# CloudLeak:
# Large-Scale Deep Learning Models Stealing Through Adversarial Examples

Honggang Yu[1], Kaichen Yang[1], Teng Zhang[2], Yun-Yun Tsai[3], Tsung-Yi Ho[3], Yier Jin[1]
[1]University of Florida, [2]University of Central Florida, [3]National Tsing Hua University
Email: yier.jin@ece.ufl.edu

# Outline

## Background and Motivation

- AI Interface API in Cloud
- Existing Attacks and Defenses

## Adversarial Examples based Model Stealing

- Adversarial Examples
- Adversarial Active Learning
- FeatureFool
- MLaaS Model Stealing Attacks

## Case Study

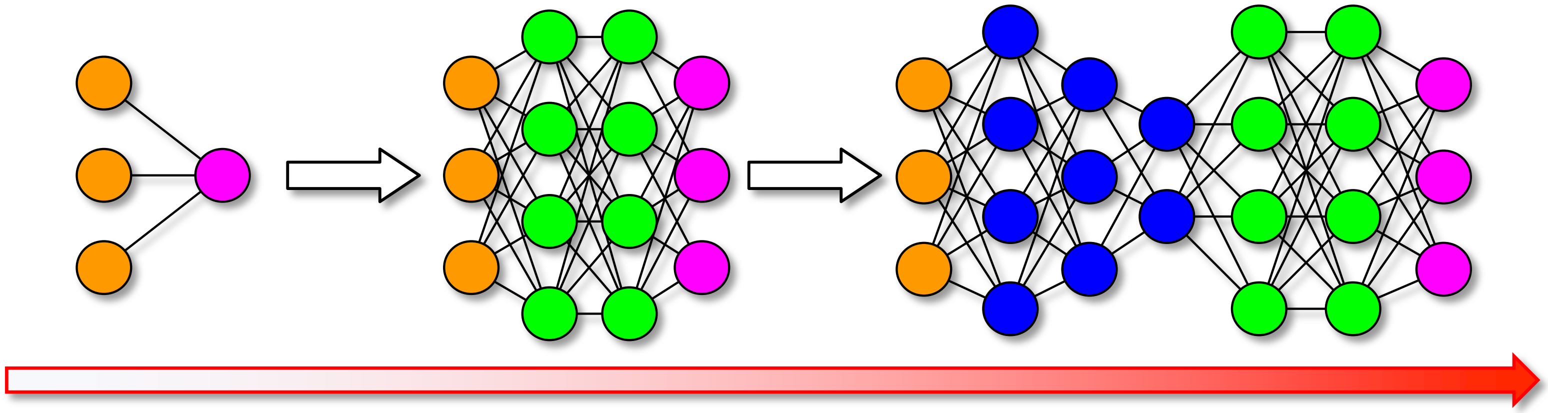
- Commercial APIs hosted by Microsoft, Face++, IBM, Google and Clarifai

## Defenses

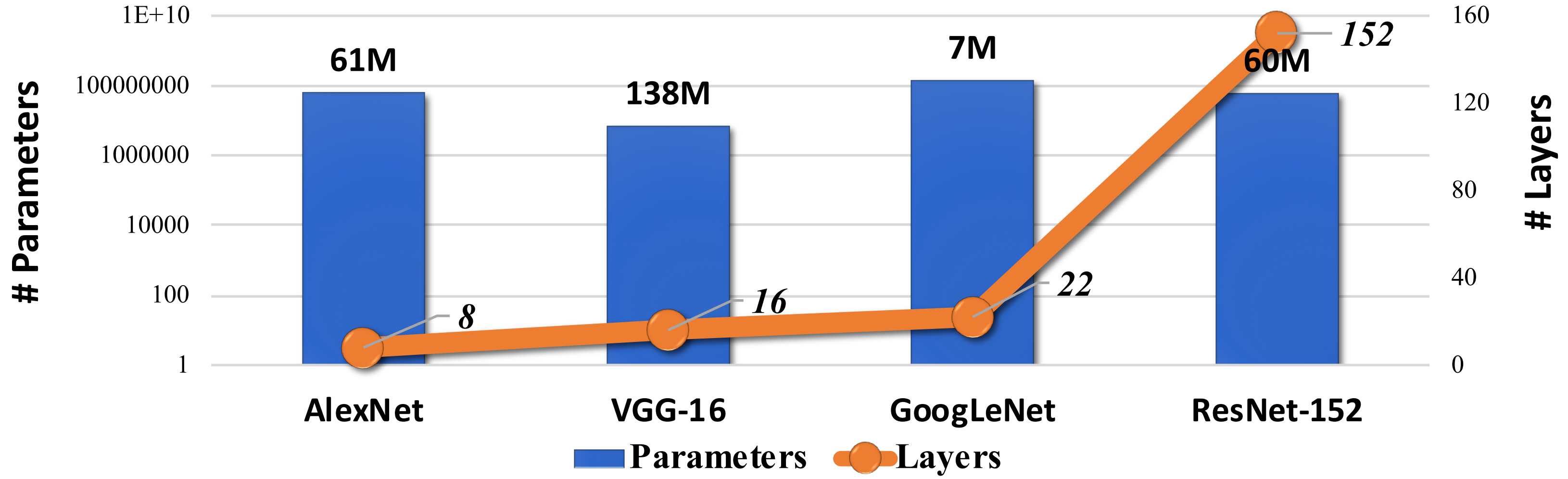
## Conclusions

# Success of DNN

"**Perceptron**" "**Multi-Layer Perceptron**" "**Deep Convolutional Neural Network**"

