QUDA: Query-Limited Data-Free Model Extraction

Zijun Lin*†
Nanyang Technological University
Singapore
linz0048@e.ntu.edu.sg

Ke Xu*‡
Huawei International
Singapore
xuke64@huawei.com

Chengfang Fang
Huawei International
Singapore
fang.chengfang@huawei.com

Huadi Zheng
Huawei Technology
Shenzhen, China
zhenghuadi@huawei.com

Jaheezuddin Aneez Ahmed†
Nanyang Technological University
Singapore
aneezahm001@e.ntu.edu.sg

Jie Shi
Huawei International
Singapore
SHI.JIE1@huawei.com

ABSTRACT

Model extraction attack typically refers to extracting non-public information from a black-box machine learning model. Its unauthorized nature poses significant threat to intellectual property rights of the model owners. By using the well-designed queries and the predictions returned from the victim model, the adversary is able to train a clone model from scratch to obtain similar functionality as victim model. Recently, some methods have been proposed to perform model extraction attacks without using any in-distribution data (Data-free setting). Although these methods have been shown to achieve high clone accuracy, their query budgets are typically around 10 million or even exceed 20 million in some datasets, which lead to a high cost of model stealing and can be easily defended by limiting the number of queries. To illustrate the severe threats induced by model extraction attacks with limited query budget in realistic scenarios, we propose QUDA – a novel **QU**ey-limited **DA**ta-free model extraction attack that incorporates GAN pre-trained by public unrelated dataset to provide weak image prior and the technique of deep reinforcement learning to make query generation strategy more efficient. Compared with the state-of-the-art data-free model extraction method, QUDA achieves better results under query-limited condition (0.1M query budget) in FMNIST and CIFAR-10 datasets, and even outperforms the baseline method in most cases when QUDA uses only 10% query budget of its. QUDA issued a warning that solely relying on the limited numbers of queries or the confidentiality of training data is not reliable to protect model's security and privacy. Potential countermeasures, such as detection-based defense approach, are also provided.

CCS CONCEPTS

• **Security and privacy**; • **Computing methodologies** → **Machine learning**;

KEYWORDS

security for artificial intelligence, model extraction attack, adversarial machine learning

ACM Reference Format:
Zijun Lin, Ke Xu, Chengfang Fang, Huadi Zheng, Jaheezuddin Aneez Ahmed, and Jie Shi. 2023. QUDA: Query-Limited Data-Free Model Extraction . In *ACM ASIA Conference on Computer and Communications Security (ASIA CCS '23), July 10–14, 2023, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3579856.3590336

1 INTRODUCTION

With Artificial Intelligence increasingly embedded in enterprise tools, products, and services, companies who adopted AI in their business have grown and expanded steeply. AI becomes one of the most important factors for them to gain competitive advantages over other players. Companies usually need massive material and financial resources as well as a team of experts to build a high precision, commercially available deep learning model. Therefore, protecting the intelligent properties from competitors is essential to retain their advantage.

AI capability is typically deployed as a service that is available to the public. Since machine learning models have been shown to have the potential to extract similar models from available APIs [39], there are severe concerns about model extraction attacks against such services. A series of attacks and defence mechanisms have been proposed, and the concern is extended to more complicated models [27, 29, 36, 42]. Adversaries may use extracted models to mount other attacks, such as adversarial attack [12, 17], membership inference attack [37] or model inversion attack [46], which will further compromise the privacy and security of the victim models.

Recently, researchers proposed some methodologies utilizing GAN to extract image classification models with high clone accuracy in data-free setting [20, 33, 40], which means there is no in-distribution data needed during the whole process of model extraction. However, the query budgets of these methods are mostly over 10 million. Such a large number of queries are not practical in real life. According to major MLaaS (Machine Learning as a Service) platform providers like Google, Alibaba Cloud and Tencent Cloud, for most of image-related tasks, the approximate cost per 1,000

*Both authors contributed equally to this work.
†Work done during internship at Huawei International.
‡Corresponding author.

ASIA CCS '23, July 10–14, 2023, Melbourne, VIC, Australia

ACM ISBN 979-8-4007-0098-9/23/07…$15.00
https://doi.org/10.1145/3579856.3590336

queries ranges from 0.5 USD to 1.5 USD [1–3]. If the adversary queries the victim model more than 10 million times as in the lately proposed algorithm, the cost of performing model extraction would exceed 20,000 USD even without considering hyperparameter tuning. Hence, it is impractical for the adversary to extract models from others in this condition. These attacks using a plethora of queries could be easily defended by limiting the maximum number of times that a user can access the model. Our following experiments also show that the clone accuracy would decrease significantly when the query budget is constrained to less than 1 million.

As observed in these recently proposed approaches, launching a data-free model extraction attack without exploiting a large number of queries is still challenging for adversary. To overcome the challenge, we propose a novel model extraction attack, QUDA, aiming to extract the model in query-limited and data-free setting.

Technically, we find out that most of data-free model extraction methods utilize GAN to generate the query images that maximize the disagreement between victim model and clone model so that the clone model could learn more meaningful information and unexplored knowledge from the queries. This type of approach indeed improves the clone accuracy but the drawback is that the number of queries required for training GAN from scratch is quite high. To remedy this, we initially pre-train GAN using public unrelated dataset. It provides a weak image prior to facilitate the extraction process without any preliminary knowledge of the in-distribution data, which in this work denotes samples that are the same or similar to the victim dataset. Additionally, this approach saves lots of queries to train GAN during sampling process and also shows the ability to boost the clone accuracy.

Since the distribution of the base images generated by the pre-trained GAN is different from that of the victim dataset, it may not be the best strategy to query the victim model directly with them. QUDA uses deep reinforcement learning (DRL) methods – DDPG to guide the I-FGSM algorithm to do certain perturbations on the base images. The DRL agent is expected to output the desired soft labels of the victim model on query images based on the reward function, which encourages the perturbed query images to better explore the decision boundary and promote the diversity of the samples. In this way, we can reduce the number of redundant queries and stage model extraction attack in a more efficient and less costly way.

QUDA is evaluated with FMNIST and CIFAR-10 datasets and compared with the state-of-the-art data-free model extraction method – DFME [40]. The experiment results demonstrate the capability of QUDA to extract deep learning models under the constraints of limited queries and no prior knowledge of in-distribution data. When the query budgets are limited to 0.1M for both QUDA and DFME (0.5% query budget of DFME in original paper), QUDA's clone accuracy in various model architectures is on average 4.52 times better than DFME in FMNIST and 3.95 times better in CIFAR-10. Furthermore, QUDA still outperforms DFME in most cases when the query budget of DFME is expanded to 1 million (10 times query budget of QUDA). Additionally, when we relax the data-free setting to the situation where the adversary could access a small portion of victim dataset and use 500 in-distribution images to fine-tune the clone models, the clone accuracy of QUDA can be further improved by 4% on average and also remain higher than DFME, which use the same 500 images to fine-tune the clone models. We also conduct detailed ablation studies and draw several insightful conclusions that some parameter settings achieving good performance in training the deep learning models might not be applicable to the case of model extraction, such as the choice of optimizer, learning rate etc.

In summary, our key contributions are as follows.

- We propose QUDA, a novel query-limited data-free model extraction attack, which applies GAN to provide weak image prior and DRL to improve the query generation strategy.
- We conduct extensive experiments to demonstrate that QUDA can effectively extract different kinds of deep learning models and outperforms the state-of-the-art baseline method in query-limited and data-free setting.
- We show that the clone accuracy of QUDA can be further improved by being fine-tuned using a small portion of in-distribution data, which also proves the feasibility of QUDA in practical setting.
- We take an important step towards a more efficient way of performing model extraction attack, and issue a warning that solely relying on the limited numbers of queries or the confidentiality of training data is not a reliable way to protect models' security and privacy.

