×	79.80×	16.55×
		JBDA	11.60×	8.30×	3.30×	17.50×	19.90×	12.10×	11.30×	13.00×	13.10×	28.53×
		D-JBDA	9.60×	20.10×	49.70×	22.90×	15.60×	14.10×	13.10×	13.11×	19.81×	34.25×
		ActiveThief	13.70×	30.70×	77.60×	93.80×	92.30×	91.30×	89.70×	48.60×	72.76×	6.42×
		D-ActiveThief	3.60×	23.30×	75.80×	93.30×	58.80×	57.10×	65.70×	49.71×	73.43×	17.28×
	10k	KnockoffNet	21.44×	26.07×	29.97×	90.29×	90.86×	90.28×	89.94×	55.90×	33.81×	7.51×
		D-KnockoffNet	20.38×	35.87×	73.13×	92.80×	93.20×	92.80×	92.30×	56.04×	69.85×	16.73×
		JBDA	11.90×	13.90×	21.80×	13.20×	9.70×	9.90×	12.00×	7.00×	9.76×	7.43×
		D-JBDA	8.40×	13.70×	45.20×	21.10×	18.00×	17.80×	17.00×	13.34×	25.99×	10.30×