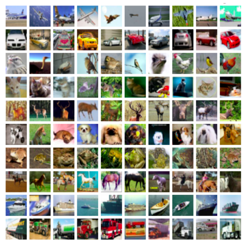


**Revolution of DNN Struture**




DNN based systems are widely used in various applications:



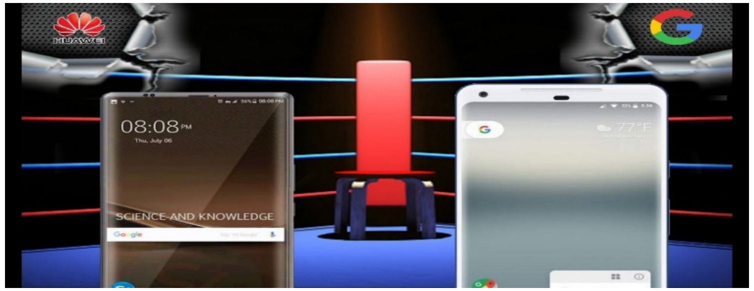
# Commercialized DNN

## Machine Learning as a Service (MLaaS)

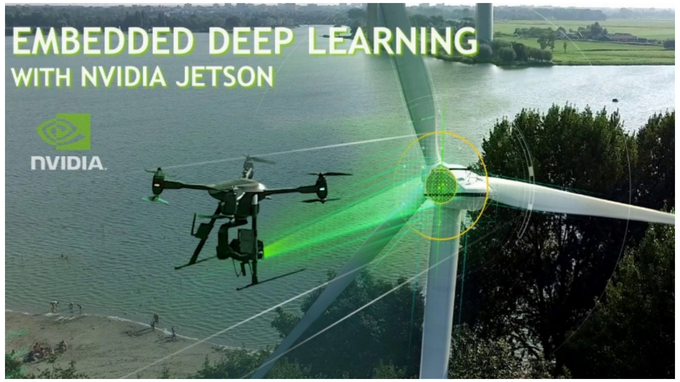
- Google Cloud Platform, IBM Watson Visual Recognition, and Microsoft Azure

## Intelligent Computing System (ICS)

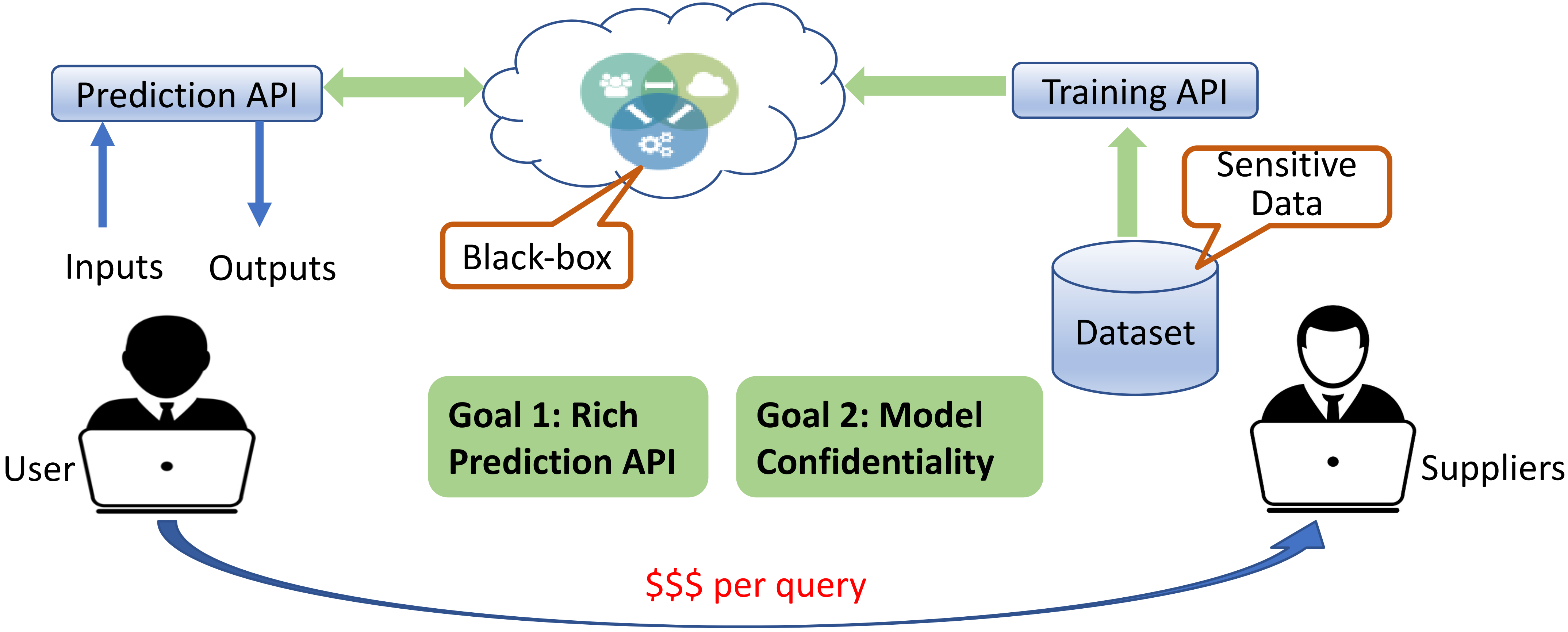
- TensorFlow Lite, Pixel Visual Core (in Pixel 2), and Nvidia Jetson TX