2 BACKGROUND

2.1 Model Extraction

Earlier works in model extraction focus on simple ML models like SVM or decision tree [39]. As Deep Learning gains popularity, recent works [23, 29, 42] start to explore complex deep neural networks. These works present the effectiveness of their attacking methodologies in computer vision and nature language processing.

As shown in Figure 1, the victim model F_v is treated as black-box in ME, only the predictions of given inputs are revealed to the adversary. To perform an effective model extraction attack, the adversary needs to build their own transfer set D_t by using the well-designed inputs and the predictions returned from the victim model and train the clone model F_c in this transfer set to replicate victim model's functionality as much as possible. Moreover, the disclose of the model functionality further jeopardizes the model, as the information facilitates adversarial attacks and leaks confidential information of the victim model.

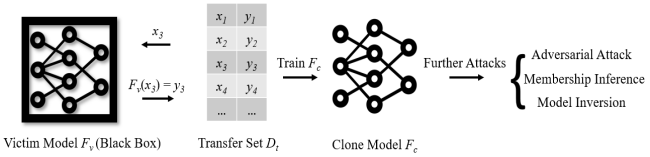

Figure 1: Overview of Model Extraction Attack.

Initial model extraction techniques use the in-distribution images to directly query the victim model [29, 31], which is impractical in cases where the adversary does not have any preliminary knowledge of the victim dataset. Some data-free model extraction methodologies are proposed recently to address this problem but they are mostly query-excessive [20, 33, 40], which can be defended by limiting the query budget. In our work, we overcome the barrier

and show that data-free model extraction attack is still feasible in query-limited (0.1M query budget) setting.

2.2 Generative Adversarial Network

Generative Adversarial Network (GAN) proposed by Ian Goodfellow et al. [8] is a method of unsupervised learning. GAN consists of a generator and a discriminator, and the learning process could be treated as these two neural networks contesting against each other in a zero-sum game. The generator takes random inputs from the latent space, and its output needs to simulate the real samples in the training set as closely as possible. The input of the discriminator is either the real sample or the output of the generator, and its purpose is to distinguish generator's output from the real sample.

Training GAN during sampling process tends to cost high query budget which might not be necessary in the case of model extraction. To save the budget, we initially train GAN by public unrelated dataset like the previous work [33] to constrain our generated queries into a more meaningful distribution and make model extraction attack more efficient.

2.3 Adversarial Perturbation

With a human-imperceptible synthesized noise [5] added to the input, adversarial examples are known to be capable of greatly affecting the output of neural networks. Due to its nature of manipulating the model output, adversarial examples are also used in model extraction [31] as a way to find decision-boundary. Notably, I-FGSM [25] is one of the most popular methods in generating adversarial examples due to its simplicity and efficiency.

Ilyas et al. [14] interpret the adversarial noise as non-robust features, which we adopt in our method to enrich the information retrieved from the victim model and push the distribution of the sampled data to a desired state. In our experiment, ϵ and y_{target} are guided by deep reinforcement learning algorithm to generate queries in a more diverse and efficient way.

2.4 Active Learning

Active learning (AL) is a technique where data is adaptively selected and trained [35, 49] based on the state of the evolving model. The purpose of active learning is to save data labeling effort, which, in the context of model extraction, is the interaction with the target black-box model. KnockoffNets [29] adopts active learning to select the samples from a surrogate dataset to query the victim model. However, several researches [20, 40] have shown that the extraction performance of this approach mainly depends on the availability of the appropriate surrogate dataset.

In this work, we use deep deterministic policy gradient (DDPG) [26] to guide active learning strategies and obtain the desired query image based on the distribution of predictions in transfer set. DDPG is a model-free and off-policy deep reinforcement learning algorithm for continuous and high-dimensional action spaces to learn competitive policies using low-dimensional observations. With the help of DDPG, I-FGSM algorithm is able to perturb the base images generated from GAN into more diverse and uncertain examples. In this way, we can encourage the exploration of the victim classifier's prediction space especially alongside the decision boundaries and reduce the redundancy across images in transfer set.

3 QUDA

In this paper, we consider the setting where an attacker intends to train a clone model F_c by querying a victim model $F_v : \mathcal{X} \rightarrow \mathcal{Y}$, which is trained on victim dataset D_v. The attacker has a query budget B and the set of query-prediction pairs he constructs are referred to the transfer set D_t.

We consider model extraction problem as a "Task Accuracy Extraction" as defined in [15]. The effectiveness of model extraction attack is evaluated with clone accuracy (1) and normalized clone accuracy (2) over a task distribution D_{task}. The equations of these two metrics are respectively shown below:

$$\Pr_{(x,y)\in D_{task}}[argmax(F_c(x)) = y] \tag{1}$$

$$\frac{\Pr_{(x,y)\in D_{task}}[argmax(F_c(x)) = y]}{\Pr_{(x,y)\in D_{task}}[argmax(F_v(x)) = y]} \tag{2}$$

Our goal is to maximize these two metrics under a highly constrained condition of model extraction.

- Query-limited: The number of queries to victim model F_v is limited to 0.1M (0.5% query budget of DFME).
- Data-free: No in-distribution data is used when performing model extraction while public unrelated dataset is allowed to use because it does not provide any task-related information but offers weak image prior and facilitates the extraction process.

3.1 Proposed Algorithm

To explain each process of QUDA more clearly, the overall algorithm is outlined in Algorithm 1.

Algorithm 1 QUDA Algorithm

Input: Victim Model F_v, Generator G, DRL Agent P
Parameter: Query Budget B, Evaluation Scope X, Training Scope Y, I-FGSM Iteration M, Learning Rate of Clone Model η, Epoch for Clone Model N
Output: Trained Clone Model F_c

Initialize P, F_c, $obs_0 \leftarrow 0$, $j \leftarrow 0$, $D_t \leftarrow \{\}$
while $j \leq B$ **do**
  $\epsilon, y_{target} \leftarrow P(obs_j)$
  $I \leftarrow G(z), z \sim \mathcal{N}(0, 1)$
  $I_j \leftarrow$ I-FGSM(ϵ, y_{target}) for M iterations
  $D_t \leftarrow D_t \cup \{(I_j, F_v(I_j))\}$
  $\theta_C \leftarrow \theta_C - \eta \nabla_{\theta_C} L_C$ with last Y samples in D_t
  Evaluate the distribution and the loss of F_c on the last X samples in D_t with equation (8), (9) and (10). Set the evaluation result as obs_j
  Calculate the reward r_j with equation (15)
  Update P and j
end while
repeat
  $(I_j, F_v(I_j)) \sim D_t$
  $\theta_C \leftarrow \theta_C - \eta \nabla_{\theta_C} L_C$
  $N \leftarrow N - 1$
until $N = 0$
return F_c

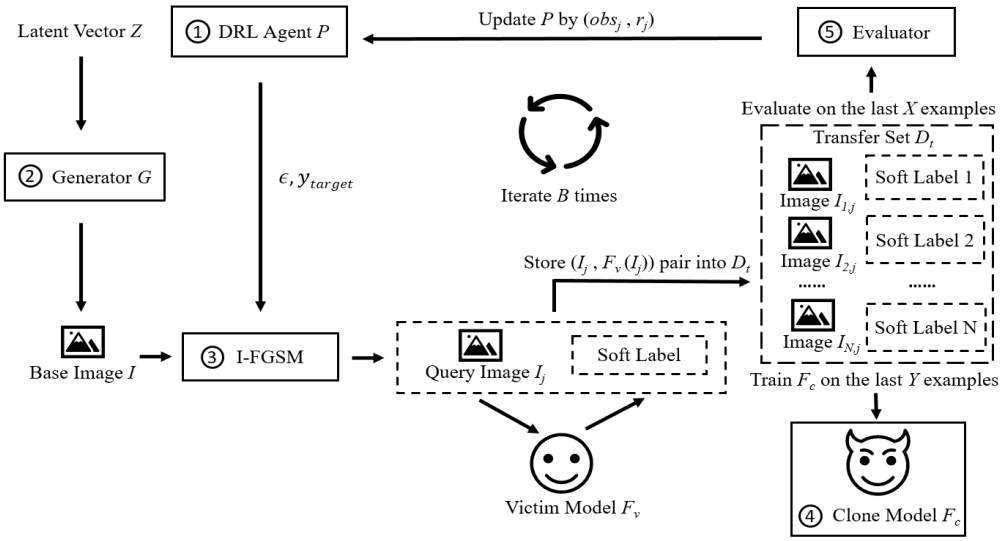

Figure 2: QUDA Framework. The five core modules in QUDA are labeled sequentially with numbers and framed by solid lines

