CloudLeak: DNN Model Extractions from Commercial MLaaS Platforms

Yier Jin[*], Tsung-Yi Ho[**], and Honggang Yu[*]
[*]University of Florida, [**]National Tsing Hua University
Email: yier.jin@ece.ufl.edu

1 Introduction

Deep Neural Networks (DNN) have been widely deployed for a variety of tasks across many disciplines including image processing, natural language processing, and voice recognition. However, creating a successful DNN model depends on the availability of large amounts of data as well as enormous computing power, not to mention that the model training process is often an arduously slow process. This presents a large barrier to those interested in utilizing a DNN. To meet the demands of users who may not have sufficient resources, cloud-based deep learning services arose as a cost-effective and flexible solution, allowing users to complete their machine learning (ML) tasks efficiently. Machine Learning as a Service (MLaaS) platform providers may spend great effort in collecting data and training models, and thus want to keep trained models proprietary. The DNN models of MLaaS platform can only be used as web-based API interface and thus is isolated from users. In this work, we develop a novel type of attack that allows the adversary to easily extract the large-scale DNN models from various cloud-based MLaaS platforms, which are hosted by Microsoft, Face++, IBM, Google and Clarifai. We argue that existing model protection methods may not be sufficient in protecting the models in cloud and more research should be performed before MLaaS platform can be safely used.

2 Model Theft Attacks

Our work identified a fundamental vulnerability of Machine Learning as a Service (MLaaS) in commercial clouds. That is, although these services are constructed based on Deep Neural Networks with millions of parameters, attackers can easily replicate the DNN model, making this commercial model less profitable.

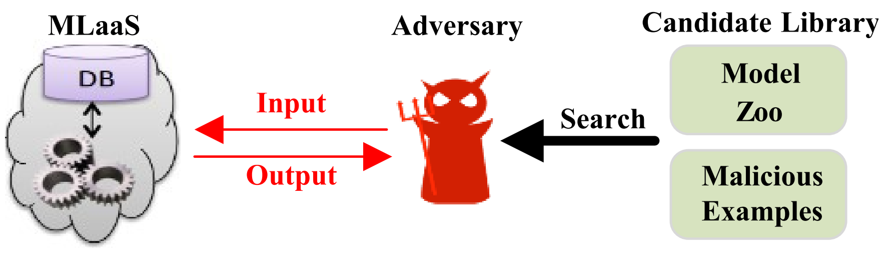

Figure 1: Illustration of our MLaaS model stealing attacks.

The key idea of our attack is to use input-output pairs obtained by querying such black-box APIs with specially crafted inputs to retrain the substitute models (see Figure 1).

From the technical perspective, we present a new DNN adversarial attack method, FeatureFool, against the local substitute models that adopts the features from model's internal hidden layers for generating a subset of training samples. These training samples are used to query the target model in order to efficiently learn the distance between decision boundaries of the target model and of the stolen model. By leveraging this new DNN adversarial attack method, we design an efficient and effective black-box model stealing attack targeting the large-scale DNN models provided by commercial platforms. In comparison to existing attack methods, we significantly reduce the number of queries required to steal the target model by incorporating various algorithms, including active learning, transfer learning, and adversarial examples.

The experimental results show that our model stealing attack can successfully build a local substitute model with similar performance to the target model with a small set of queries. An adversary can use this attack to easily construct a free version of the victim model which bypasses the monetary costs involved in collecting data and training models.

2.1 Margin-based Adversarial Active Learning

The developed model extraction attack is constructed based on a new concept named *margin-based active learning*. The key idea is that only a few examples from the pool of unlabeled data are useful or informative for determining the separating surface/boundary of the victim classifier while all other examples are superfluous to the classifier.