# Machine Learning as a Service




Overview of MLaaS Working Flow

# Machine Learning as a Service

| Services | Products and Solutions | Customization | Function | Black-box | Model Types | Monetize | Confidence Scores |
|---|---|---|---|---|---|---|---|
| Microsoft | Custom Vision | √ | Traffic Recognition | √ | NN | √ | √ |
| | Custom Vision | √ | Flower Recognition | √ | NN | √ | √ |
| Face++ | Emotion Recognition API | × | Face Emotion Verification | √ | NN | √ | √ |
| IBM | Watson Visual Recognition | √ | Face Recognition | √ | NN | √ | √ |
| Google | AutoML Vision | √ | Flower Recognition | √ | NN | √ | √ |
| Clarifai | Not Safe for Work (NSFW) | × | Offensive Content Moderation | √ | NN | √ | √ |

# Model Stealing Attacks

Various model stealing attacks have been developed

None of them can achieve a good tradeoffs among query counts, accuracy, cost, etc.

| Proposed Attacks | Parameter Size | Queries | Accuracy | Black-box? | Stealing Cost |
|---|---|---|---|---|---|
| F. Tramer (USENIX'16) | ~ 45k | ~ 102k | High | √ | Low |
| Juuti (EuroS&P'19) | ~10M | ~ 111k | High | √ | - |
| Correia-Silva (IJCNN'18) | ~ 200M | ~66k | High | √ | High |
| Papernot (AsiaCCS'17) | ~ 100M | ~7k | Low | √ | - |

# Adversarial Example based Model Stealing

# Adversarial Examples in DNN

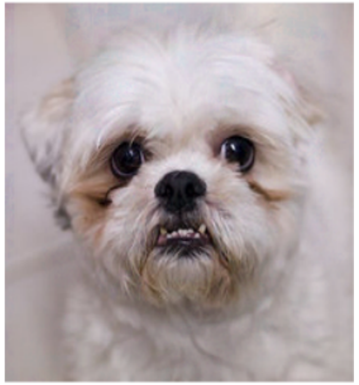
Adversarial examples are model inputs generated by an adversary to fool deep learning models.

“source example” “adversarial perturbation” “advesarial example” “target label”

+

=

$\overset{?}{=}$

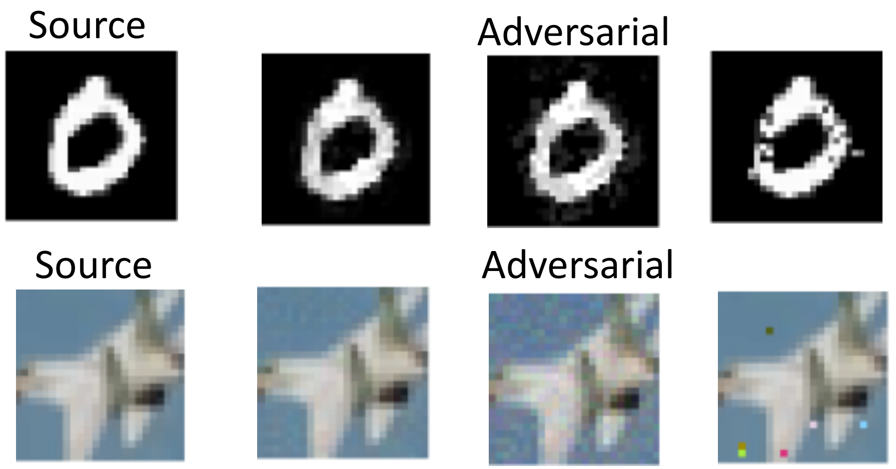


Goodfellow et al, 2014

# Adversarial Examples

## Non-Feature-based

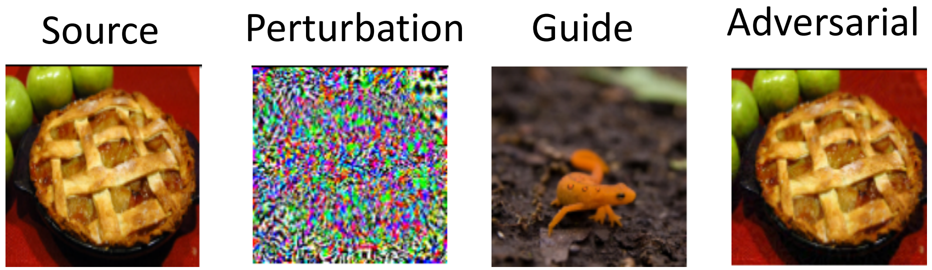
- Projected Gradient Descent (PGD) attack
- C&W Attack