With the above defined notations, we hereby illustrate QUDA framework. As shown in Figure 2, the framework consists of five loose-coupling modules and the detailed steps of each module in the j-th iteration are as follows:

(1) **DRL Agent P** gives the instruction ϵ and y_{target} to I-FGSM to generate required images.
(2) **Generator G** generates base images I by random noise input.
(3) **I-FGSM** crafts base images according to the instruction given by the DRL Agent. After the perturbation, the query images I_j are sent to the victim model F_v.
(4) **Clone Model F_c** is trained by adversary based on the last Y queries and returned predictions of the victim model. These input-output pairs $(I_j, F_v(I_j))$ are stored in transfer set D_t.
(5) **Evaluator** is deployed to observe the soft-label predictions of X lately stored queries in transfer set and provide feedback obs_j to the DRL Agent, which will be updated based on the reward r_j.

These steps will be iterated by B (query budget) times during the sampling process. The details of algorithm, functionality and design choices for each module will be illustrated in the following sections.

3.2 DRL Agent P

DRL agent in QUDA leads the process of active learning. In each iteration, the agent will be updated according to the reward function returned from the evaluator, which will be further discussed in Section 3.6. Based on the current extraction state, the DRL agent is expected to issue instructions for the system to execute.

For the choice of the DRL algorithm, DDPG [26], which has been proven to outperform human-level control [28] in many systems, is selected to learn active learning strategies and guide I-FGSM algorithm to perform certain perturbations on base images. DDPG implements a model-free and actor-critic reinforcement learning method [26], which searches for a policy that maximizes the expected cumulative long-term reward. It comprises of the ideas from deterministic policy gradient and deep Q-network, which support slow-learning target networks.

Given that the agent needs to determine adjustment action for noise value in I-FGSM, DDPG can be inherently used for training in continuous action space. Furthermore, DDPG algorithm implements off-policy method to update the Q values (i.e., state-action value) without following the policy used for current decisions to generate the data. In other words, it allows exploration outside of the samples created in the current strategy. Similar to Q-learning [11], if optimal Q value $Q^*(S, A)$ is provided with S as environment state and A as action, the optimal action $A^*(S)$ can be solved by maximizing the optimal Q value. Formally,

$$A^*(S) = \arg\max_A Q^*(S, A) \tag{3}$$

DDPG learns approximation of $Q^*(S, A)$ while alternatively finding approximation of $A^*(S)$. Specifically, for the critic network in each iteration i, it minimizes the cost function J_c across all n sampled experiences as follow,

$$J_c = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - Q\left(S_i, A_i\right)\right)^2 \tag{4}$$

As for the actor network, the expected discounted reward is maximized through

$$\begin{aligned} \nabla J_a &\approx \frac{1}{N}\sum_{i=1}^{N} M_{ai} M_{ci} \\ M_{ai} &= \nabla_A Q\left(S_i, A(S_i)\right) \\ M_{ci} &= \nabla A(S_i) \end{aligned} \tag{5}$$

where the gradient of the critic network (w.r.t. the action) computed by the actor network is denoted as M_{ci}, and the gradient of

the actor (w.r.t. the actor network) is M_{ai}. In addition, the gradients are evaluated for state S_i.

In QUDA, the DRL agent guided by DDPG outputs desired soft-label y_{target} and I-FGSM parameter ϵ for the next query. In addition to DDPG, the agent is also compatible with other continuous action space DRL algorithms such as TRPO [34]. Classic controlling algorithms such as MPC [4] may also be adopted in this scenario. The most efficient DDPG algorithm for model extraction with query constraints is subject to future research.

3.3 Generator G

Generator is responsible for generating the base image, which contributes in the next step of query image generation through I-FGSM. A deep de-convolutional generator is adopted to provide a deterministic function mapping a latent vector, z, to an image.

QUDA uses ImageNet [6], a large-scale public unrelated dataset, to pre-train the generator so that the base images would lie in a more meaninngful distribution. After pre-training, the generator is frozen, which means the newly created query-prediction pairs are not used to train the generator and the output of the generator is solely dependent on the pre-trained dataset. The final layer of the generator will be adjusted so that the dimensions of the generated base image match the input dimensions of the victim model.

Notably, the purpose of using pre-trained generator is to provide weak image prior to facilitate the extraction process instead of giving clone models any task-related information.

3.4 I-FGSM

I-FGSM mainly uses the model loss function to obtain the gradient of the input and the adversarial perturbation, which is created towards the direction of the loss function gradient $\nabla_x J(X_N^{adv}, y_{target})$). Afterwards, perturbations will be added to the original image X, generating adversarial samples X_N^{adv} through N iterations aimed at making the model misclassify it as the desired state y_{target}. When adopting I-FGSM, we set the upper and lower bound of value in each pixel based on ϵ to not exceed the perturbation range. The whole process of I-FGSM can be defined as:

$$X_0^{adv} = X \tag{6}$$
$$X_{N+1}^{adv} = Clip_{X,\epsilon}\{X_N^{adv} - \alpha sign(\nabla_x J(X_N^{adv}, y_{target}))\}$$

I-FGSM and DRL agent work hand-in-hand in QUDA. While the DRL agent is responsible for producing a set of meaningful posterior probabilities that would benefit the clone model's learning, the I-FGSM algorithm is responsible for adversarially perturbing the base image I to the query image I_j such that $F_v(I_j)$ at a particular step in the attack closely resembles the output probabilities of DDPG y_{target}. Each base image's perturbation extent ϵ is also controlled by the DRL agent for better learning strategy.

In this way, I-FGSM algorithm uses the output of the DRL agent at every step to manage the returned predictions of victim model such that the transfer set could be configured in a balanced and diverse manner. Furthermore, the cooperation of I-FGSM and DRL agent also maximizes the contribution of each query image in attacker's learning, which improves the efficiency of extraction process.

3.5 Clone Model F_c

In QUDA, the clone model is trained on Y most recently collected samples in each iteration of the sampling process to timely reflect the state of extraction. Considering the specialty of query data distribution, we use SGD as the optimizer, which will be discussed further in Section 5.2. The clone model is trained with cross-entropy loss between $F_c(x)$ and $F_v(x)$ (7).

$$L_C = L_{CE}(F_c(x), F_v(x)) \tag{7}$$

Noted that $F_c(x)$ and $F_v(x)$ indicate the probability (i.e., soft label) of F_c and F_v on input x. For better information utilization, QUDA additionally trains F_c for 100 epochs with transfer set after the sampling process.