We apply the margin-based uncertainty sampling methodology as the adaptive strategy for boosting examples with the least confidence in the target classifier, meaning that these selected adversarial examples are located on the global margin

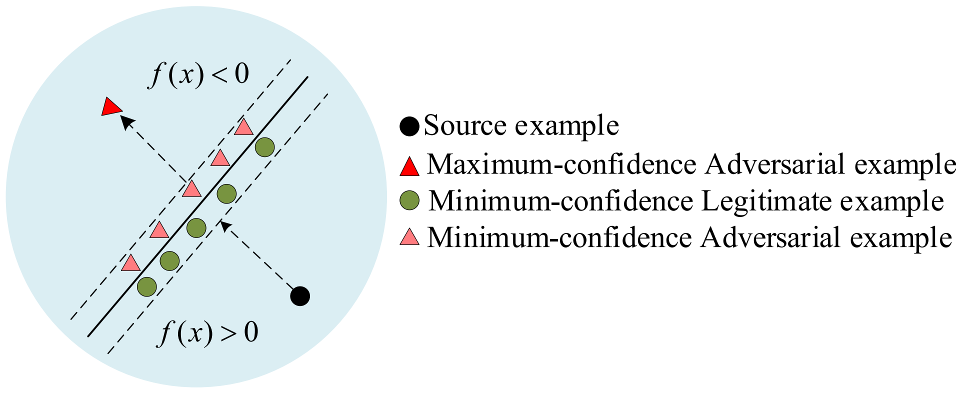
$f(x) < 0$
$f(x) > 0$
● Source example
Maximum-confidence Adversarial example
Minimum-confidence Legitimate example
Minimum-confidence Adversarial example

Figure 2: Illustration of the margin-based uncertainty sampling strategy.

of target classifier. Since a multiclass classifier can be considered as a set of binary classifiers, we first propose the margin-based active learning algorithm for a linear binary classifier and provide a geometric illustration of the uncertainty sampling theory in Figure 2.

We assume a learned affine classifier is a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ which returns the prediction results (e.g., labels and confidence) within the range $\mathcal{Y}$ when given random input images $\mathbf{x} \in \mathcal{D}(\mathbf{x})$ (e.g., extracted from test dataset with the same distribution as the training dataset). We also denote the affine hyperplane as $\mathcal{H} = \{\mathbf{x} : f(\mathbf{x}) = 0\}$. A new iterative attack procedure, named *FeatureFool* is then developed to generate the adversarial examples with different confidence (More technical details can be found in [1]). The synthetic dataset generated by an adversary consists of two types of examples, one is minimum-confidence legitimate examples, and the other is minimum-confidence adversarial examples. In comparison to those examples with high confidence, the examples in synthetic dataset are more likely to provide useful information about affine hyperplane $\mathcal{H}$ of the binary classifier. For instance, as shown in Figure 2, we can see that the green circles (minimum-confidence legitimate examples) and pink triangles (minimum-confidence adversarial examples) are near the affine hyperplane $\mathcal{H}$. They are of high uncertainty (i.e., the least confidence) and hence of maximum performance with limited black-box queries.

2.2 Synthetic Dataset Generation

We utilize the margin-based adversarial active learning algorithm to craft the informative examples and then query the victim model f_v for labels. The resulting image prediction pairs can be viewed as a synthetic dataset to train the substitute