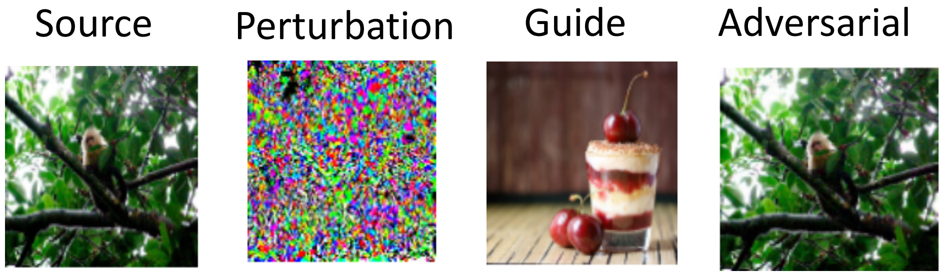



Carlini et al, 2017

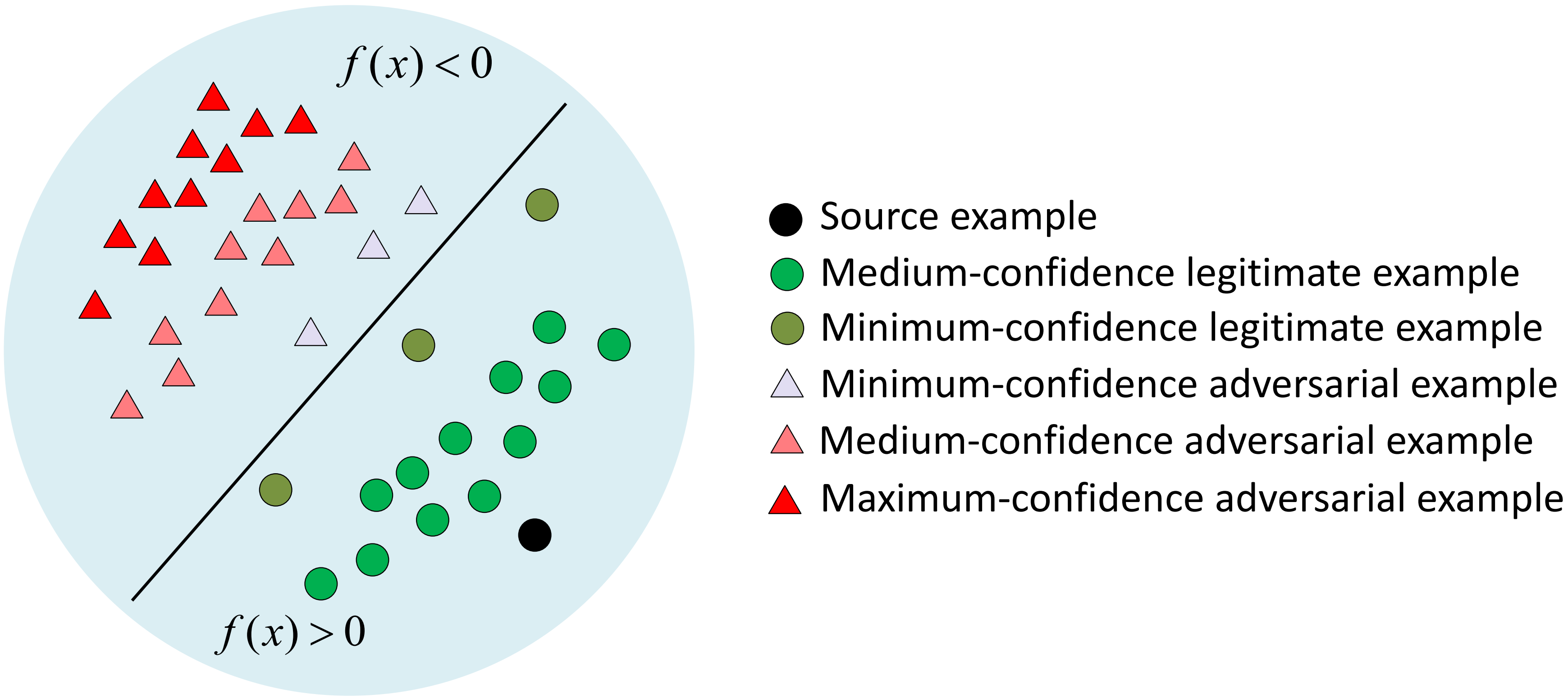
## Feature-based

- Feature adversary attack
- FeatureFool



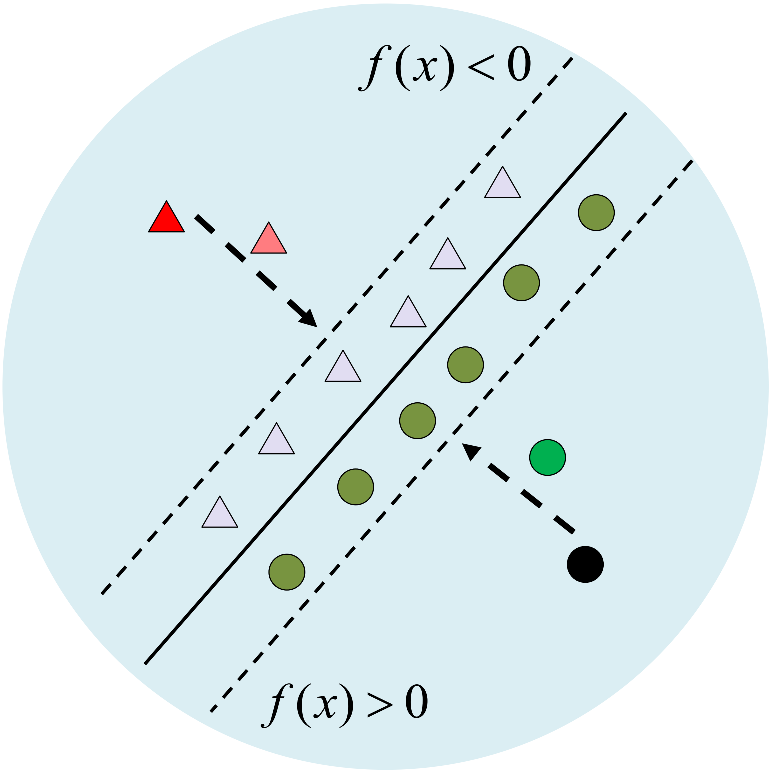
# A Simplified View of Adversarial Examples




A high-level illustration of the adversarial example generation

# Adversarial Active Learning

We gather a set of “useful examples” to train a substitute model with the performance similar to the black-box model.