3.6 Evaluator

In deep reinforcement learning, feedback from the environment to the agent is critical, which consists of two parts: observation *obs* and reward r. *obs* reflects the state or partial-state of the current environment, and r reflects how good did the agent do. Thus, the evaluator module gathers critical information about the current extraction process, then awards or penalizes the DRL agent. As shown in Algorithm 1, QUDA limits the evaluation scope to the latest X elements in the transfer set. With $D_{X,j}$ denoting the newly added X elements into the training set, i.e., $D_{X,j} = (D_{t,j} \setminus D_{t,j-X})$, the j-th *obs* contains the following information:

- The mean $\bar{m}_j$ (8) of the (soft) labels for each class in $D_{X,j}$.

$$\bar{m}_j = \frac{1}{X} \sum_{x \in D_{X,j}} F_v(x) \tag{8}$$

- The standard deviation d_j (9) of the (soft) labels for each class in $D_{X,j}$.

$$d_j = \sqrt{\frac{1}{X} \sum_{x \in D_{X,j}} (F_v(x) - \bar{m}_j)^2} \tag{9}$$

- Mean Cross-entropy Loss of F_c on each class (10) with $D_{X,j}$.

$$\bar{L}_j = \frac{1}{X} \sum_{x \in D_{X,j}} L_{CE}(F_c(x), F_v(x)) \tag{10}$$

Given *obs*, the agent decides an action that attempts to maximize the accumulated reward, calculated by a reward function. The reward function assesses whether the changes of the transfer set will lead to a better clone model, with respect to the newly added elements. To conduct model extraction attack more efficiently and effectively, QUDA encourages diversity in classes, and discourage similar samples to existing ones in D_t. The following equations are used for reward in Algorithm 1:

- To promote the diversity of samples, $\bar{m}_j$ is an important indicator to check the distribution of each class. Following reward equations are measured by both the range and standard deviation of $\bar{m}_j$:

$$r_{range} = \max \bar{m}_j - \min \bar{m}_j \tag{11}$$

Let $\sigma(m)$ denote the standard deviation of the given vector across all elements,

$$r_{std} = \sigma(\bar{m}_j) \tag{12}$$

Minimizing r_{range} and r_{std} ensures that the query images to be evenly distributed in each class, thus encouraging the samples' diversity in transfer set.

- To encourage the exploration of each class's decision boundary, value of d_j in obs is utilized to reflect the standard deviation of probabilities from each class.

$$r_{min} = \min d_j \tag{13}$$

By maximizing r_{min}, the variation of prediction in each class will be enlarged. In this way, the evaluator better encourages exploration of decision boundaries of the victim classifier.

- To generate difficult and uncertain examples, the cross-entropy loss of F_c for the last X elements is adopted in the reward system. By encouraging a higher cross-entropy loss, the agent learns to output probabilities that produce images the clone model and the victim model disagree on, which makes these images more meaningful for the clone model to learn from as it indicates that such images lie in the clone model's unexplored space. This also ensures that the generated images are not similar as the existing images in the transfer set. That is:

$$r_{ce} = \bar{L}_j \tag{14}$$

The final reward is the weighted sum of the above factors:

$$\begin{aligned} r_j = &\lambda_0 * r_{ce} + \lambda_1 * r_{min} \\ &-\lambda_2 * r_{range} - \lambda_3 * r_{std} \end{aligned} \tag{15}$$

Based on the above reward system, QUDA encourages that the DRL agent learns to promote diversity in the transfer set across all classes by reducing redundancy of query images. Moreover, the agent will motivate the exploration of the victim classifier's prediction space especially along the decision boundaries and unexplored regions. This DRL-based query generation strategy is expected to save unnecessary samples, reduce the query budget and improve the efficiency of model extraction attack without compromising the clone accuracy.

A maybe not-so-elegant method, which I think is just a variation of what some other people have said, is to just hardcode it. Many journals have a template that in some way allows for table footnotes, so I try to keep things pretty basic. Although, there really are some incredible packages already out there, and I think this thread does a good job of pointing that out.

4 EXPERIMENT

To demonstrate the effectiveness of QUDA, we compare QUDA with DFME [40] through clone accuracy and normalized clone accuracy. DFME is the state-of-the-art model extraction attack without using any in-distribution data. DFME utilizes GAN to maximize the disagreement between the predictions of victim model and clone model on each query and generate difficult examples for clone model. We choose DFME as the baseline method since it outperforms other methods, e.g., MAZE [20], as evaluated in their paper.

To illustrate the adaptability of QUDA, the hyperparameters of QUDA are not tuned to achieve the best performance while DFME's performance is highly sensitive to the choice of hyperparameters. Moreover, the experiment results show that QUDA can maintain high clone accuracy when the attacker has no knowledge about the victim model structure.

4.1 Experiment Setup

4.1.1 Datasets. The effectiveness of QUDA is evaluated with the following two datasets.

- FashionMNIST (FMNIST) [43] has 70,000 28×28 grayscale images in 10 classes, which contains 60,000 images for training and 10,000 images for testing. We use all 60,000 images to train our victim models and 10,000 images to test the accuracy of both the victim models and clone models.
- CIFAR-10 [24] contains 60,000 images in 10 classes. The size of the RGB images in CIFAR-10 is 32×32. The number of training samples and testing samples in CIFAR-10 is 50,000 and 10,000 respectively. The victim models are trained by all training samples in CIFAR-10, and the accuracy of both victim models and clone models are evaluated by 10,000 testing samples.

The dataset used to pre-train GAN is as follow.

- ImageNet [6] is one of the most popular datasets in computer vision. It contains more than 14 million images from over 20,000 categories.

The dataset of ImageNet we used to pre-train GAN in QUDA is not in the same distribution of the two victim datasets – FMNIST and CIFAR-10. Therefore, the base images generated from GAN is also from different distribution and thereby the adversary has no preliminary knowledge of victim dataset. The detailed discussions and samples of the query images are presented in Section 5.5.

4.1.2 Victim Models and Clone Models. The experiments are divided into two parts. In the first part, QUDA is evaluated with FMNIST and relatively simple victim models – LeNet and AlexNet, and the architecture of clone model is set to be the same as victim model. Due to the simplicity of the model architecture, QUDA performs model extraction without pre-training GAN (Table1). In the second part, the evaluation of QUDA is extended to more complex victim model architecture – AlexNet, VGG11 and ResNet18. In order to verify the generality of QUDA, we cross-validate these three models in FMNIST and CIFAR-10 (Table 2 & Table 3).

4.1.3 The Architecture of Generator. The generator uses four deconvolution layers followed by batch-normalization. Moreover, ReLU is the activation function used in all deconvolution layers except the last layer, where Tanh is adopted as activation unit to normalize the range of generated images to [-1,1]. The generator is pre-trained by ImageNet, so the dimension of last deconvolution layer of generator needs to be adjusted to match the size of two different datasets.

4.1.4 Hyperparameters of Training. To demonstrate the adaptiveness of QUDA, hyperparameters are set intuitively without being specifically tuned. QUDA chooses SGD as optimizer with maximum learning rate of 10^{-2}, momentum of 0.5, weight decay of 5×10^{-4} and batch size of 64 to train the clone model. All λ in the reward function are set to be 1. The query budget B is set to 0.1M. To encourage exploration, a Gaussian noise $n \sim N(0, 0.1)$ is applied to the DRL action for the first 2×10^3 steps. To balance the trade-off between attacker training and experiment run-time, we set the evaluation scope X to be 64 and the training scope Y to be 640. Meanwhile, each F_c is trained additionally for 100 epochs with D_t after the sampling process for better information utilization.

Table 1: Simple Models in FMNIST: Non-pre-trained GAN

D_v	F_v	$Acc(F_v)$ [1]	F_c	$QUDA(0.1M)$ [2]	$DFME(0.1M)$ [2]	$DFME(1M)$ [3]
FMNIST	LeNet	90.01	LeNet	72.63%(0.81×)	67.43%(0.74×)	77.89%(0.87×)
	AlexNet	89.8	AlexNet	80.05%(0.89×)	10.33%(0.12×)	46.33%(0.52×)

[1] The accuracy of victim models.
[2] The clone accuracy and the normalized clone accuracy when the query budget is 0.1 million.
[3] The clone accuracy and the normalized clone accuracy when the query budget is 1 million.