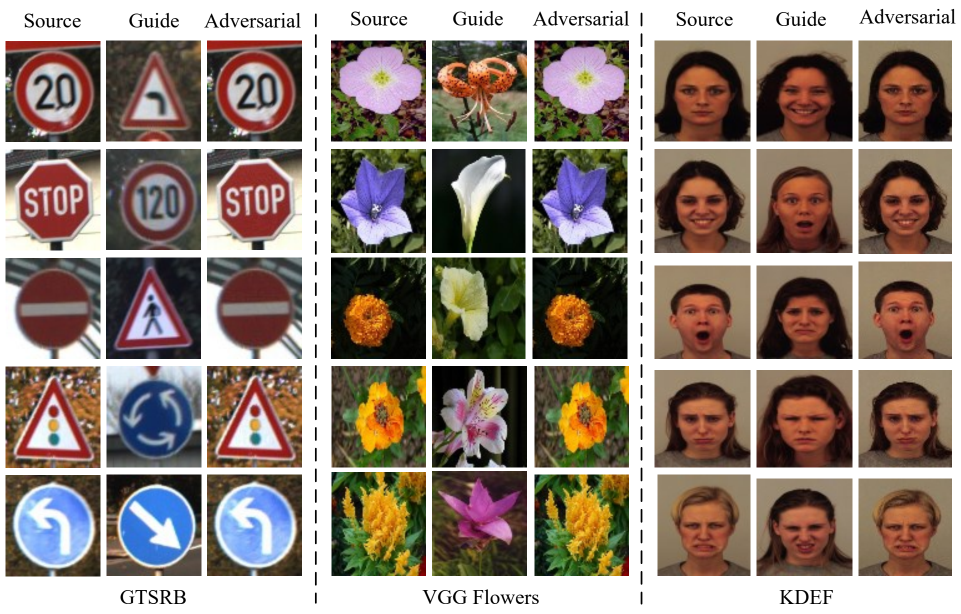

Figure 3: Adversarial Examples generated by the *FeatureFool* algorithm.

model f_s for the purpose of replicating the victim model f_v inside the commercial API. We formally define the problem of finding an informative example x'_s selected by multiclass active function $\mathcal{Q}^{LC}_{multiclass}$ as follows:

$$x^{'}_{s} = \mathcal{Q}^{LC}_{multiclass}(x^{'}) \tag{1}$$

For the x', five generation strategies are considered (Note that almost all adversarial examples generation methods can be applied and we only select five representative attacks), *Random Sample (RS)*, *Projected Gradient Descent (PGD)* [2], *Carlini and Wagner Attack (CW)* [3], *FeatureAdversary (FA)* [4] and *FeatureFool (FF)* [1]. For example, the adversarial examples generated by our *FF* are shown in Figure 3.

Our model stealing attack aims to retrain a substitute model in the target domain with near-perfect performance of the victim model. We adopt these five synthetic dataset generation strategies. For the *RS* strategy, we randomly sample a set of examples as the training dataset to re-train our substitute model. Different from the *RS* strategy, the training procedure using adversarial examples generated by the these approaches is described in Algorithm 1 of the paper [1].

3 Experimentation

3.1 Experimental Setup

In this section, we discuss the experimental results of large-scale evaluations on five popular MLaaS platforms, including those hosted by Microsoft, Face++, IBM, Google and Clarifai. Specifically, the Microsoft Cloud Vision Service, IBM Watson Visual Recognition, and Google AutoML Vision are trained for traffic sign recognition, flower recognition and face recognition, respectively. For the Face Emotion Recognition API and Offensive Content Moderation API, users access these APIs by querying it with image inputs and receive resulting confidence scores for two output labels. For all these services, the details of the victim model inside these APIs, such as training set and network architecture, are generally inaccessible to users.

The training starts from a relatively large learning rate and then the learning rate would decrease during training to allow for more fine-grained weight updates. The pre-trained weights are used to initialize our model extraction attack framework. We split the training vectors into two parts: a training dataset and a validation dataset. Then we use the stochastic gradient descent (SGD) method to minimize the cross-entropy loss while training the designed framework. We also apply some basic but powerful data augmentation techniques like flips, rotations, and scaling.

3.2 MLaaS Models Extraction Attacks

Take the Emotion Recognition Model as an example, the transfer architecture of the substitute model is the VGGFace trained on VGG-Face dataset to recognize 2622 identities (For the other commercialized MLaaS platforms hosted by Microsoft, IBM, Google and Clarifai, please refer to [1] for more testing results). The dataset utilized to train the substitute model is the KDEF dataset [5], which contains 4900 pictures of human facial expressions. The set of pictures contains 70 individuals displaying 7 different emotional expressions, including happy, fear, sad, surprise, angry, neutral and disgust. Each expression is viewed from 5 different angles. The initial training data consists of random 224×224 pixel patches cropped from these images and it is further augmented by rotating 90 degree or transforming to gray scale with 50% probability of each image. The substitute model achieves 65.33% ($90.61\times$) accuracy with 1.36k queries and 70.76% ($98.14\times$) accuracy with 2k queries by using *FF* adversarial examples. The model can achieve 71.17% ($98.74\times$) accuracy by the substitute model if trained by the *CW* adversarial examples. The substitute model trained by adversarial examples always achieves better performance than the model trained by random samples.