- Source example
- Medium-confidence legitimate example
- Medium-confidence adversarial example
- Maximum-confidence adversarial example
- Minimum-confidence legitimate example
- Minimum-confidence adversarial example

“Useful examples”

Illustration of the margin-based uncertainty sampling strategy.

# FeatureFool: Margin-based Adversarial Examples

To reduce the scale of the perturbation, we further propose a feature-based attack to generate more robust adversarial examples.

- Attack goal: Low confidence score for true class (we use $M$ to control the confidence score).

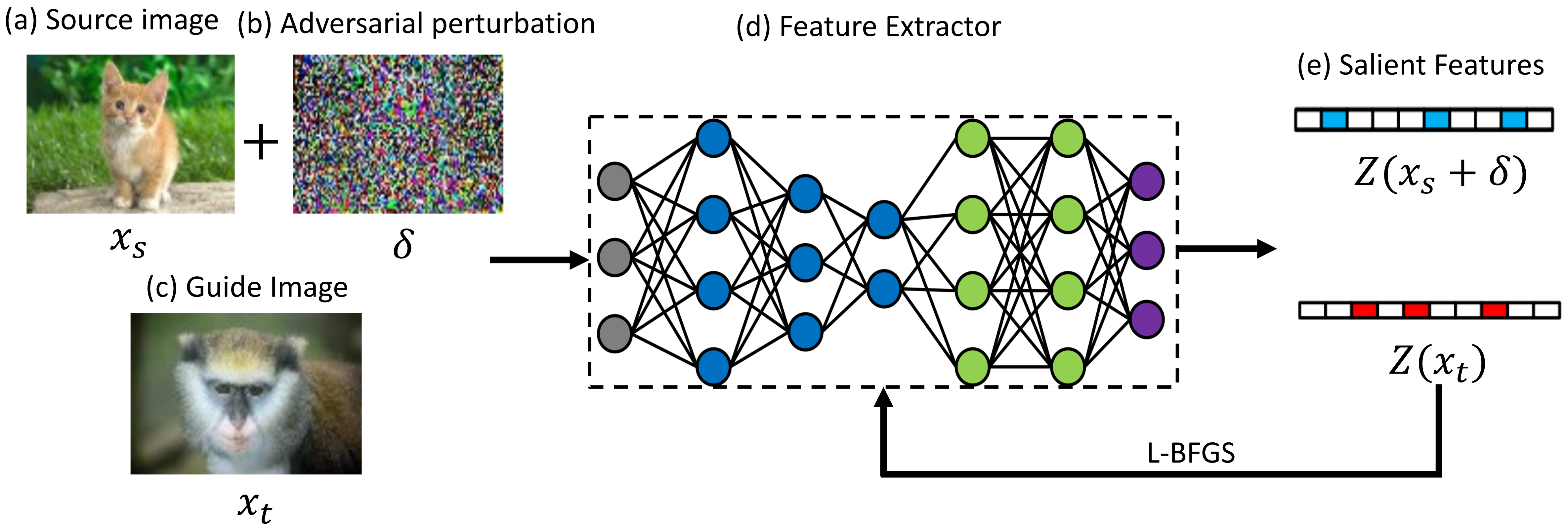
$$\text{minimize } d(x_s', x_s) + \alpha \cdot loss_{f,l}(x_s')$$
$$\text{such that } x_s' \in [0,1]^n$$

For the triplet loss $loss_{f,l}(x_s')$ , we formally define it as:

$$loss_{f,l}(x_s') = \max\Big(D\big(\emptyset_K(x_s'), \emptyset_K(x_t)\big) - D\big(\emptyset_K(x_s'), \emptyset_K(x_s)\big) + M, 0\Big)$$

- In order to solve the reformulated optimization problem above, we apply the box-constrained L-BFGS for finding a minimum of the loss function.