Table 2: Complex Models in FMNIST: Pre-trained GAN

D_v	F_v	$Acc(F_v)$	F_c	$QUDA(0.1M)$	$DFME(0.1M)$	$DFME(1M)$
FMNIST	AlexNet	89.8	AlexNet	83.82%(0.93×)	10.33%(0.12×)	46.33%(0.52×)
			VGG11	78.05%(0.87×)	10%(0.11×)	66.26%(0.74×)
			ResNet18	83.22%(0.93×)	73.23%(0.82×)	83.19%(0.93×)
	VGG11	90.25	AlexNet	73.84%(0.82×)	10%(0.11×)	26.26%(0.29×)
			VGG11	69.94%(0.77×)	10%(0.11×)	10%(0.11×)
			ResNet18	83.5%(0.93×)	68.75%(0.76×)	86.16%(0.95×)
	ResNet18	90.33	AlexNet	28.16%(0.31×)	10%(0.11×)	11.15%(0.12×)
			VGG11	41.57%(0.46×)	10%(0.11×)	10.1%(0.11×)
			ResNet18	14.3%(0.16×)	11.23%(0.12×)	19.77%(0.22×)

Table 3: Complex Models in CIFAR-10: Pre-trained GAN

D_v	F_v	$Acc(F_v)$	F_c	$QUDA(0.1M)$	$DFME(0.1M)$	$DFME(1M)$
CIFAR-10	AlexNet	89.12	AlexNet	61.91%(0.7×)	10%(0.11×)	10.1%(0.11×)
			VGG11	67.71%(0.76×)	17.71%(0.2×)	27.69%(0.31×)
			ResNet18	60.75%(0.68×)	12.59%(0.14×)	26.69%(0.3×)
	VGG11	84.3	AlexNet	43.93%(0.52×)	10%(0.12×)	12.4%(0.15×)
			VGG11	51.24%(0.61×)	14.91%(0.18×)	22.14%(0.26×)
			ResNet18	42.55%(0.5×)	17.98%(0.21×)	23.64%(0.28×)
	ResNet18	81.74	AlexNet	60.44%(0.74×)	11.41%(0.14×)	16.4%(0.2×)
			VGG11	70.3%(0.86×)	23.64%(0.29×)	36.55%(0.45×)
			ResNet18	68.99%(0.84×)	30.93%(0.38×)	34.97%(0.43×)

4.2 Results

We compare QUDA with the state-of-the-art data-free model extraction method DFME [40]. The experiment is firstly conducted in the same query budget (0.1M). Then the query budget of DFME is relaxed to 1M to find out whether QUDA could still outperform DFME using only 0.1 times the query budget. The hyperparameter settings of DFME are the same as in the original paper. Both the victim model and the clone model of DFME are structured in a manner consistent with QUDA for a fair comparison.

As shown in Table 1, the results indicate that even without pre-training the GAN, the normalized clone accuracy of QUDA when extracting the simple victim model is above 0.8× and the extraction performances of both cases are better than DFME under the same query budget setting (0.1M). Especially, QUDA significantly outperforms DFME by 7.7 times when the victim model is AlexNet and the query budget is 0.1M.

As shown in Table 2 and Table 3, for the extraction of complex models, QUDA also achieves higher clone accuracy in all experimental settings compared with DFME constrained by the same 0.1M query budget. QUDA's optimal performance can reach up to 7 times higher than DFME in FMNIST taking the victim model of AlexNet and the clone model of VGG11 as an example. In CIFAR-10, the best performance of QUDA is approximately 6 times higher than DFME when the victim model is ResNet18 and the clone model is AlexNet. On average, QUDA's clone accuracy in various model architectures is 4.52 times better than DFME in FMNIST and 3.95 times better in CIFAR-10.

Except for some specific settings of victim models (ResNet18 in FMNIST and VGG11 in CIFAR-10), the normalized clone accuracy of QUDA is mostly above 0.8× in FMNIST and 0.7× in CIFAR-10. As shown in Section 4.3 that QUDA provides a strong prior information about the victim model, and the clone accuracy is significantly improved after fine-tuning the clone models using 500 in-distribution samples.

Furthermore, QUDA still outperforms DFME in most cases when the query budget is only 10% of DFME. To be specific, the clone accuracy of QUDA is on average 2.46 times higher than DFME in FMNIST and 2.89 times higher in CIFAR-10 when the query budget of QUDA is set to 0.1M while DFME is set to 1M.

Table 4: Fine-tuned Simple Models in FMNIST

D_v	F_v	$Acc(F_v)$	F_c	$QUDA$	$QUDA(FT500)$ [1]	$DFME(FT500)$ [1]
FMNIST	LeNet	90.01	LeNet	72.63%(0.81×)	83.8%(0.93×)	83.4%(0.93×)
	AlexNet	89.8	AlexNet	80.08%(0.89×)	83.51%(0.93×)	80.08%(0.89×)

[1] The clone accuracy and the normalized clone accuracy when using 500 in-distribution images to fine-tune the clone models, which are both originally trained with a query budget of 0.1 million.

Table 5: Fine-tuned Complex Models in FMNIST

D_v	F_v	$Acc(F_v)$	F_c	$QUDA$	$QUDA(FT500)$	$DFME(FT500)$
FMNIST	AlexNet	89.8	AlexNet	83.82%(0.93×)	84.57%(0.94×)	80.08%(0.89×)
			VGG11	78.05%(0.87×)	83.32%(0.93×)	81.61%(0.91×)
			ResNet18	83.22%(0.93×)	85.32%(0.95×)	85.25%(0.95×)
	VGG11	90.25	AlexNet	73.84%(0.82×)	82.9%(0.92×)	80.34%(0.89×)
			VGG11	69.94%(0.77×)	83.27%(0.92×)	80.91%(0.9×)
			ResNet18	83.5%(0.93×)	86.29%(0.96×)	86.26%(0.96×)
	ResNet18	90.33	AlexNet	28.16%(0.31×)	81.81%(0.91×)	80.23%(0.89×)
			VGG11	41.57%(0.46×)	81.12%(0.9×)	81.89%(0.9×)
			ResNet18	14.3%(0.16×)	76.1%(0.84×)	81.73%(0.9×)

Table 6: Fine-tuned Complex Models in CIFAR-10

D_v	F_v	$Acc(F_v)$	F_c	$QUDA$	$QUDA(FT500)$	$DFME(FT500)$
CIFAR-10	AlexNet	89.12	AlexNet	61.91%(0.7×)	65.26%(0.73×)	39.45%(0.44×)
			VGG11	67.71%(0.76×)	68.78%(0.77×)	50.72%(0.57×)
			ResNet18	60.75%(0.68×)	66.11%(0.74×)	48.79%(0.55×)
	VGG11	84.3	AlexNet	43.93%(0.52×)	57.84%(0.69×)	40.23%(0.48×)
			VGG11	51.24%(0.61×)	60.45%(0.72×)	49.76%(0.59×)
			ResNet18	42.55%(0.5×)	57.35%(0.68×)	45.75%(0.54×)
	ResNet18	81.74	AlexNet	60.44%(0.74×)	62.82%(0.77×)	41.09%(0.5×)
			VGG11	70.3%(0.86×)	68.01%(0.83×)	51.98%(0.64×)
			ResNet18	68.99%(0.84×)	69.08%(0.85×)	50.64%(0.62×)

These experimental results indicate that QUDA has strong capability to stage effective model extraction in query-limited setting while the existing state-of-the-art method fails to achieve decent performances in most cases.

4.3 Fine-tuning Clone Models

In real life, it is possible for an adversary to obtain a small amount of in-distribution data. Although the amount of leaked information is negligibly small, it is still useful for the adversary to fine-tune their clone models. In this experiment, we use 500 in-distribution images, which is less than 1% of the entire training dataset of FMNIST and CIFAR-10, to evaluate the threat of model extraction attacks under the assumption that the adversary has access to a small portion of the victim dataset. The initial clone models of both QUDA and DFME are trained with a query budget of 0.1M.

The results are shown in Table 4, 5 and 6. The clone accuracy of QUDA is improved by 4% on average for all the clone models in two datasets. Particularly, there is a significant performance upgrade on the victim model ResNet18 in FMNIST and VGG11 in CIFAR-10, with an average clone accuracy improvement of 130%. Although the initial performance of these two victim model settings is lower than the others in QUDA, the huge boost of accuracy indicates that they have rich prior knowledge of the victim models and great potential for further improvement with the help of fine-tuning.