We also analyze the performance of substitute models while using different pre-trained models as our transfer architectures. From Figure 4, we can see that performance of our substitute model can be influenced by both the model complexity and task relevance. Therefore, in order to extract a copy of the victim model, an adversary can focus on the following aspects.

- Choosing a more complex/relevant network and the transfer architecture. In both cases, AlexNet networks achieve the lowest accuracy after extracting a victim model. A significantly more complex model VGGNet (VGG19 and VGGFace)/ResNet50 is better option while extracting a victim model. Further, as seen in Figure 4, VGGFace, which is relevant to face recognition tasks, achieves the best accuracy across all choices of substitute model architectures while targeting the face emotion recognition API. This further indicates that if the attacker does not know the exact architecture of the victim model, using a more complex and task relevant model as the transfer architecture is always a better option for the adversary.
- Sampling images relevant to the classification problem (relevant queries). This is because irrelevant queries generally lead to noisy labels and hence impose additional difficulty for re-training the substitute model.

4 Conclusion

Machine learning as a service (MLaaS) provided by cloud-based platforms, including Microsoft, Google, Face++ and Clarifai, has been widely applied in real-world applications. These services, however, tend to suffer from the model extraction attacks even with black-box access. Although previous works on model stealing attacks have shown good performance, their effectiveness is generally constrained by massive prediction queries and high costs. To address these challenges, we study the practicality of model stealing attacks against DNN models trained on commercial MLaaS platforms. Through local experiments and online attacks on commercialized MLaaS platforms we demonstrate that our model stealing attack can sufficiently train a local substitute model with almost equivalent performance to the target model. Our attack method requires significantly less queries to the target model compared to previous works of model extraction attack due to our novel design of architecture and training process of the local substitute model. In the future, we will mainly focus on designing effective defense mechanisms against model stealing attacks, and therefore enhance the robustness of DNN based MLaaS services.

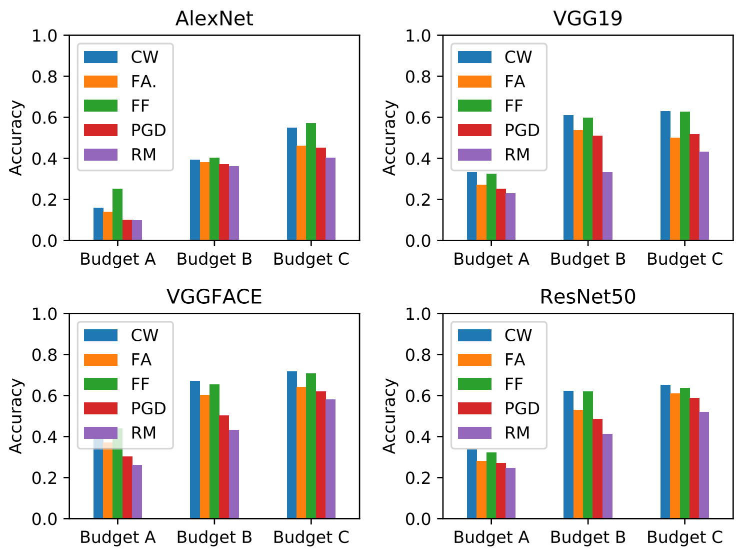

Figure 4: Architecture Choice for stealing Face++ **Emotion Recognition** API ($A = 0.68$k, $B = 1.36$k, $C = 2$k)

References

[1] H. Yu, K. Yang, T. Zhang, Y.-Y. Tsai, T.-Y. Ho, and Y. Jin, "Cloudleak: Large-scale deep learning models stealing through adversarial examples," in *Proceedings of Network and Distributed Systems Security Symposium (NDSS)*, 2020.

[2] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *ICLR*, 2018.

[3] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Security and Privacy (S&P), 2017 IEEE Symposium on*, pp. 39–57, 2017.

[4] S. Sabour, Y. Cao, F. Faghri, and D. J. Fleet, "Adversarial manipulation of deep representations," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

[5] "Kdef: A resource for studying face recognition in personal photo collections." `http://kdef.se/home/aboutKDEF.html`.