# FeatureFool: A New Adversarial Attack

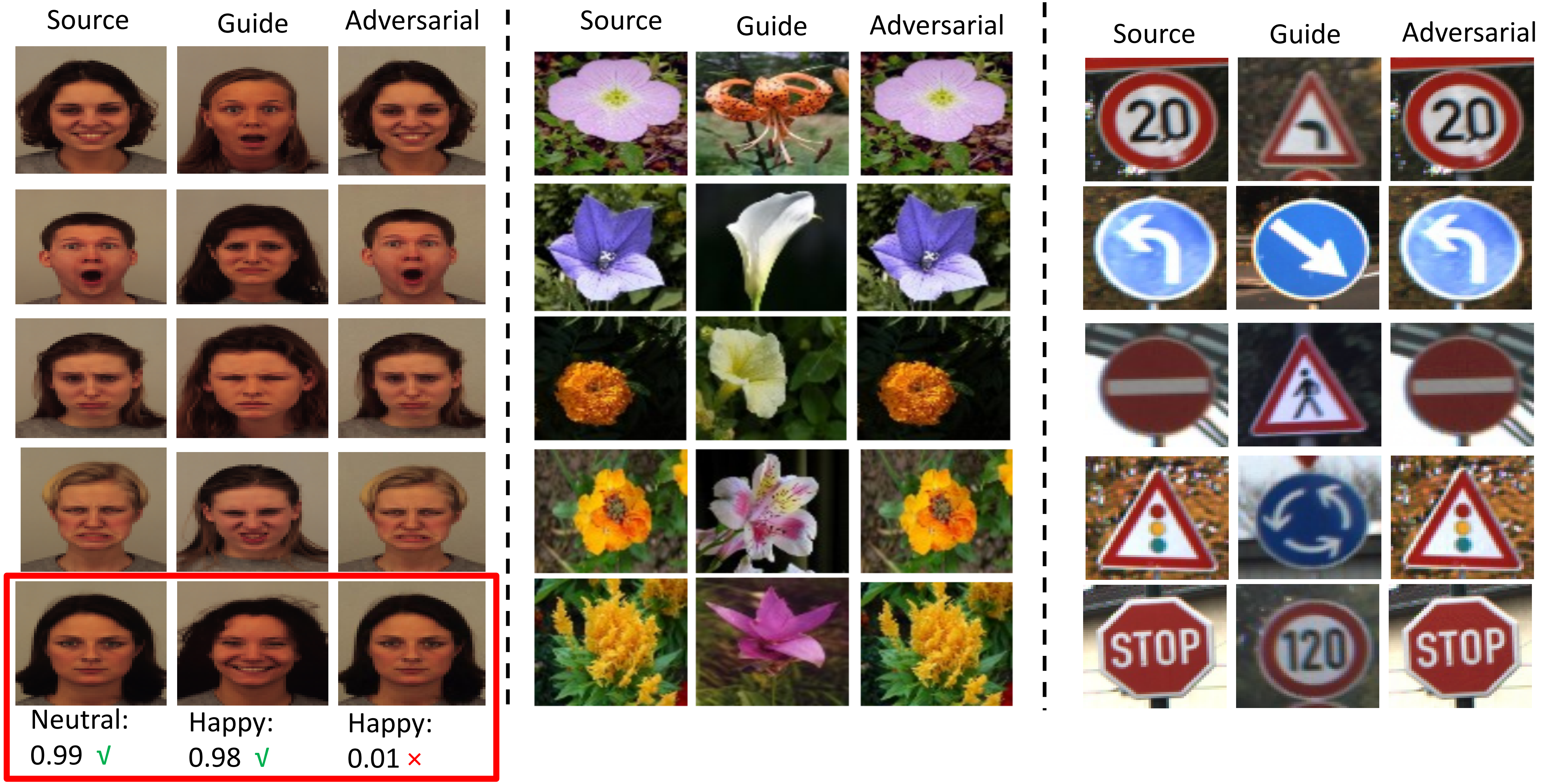



(1) Input an image and extract the corresponding n-th layer feature mapping using the feature extractor;

(2) Compute the class salience map to decide which points of feature mapping should be modified;

(3) Search for the minimum perturbation that satisfies the optimization formula.

# FeatureFool: A New Adversarial Attack



# MLaaS Model Stealing Attacks

Our attack approach:

- Use all adversarial examples to generate the malicious inputs;
- Obtain input-output pairs by querying black-box APIs with malicious inputs;
- Retrain the substitute models which are generally chosen from candidate Model Zoo.