Furthermore, QUDA still achieves higher clone accuracy compared to DFME when both of their clone models are fine-tuned using the same 500 in-distribution images (approximately 2% higher in FMNIST and 17.48% higher in CIFAR-10 overall). These results further demonstrate the power of applying QUDA to real-world model extraction attacks.

5 ABLATION STUDIES

In this section, we would like to discuss some implementation choices and the reasons for adopting them in our experiments. We also step further to investigate the results in hard label setting and the query images to give more insights into QUDA.

In the analysis, the victim model is set as AlexNet, and the clone models are AlexNet, VGG11, ResNet18 to evaluate the factors affecting the clone accuracy, which will be applied in the discussion from Section 5.1 to Section 5.4

5.1 Importance of Experience Replay

After the sampling process, we retrain the clone model by 100 epochs using the constructed transfer set. The intuition is to review all generated samples to avoid overfitting the clone model with the most recently generated samples. To prove the importance of performing experience replay, we conducted experiments to compare the difference in accuracy between the clone models with and without performing experience replay.

The experimental results in Figure 3 show that the accuracy of clone models are significantly improved by approximately 20% on average after experience replay is performed. The reason behind the improvement is that performing experience replay after the sampling process helps QUDA avoid catastrophic forgetting [9], where the clone model learn recently generated samples in transfer set, but almost forget the previously trained samples.

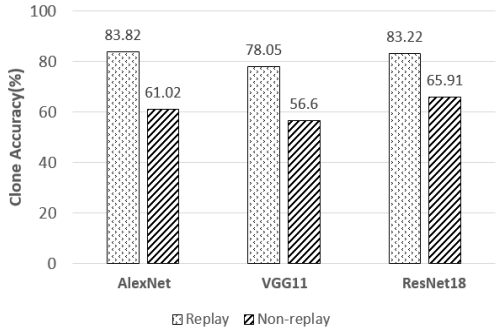

Figure 3: Impact of Experience Replay on QUDA.

5.2 Choice of Optimizer

The choice of optimizer needs to be adaptively configured for training F_c. One significant difference for optimizer choice compared to the typical machine learning is that, overfitting the training accuracy might not be adverse for testing accuracy. In QUDA, F_c are trained for a large number of epochs with little learning rate decay, which by common sense will lead to overfitting, but the test accuracy is observed to be continuously improving. We believe that in the scenarios where the adversary does not have a good prior knowledge about the dataset, the best strategy is to overfit to the soft labels given by F_v.

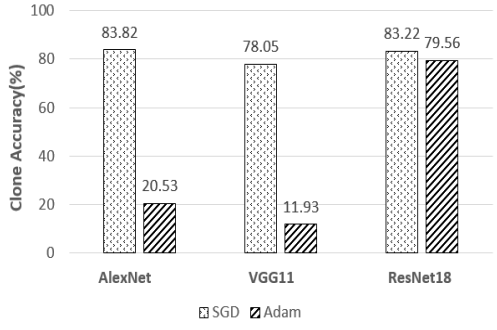

Figure 4: Impact of Optimizer Choice on QUDA.

One interesting observation shown in Figure 4 is that Adam optimizer [22] always leads to a lower clone accuracy than SGD does and has significantly worse performances in some cases.

Thus, the choice of optimizer is counter-intuitive in model extraction because the best strategy of the attacker is to overfit the output of victim model. The normal setting for training a neural network from scratch might not be applicable for the context of model extraction.

5.3 Impact of Pre-training GAN

Most of existing data-free model extraction methods train GAN from scratch in their algorithm [20, 40]. However, these approaches often cost unnecessarily large query budget budgets, as millions of queries are needed to make GAN generate useful images that are beneficial for model extraction. To improve the efficiency, QUDA pre-trains GAN by public unrelated dataset to provide a weak image prior instead of spending a large number of queries to train the GAN as existing methods do.

Figure 5 indicates that providing a weak image prior is sufficient to allow adversaries to conduct effective model extraction attack, as all three cases show an increase in clone accuracy after pre-training GAN. In all three cases, the clone model of ResNet18 shows the most significant improvement, with an increase in clone accuracy of 21.25%. Hence, this is a feasible approach to save the query budget without degrading the effectiveness of model extraction attack.

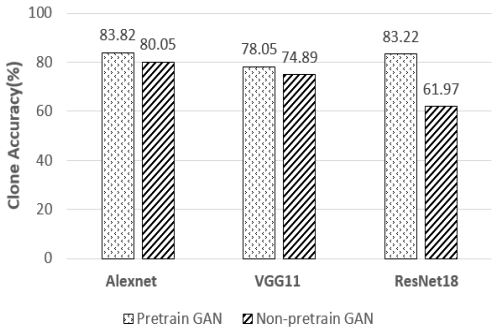

Figure 5: Impact of Pre-training GAN on QUDA.

5.4 QUDA in Hard Label Setting

To mitigate the threat of model extraction and protect the privacy, some MLaaS platforms restrict the outputs of their models to only provide top-k predictions or even top-1 prediction. Although such a defence may limit the normal usage of benign users, we would like to evaluate the extent to which QUDA performance is affected by this more restrictive condition, i.e., the hard label setting.

The results in Figure 6 show that the clone accuracy is still higher than 70% for all three cases, but there is a different degree of decrease compared with soft label setting (3.24% decrease in the clone model of AlexNet, 7.41% in VGG11 and 13.13% in ResNet18). It can be observed that the less complex the clone model is, the less the clone accuracy is affected. We also extend our experiments to simpler clone models – LeNet and the clone accuracy decrement is less than 2% in hard label setting, which further demonstrates

the conclusion. The decrease of performance in different degrees is expected because the soft label contains much richer information and the adversary could utilize the relative difference of prediction probabilities between each class to perform more powerful attacks.

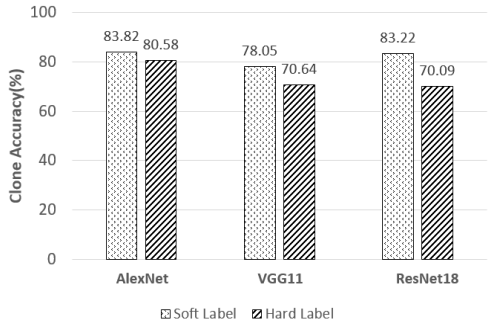

Figure 6: Soft Label vs Hard Label on QUDA.

5.5 Query Images

We investigate what images are used to query the victim models. Some representative examples of query images are shown in Figure 7. We randomly select query images from nine different classes (the first to ninth rows correspond to classes 0 to 8 respectively) in both FMNIST and CIFAR-10. In FMNIST, the victim model and clone model are AlexNet and ResNet18 respectively. In CIFAR-10, we choose ResNet18 as victim model and VGG11 as clone model.

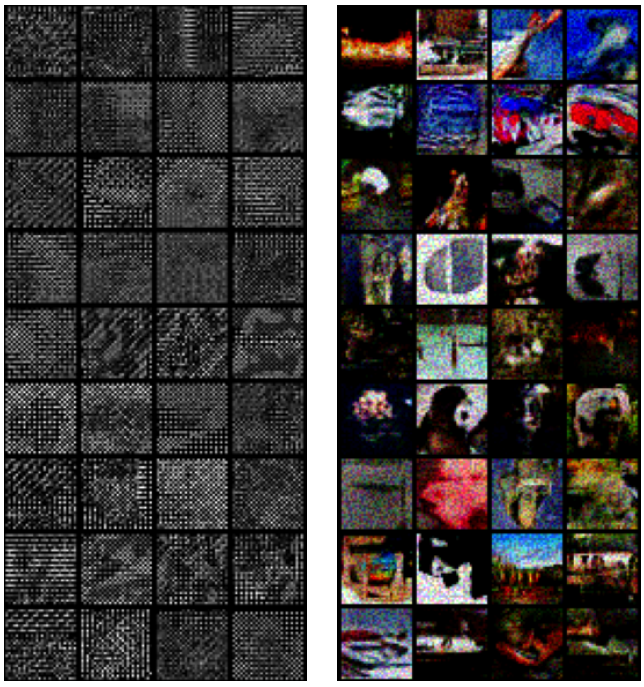

Figure 7: Samples of Query Images in FMNIST (left) and CIFAR-10 (right).