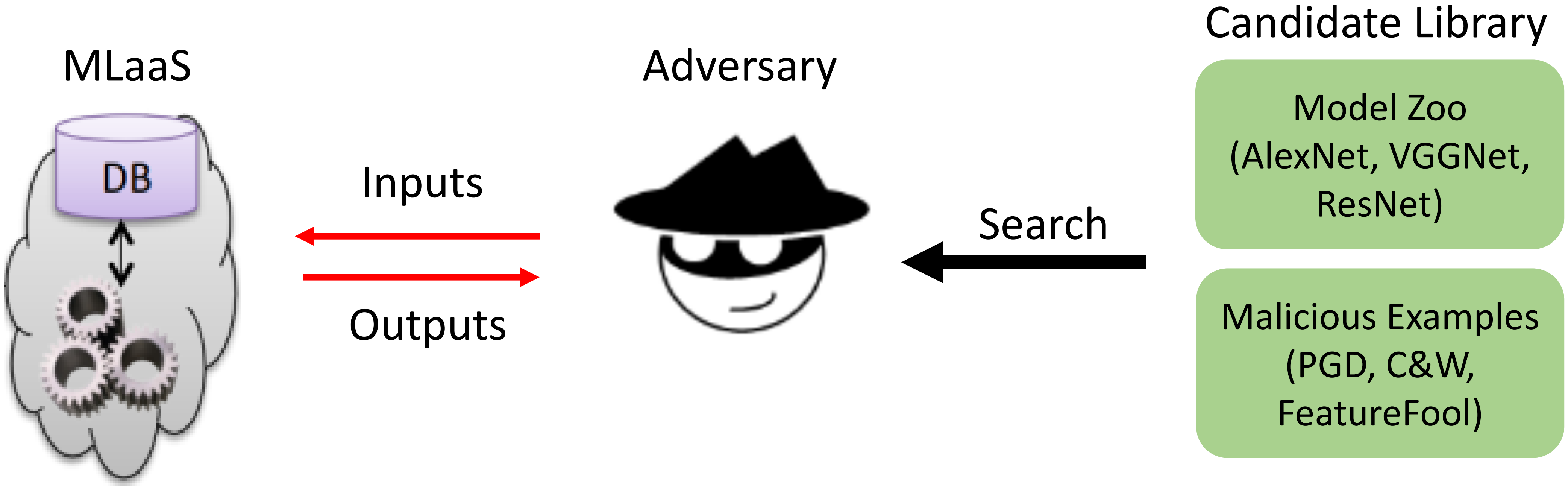
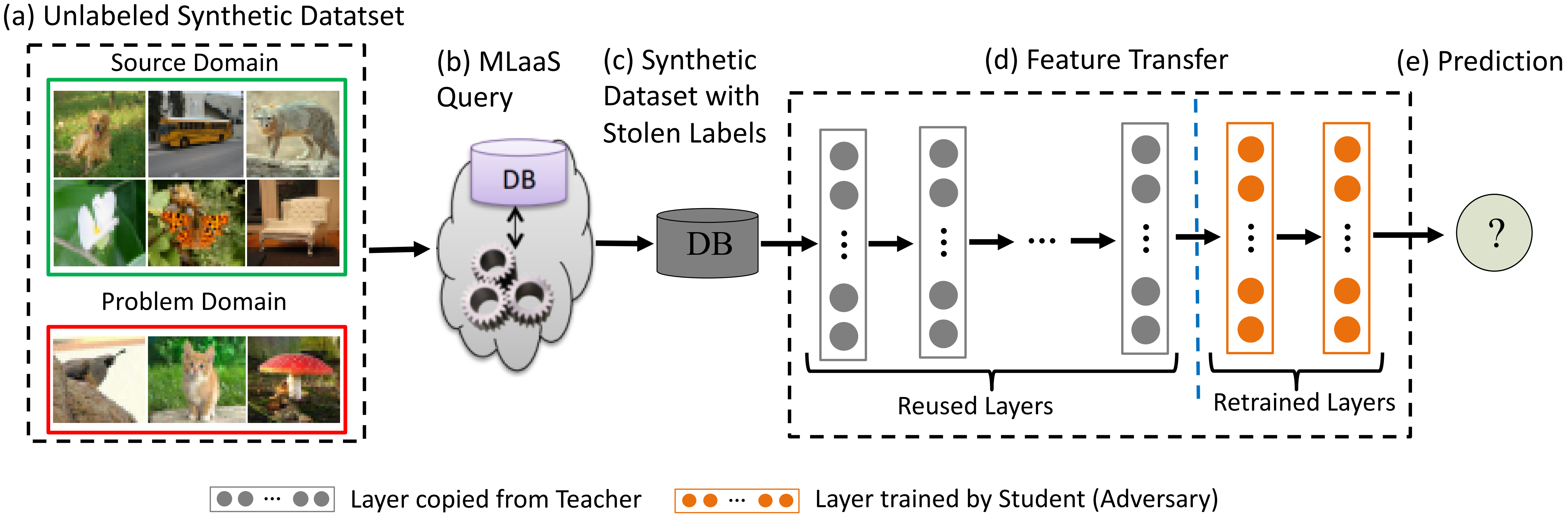



Illustration of the proposed MLaaS model stealing attacks

# MLaaS Model Stealing Attacks

## Overview of the transfer framework for the model theft attack

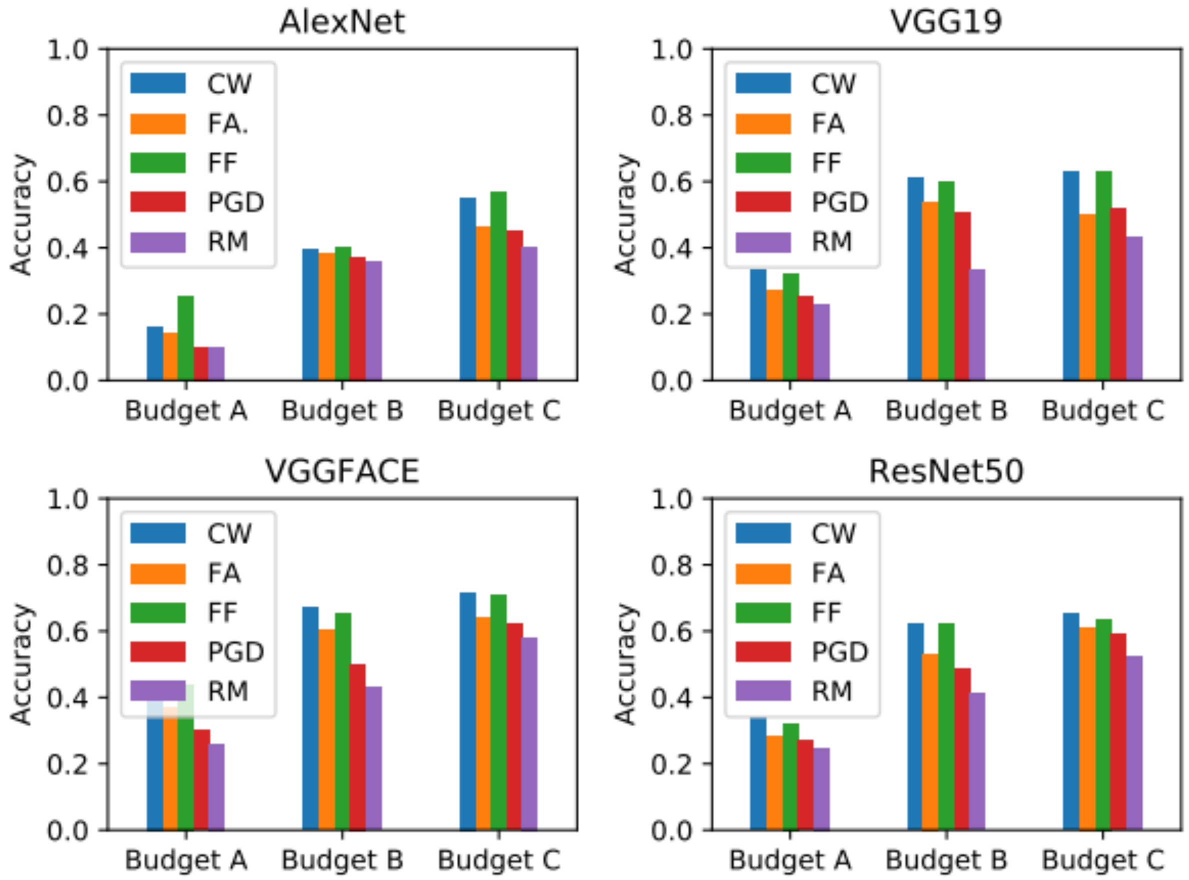



(1) Generate unlabeled dataset (2) Query MLaaS (3) Use transfer learning method to retrain the substitute model

# Example: Emotion Classification

Procedure to extract a copy of the Emotion Classification model

1) Choose a more complex/relevant network, e.g., VGGFace.
2) Generate/Collect images relevant to the classification problem in source domain and in problem domain (relevant queries).
3) MLaaS query.
4) Local model training based on the cloud query results.




Architecture Choice for stealing Face++ Emotion Classification API (A = 0.68k; B = 1.36k; C = 2.00k)

# Experimental Results

Adversarial perturbations result in a more successful transfer set.

In most cases, our FeatureFool method achieves the same level of accuracy with fewer queries than other methods

| Service | Model | Dataset | | | | | | Price ($) |
|---|---|---|---|---|---|---|---|---|
| | | Queries | RS | PGD | CW | FA | FF | |
| Microsoft | Traffic | 0.43k | 10.21% | 10.49% | 12.10% | 11.64% | 15.96% | 0.43 |
| | | 1.29k | 45.30% | 59.91% | 61.25% | 49.25% | 66.91% | 1.29 |
| | | 2.15k | 70.03% | 72.20% | 74.94% | 71.30% | 76.05% | 2.15 |
| | Flower | 0.51k | 26.27% | 27.84% | 29.41% | 28.14% | 31.86% | 1.53 |
| | | 1.53k | 64.02% | 68.14% | 69.22% | 68.63% | 72.35% | 4.59 |
| | | 2.55k | 79.22% | 83.24% | 89.20% | 84.12% | 88.14% | 7.65 |

Comparison of performance on the victim model (Microsoft) and their local substitute models.

# Comparison with Existing Attacks

Our attack framework can steal large-scale deep learning models with high accuracy, few queries and low costs simultaneously.

The same trend appears while we use different transfer architectures to steal black-box target model.

| Proposed Attacks | Parameter Size | Queries | Accuracy | Black-box? | Stealing Cost |
|---|---|---|---|---|---|
| F. Tramer (USENIX’16) | ~ 45k | ~ 102k | High | √ | Low |
| Juuti (EuroS&P’19) | ~10M | ~ 111k | High | √ | - |
| Correia-Silva (IJCNN’18) | ~ 200M | ~66k | High | √ | High |
| Papernot (AsiaCCS’17) | ~ 100M | ~7k | Low | √ | - |
| Our Method | ~ 200M | ~3k | High | √ | Low |

A Comparison to prior works.

# Evading Defenses

## Evasion of PRADA Detection

- Our attacks can easily bypass their defense by carefully selecting the parameter M from 0.1 $D$ to 0.8 $D$.
- Other types of adversarial attacks can also bypass the PRADA defense if $\delta$ is small.

| Model ($\delta$ value) | Queries made until detection | | | | | |
|---|---|---|---|---|---|---|
| | PGD | CW | FA | FF | | |
| | | | | $M = 0.8D$ | $M = 0.5D$ | $M = 0.1D$ |
| Traffic ($\delta = 0.92$) | $\boxed{\text{missed}}$ | $\boxed{\text{missed}}$ | $\boxed{\text{missed}}$ | $\boxed{\text{missed}}$ | 150 | 130 |
| Traffic ($\delta = 0.97$) | 110 | 110 | 110 | 110 | 110 | 110 |
| Flower ($\delta = 0.87$) | 110 | $\boxed{\text{missed}}$ | 220 | $\boxed{\text{missed}}$ | 290 | 140 |
| Flower ($\delta = 0.90$) | 110 | 340 | 220 | 350 | 120 | 130 |
| Flower ($\delta = 0.94$) | 110 | 340 | 220 | 350 | 120 | 130 |

# Conclusion

- We combine the theorem saliency map and feature mapping of a neural network and demonstrate the relationship between inner feature representation and final classification output
- We propose a new adversarial attack method named featurefool against local substitute models, that adopts internal representation for generating a subset of malicious samples
- We systematically study the model stealing attack and develop a novel adversarial example based model stealing attack targeting MLaaS in the cloud
- More effective defense mechanisms against the model stealing attack will be developed to enhance the robustness of DNN based MLaaS

# Thanks!

Yier Jin

Yier.jin@ece.ufl.edu