These query images are generated from generator, then perturbed by the I-FGSM algorithm guided by the DRL Agent. As illustrated in Figure 7, the images of these queries are outside the distribution ranges. The normalized clone accuracy of both representative clone models are still higher than 0.85×. This demonstrates the capability of QUDA to extract models without any prior information related to victim dataset. Therefore, in order to perform an effective model extraction attack, it is not necessary to make the distribution of query images as close as possible to the distribution of the victim dataset, as in the existing approach [20]. Moreover, when the adversary is faced with a limited query budget, it is impractical to train GAN from scratch to generate the query images which are visually similar to the training samples of victim models. From the perspective of query images, it could also be concluded that pre-training GAN by unrelated public dataset is a feasible way to save the budget and perform more efficient attack.

6 DISCUSSION

In this section, we discuss the future extension of QUDA and potential countermeasures. In the experiment using only hard label, there are different clone accuracy decreases against model architectures, which indicates potential improvement area for agent selection. As current agent mainly considers the reward evaluated from one clone model, architecture-aware agent [13] can be more robust when extracting the decision boundary of the victim model.

QUDA has demonstrated that, it is dangerous to assume model's confidentiality even if the distribution of the training dataset is unknown to the attacker. The defender might further constrain the query limit to reduce the effectiveness of QUDA but the number of queries required by QUDA is quite small, this may affect user experience. The constrained amount of queries also renders most of the model watermarking techniques not effective against QUDA. Therefore we suggest defenders to take a more proactive approach when their models are valuable. Our experiment with hard label shows that, discarding output soft-label information while maintaining the top-1 accuracy may have limited effect, and perturbation methods such as BDPL [47][48] or targeted poisoning noise [30] would become a trade-off between accuracy and defence-effectiveness.

On the other hand, as query images of QUDA is generated with a frozen pre-trained generator, the distribution of these generated query images is different from those by legitimate users, and the detection based defence approach seems more appropriate to defend QUDA. To differentiate the query distribution, the defender need to be able to identify the user ID for the queries, and the system or business model should discourage creating massive accounts with certain means. Then the defender can track the distribution of queries for each user with techniques such as PRADA [18].

7 RELATED WORK

7.1 Model Extraction

Tramer et al. [39] first introduced the concept of model extraction attack and demonstrated the feasibility of stealing online deployed machine learning models with outstanding fidelity. Since then, there have been many studies in recent years focusing on model extraction attacks in different domains [32, 38, 44]. For example, KnockOffNets [29] and JBDA [31] were proposed to launch

model extraction attack under the assumption that adversary was able to obtain some in-distribution data or similar data as a surrogate dataset and used these data combined with the techniques of active learning or data augmentation to query the victim model. While this is reasonable when attackers try to extract models with common tasks, it might not be the cases for some of commercial critical tasks, where training data is highly confidential.

Several researches have addressed this challenge and managed to extract models in data-free setting, which prohibited the adversary from having any prior information about the victim dataset. Roberts et al. [32] explored the possibility of extracting models using various distributions of noise. The results were decent when the method was implemented on simple victim datasets, while it failed to show good capabilities when stealing models trained on relatively complex datasets, as evaluated in some studies [20].

Inspired by data-free knowledge distillation [7], MAZE [20] and DFME [40] were another two recently proposed data-free model extraction attack methods. They trained GAN to generate the queries aiming to maximize the disagreement between victim model and clone model. The gradients of the victim models were estimated by zeroth-order gradient approximation to measure the degree of disagreement and perform back-propagation to tune the parameters of the generator. The clone model was also trained to match the prediction of the victim model. This kind of approach extracted the deep learning model with high clone accuracy and also showed the effectiveness when the victim model only returned hard labels [33].

7.2 Defence Mechanisms against Model Extraction

The vulnerability of deep learning models to model extraction attack will pose a huge threat when the companies apply them to the real-life environments. To tackle this problem, defence mechanism against model extraction is necessary. The goal of the defence mechanism is to reduce the accuracy of clone model without sacrificing victim model's functionality or to flag an attack without affecting the utility of benign users.

Generally, there are two categories of model defence mechanism against model extraction attack, namely detection and perturbation. Monitor-based detection flags an attack if the distribution of the queries deviates too much from a normal distribution [18]. Kesarvani et al. [21] designed an extraction monitor that continuously reported coverage estimates of the data space to the model owner based on the samples queried by the users.

On the other hand, the perturbation method can be classified into input perturbation and output perturbation. For the input perturbation, Guiga et al. [10] proposed a method that led to noisier extracted weights for the attacker by adding random noise to the pixels which had relatively lower values in guided grad-CAM during inference phrase. For perturbation of output, Kariyappa et al. [19] introduced a defence algorithm that returned incorrect predictions for out of distribution queries. In addition, Orekondy et al. [30] perturbed the predictions to poison the adversary's gradient signal, so that if the adversary followed the prediction result of the victim model, a low accuracy clone model would be obtained.

However, the defence mechanisms mentioned above somehow compromise the utility of benign users by providing them with false prediction results under certain conditions. Hence, it is necessary to propose solid model defence methods in future work to counteract model extraction without affecting normal usage.

7.3 Model Watermark

Model watermark is the following step for protecting the intellectual property of deep learning models after model defence. It can be regarded as an alternative option to safeguard the security of the model especially when model defence is difficult to implement. To watermark a model, the model owner needs to embed secret message when training the source model and ensure this message can be subsequently extracted with a secret key.

Uchida et al. [41] chose weights in one of the convolution layers to embed the watermark, which is a white-box watermark embedding method. However, considering models are usually deployed for remote service, white-box watermark is not practical because it requires model owners to access all the parameters of the surrogate model when verifying the ownership.

To embed watermark in black-box setting, Zhang et al. [45] embedded the backdoor in some training data as trigger set and claimed ownership of the model by demonstrating knowledge of the special input-output pairs, which were only known by model owners. Furthermore, Jia et al. [16] entangled the activation neuron of watermark data with legitimate task data so that attackers were forced to learn watermark when doing model extraction.

8 CONCLUSIONS

In this paper, we study the model extraction problem when the adversary has limited query budget and lacks prior knowledge about in-distribution training data. We propose QUDA - a novel model extraction attack achieving effective and efficient extraction in query-limited and data-free setting. Compared with the state-of-the-art data-free model extraction method DFME, QUDA is proven to perform better even with lower query budget. Hence, it verifies that, although the difficulty of extraction is significantly higher, an attacker can still launch model extraction successfully and compromise model confidentiality. Therefore, companies offering MLaaS should realize that their model is vulnerable when its confidentiality depends only on the limit of the number of queries or the confidentiality of the data. Furthermore, we would like to point out that, in addition to clone accuracy, query budget of the attacker should be considered when we evaluate the effectiveness of model extraction method since an excessive budget is impractical to apply in real situations. We hope our work could be a step towards efficient data-free model extraction attack and encourage future research efforts to detect such attack without compromising the utility of the model for benign users.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their constructive feedback.

REFERENCES

[1] [n. d.]. Alibaba Cloud Content Security Detailed Pricing Information. https://cn.aliyun.com/price/product?from_alibabacloud=#/lvwang/detail/cdibag
[2] [n. d.]. Pricing | Cloud Vision API | Google Cloud. https://cloud.google.com/vision/pricing/

[3] [n. d.]. Tencent Cloud Billing items. https://cloud.tencent.com/document/product/1235/44663

[4] Brandon Amos, Ivan Jimenez, Jacob Sacks, Byron Boots, and J Zico Kolter. 2018. Differentiable mpc for end-to-end planning and control. In *Advances in Neural Information Processing Systems*. 8289–8300.

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

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 248–255. https://doi.org/10.1109/CVPR.2009.5206848

[7] Gongfan Fang, Jie Song, Chengchao Shen, Xinchao Wang, Da Chen, and Mingli Song. 2019. Data-Free Adversarial Distillation.

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[9] Ian J. Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. 2013. An Empirical Investigation of Catastrophic Forgetting in Gradient-Based Neural Networks. https://arxiv.org/abs/1312.6211

[10] L. Guiga and A. W. Roscoe. 2020. Neural network security: Hiding CNN parameters with guided grad-CAM.. In *In ICISSP 2020 - Proceedings of the 6th International Conference on Information Systems Security and Privacy*. 611–618.

[11] Hado Hasselt. 2010. Double Q-learning. *Advances in neural information processing systems* 23 (2010).

[12] Xuanli He, Lingjuan Lyu, Qiongkai Xu, and Lichao Sun. 2021. Model Extraction and Adversarial Transferability, Your BERT is Vulnerable!. In *Proceedings of NAACL-HLT*.

[13] Xing Hu, Ling Liang, Shuangchen Li, Lei Deng, Pengfei Zuo, Yu Ji, Xinfeng Xie, Yufei Ding, Chang Liu, Timothy Sherwood, and Yuan Xie. 2020. DeepSniffer: A DNN Model Extraction Framework Based on Learning Architectural Hints. In *ASPLOS '20: Architectural Support for Programming Languages and Operating Systems*, James R. Larus, Luis Ceze, and Karin Strauss (Eds.).

[14] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. 2019. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*. 125–136.

[15] Matthew Jagielski, Nicholas Carlini, David Berthelot, Alex Kurakin, and Nicolas Papernot. 2020. High Accuracy and High Fidelity Extraction of Neural Networks. In *Proceedings of the 29th USENIX Conference on Security Symposium*.

[16] H. Jia, C. A. Choquette-Choo, and N. Papernot. 2021. Entangled watermarks as a defense against model extraction. In *23th {USENIX} Security Symposium ({USENIX} Security 21)*.

[17] Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT Really Robust? A Strong Baseline for Natural Language Attack on Text Classification and Entailment. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 05 (Apr. 2020), 8018–8025. https://doi.org/10.1609/aaai.v34i05.6311

[18] Mika Juuti, Sebastian Szyller, Samuel Marchal, and N. Asokan. 2019. PRADA: Protecting Against DNN Model Stealing Attacks.. In *In IEEE European Symposium on Security and Privacy, EuroS&P*. 512–527.

[19] Sanjay K. and Moinuddin K. Qureshi. 2020. Defending Against Model Stealing Attacks With Adaptive Misinformation.. In *In IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*. 767–775.

[20] Sanjay Kariyappa, Atul Prakash, and Moinuddin K. Qureshi. 2021. MAZE: Data-Free Model Stealing Attack Using Zeroth-Order Gradient Estimation. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), 13809–13818.

[21] Manish Kesarwani, Bhaskar Mukhoty, Vijay Arya, and Sameep Mehta. 2018. Model Extraction Warning in MLaaS Paradigm. In *Proceedings of the 34th Annual Computer Security Applications Conference* (San Juan, PR, USA) *(ACSAC '18)*. Association for Computing Machinery, New York, NY, USA, 371–380.

[22] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1412.6980

[23] Kalpesh Krishna, Gaurav Singh Tomar, Ankur P Parikh, Nicolas Papernot, and Mohit Iyyer. 2020. Thieves on sesame street! model extraction of bert-based apis. (2020).

[24] Alex Krizhevsky and Geoffrey Hinton. 2009. *Learning multiple layers of features from tiny images*. Technical Report 0. University of Toronto, Ontario.

[25] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2016. Adversarial examples in the physical world. (07 2016).

[26] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. [n. d.]. Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations, ICLR, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.).

[27] Yupei Liu, Jinyuan Jia, Hongbin Liu, and Neil Zhenqiang Gong. 2022. StolenEncoder: Stealing Pre-Trained Encoders in Self-Supervised Learning *(CCS '22)*. Association for Computing Machinery, New York, NY, USA, 2115–2128.

[28] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (Feb. 2015), 529–533. http://dx.doi.org/10.1038/nature14236

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

[30] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. 2020. Prediction Poisoning: Towards Defenses Against DNN Model Stealing Attacks.. In *In International Conference on Learning Representations, ICLR, Virtual Event*.

[31] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*. 506–519.

[32] Nicholas Roberts, Vinay Uday Prabhu, and Matthew McAteer. 2019. Model Weight Theft With Just Noise Inputs: The Curious Case of the Petulant Attacker. *ArXiv* abs/1912.08987 (2019).

[33] Sunandini Sanyal, Sravanti Addepalli, and R. Venkatesh Babu. 2022. Towards Data-Free Model Stealing in a Hard Label Setting. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022), 15263–15272.

[34] John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. 2015. Trust Region Policy Optimization. *CoRR* abs/1502.05477 (2015). arXiv:1502.05477 http://arxiv.org/abs/1502.05477

[35] Burr Settles. 2009. *Active learning literature survey*. Technical Report. University of Wisconsin-Madison Department of Computer Sciences.

[36] Zeyang Sha, Xinlei He, Ning Yu, Michael Backes, and Yang Zhang. 2022. Can't Steal? Cont-Steal! Contrastive Stealing Attacks Against Image Encoders. https://arxiv.org/abs/2201.07513

[37] R. Shokri, M. Stronati, C. Song, and V. Shmatikov. 2017. Membership Inference Attacks Against Machine Learning Models. In *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, Los Alamitos, CA, USA, 3–18. https://doi.org/10.1109/SP.2017.41

[38] Sebastian Szyller, Vasisht Duddu, Tommi Gröndahl, and N. Asokan. 2021. Good Artists Copy, Great Artists Steal: Model Extraction Attacks Against Image Translation Generative Adversarial Networks. https://arxiv.org/abs/2104.12623

[39] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. 2016. Stealing machine learning models via prediction apis. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*. 601–618.

[40] Jean-Baptiste Truong, Pratyush Maini, Robert J. Walls, and Nicolas Papernot. 2021. Data-Free Model Extraction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[41] Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh. 2017. Embedding watermarks into deep neural networks. In *In Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*. 269–277.

[42] Dongdong Wang, Yandong Li, Liqiang Wang, and Boqing Gong. 2020. Neural Networks Are More Productive Teachers Than Human Raters: Active Mixup for Data-Efficient Knowledge Distillation from a Blackbox Model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1498–1507.

[43] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. https://arxiv.org/abs/1708.07747

[44] Zhenrui Yue, Zhankui He, Huimin Zeng, and Julian McAuley. 2021. Black-Box Attacks on Sequential Recommenders via Data-Free Model Extraction. In *Proceedings of the 15th ACM Conference on Recommender Systems*. Association for Computing Machinery, New York, NY, USA, 44–54.

[45] J. Zhang, Z. Gu, J. Jang, H. Wu, M. P. Stoecklin, H. Huang, and I. Molloy. 2018. "Protecting intellectual property of deep neural networks with watermarking. In *In Proceedings of the 2018 on Asia Conference on Computer and Communications Security*. 159–172.

[46] Y. Zhang, R. Jia, H. Pei, W. Wang, B. Li, and D. Song. 2020. The Secret Revealer: Generative Model-Inversion Attacks Against Deep Neural Networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, Los Alamitos, CA, USA, 250–258. https://doi.org/10.1109/CVPR42600.2020.00033

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

[48] Huadi Zheng, Qingqing Ye, Haibo Hu, Jie Shi, and Chengfang Fang. 2020. Protecting decision boundary of machine learning model with differentially private perturbation. In *IEEE Transactions on Dependable and Secure Computing*. IEEE.

[49] Zongwei Zhou, Jae Shin, Lei Zhang, Suryakanth Gurudu, Michael Gotway, and Jianming Liang. 2017. Fine-tuning convolutional neural networks for biomedical image analysis: actively and incrementally. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 7340–7351.