

How Secure are ML Applications?

Agenda

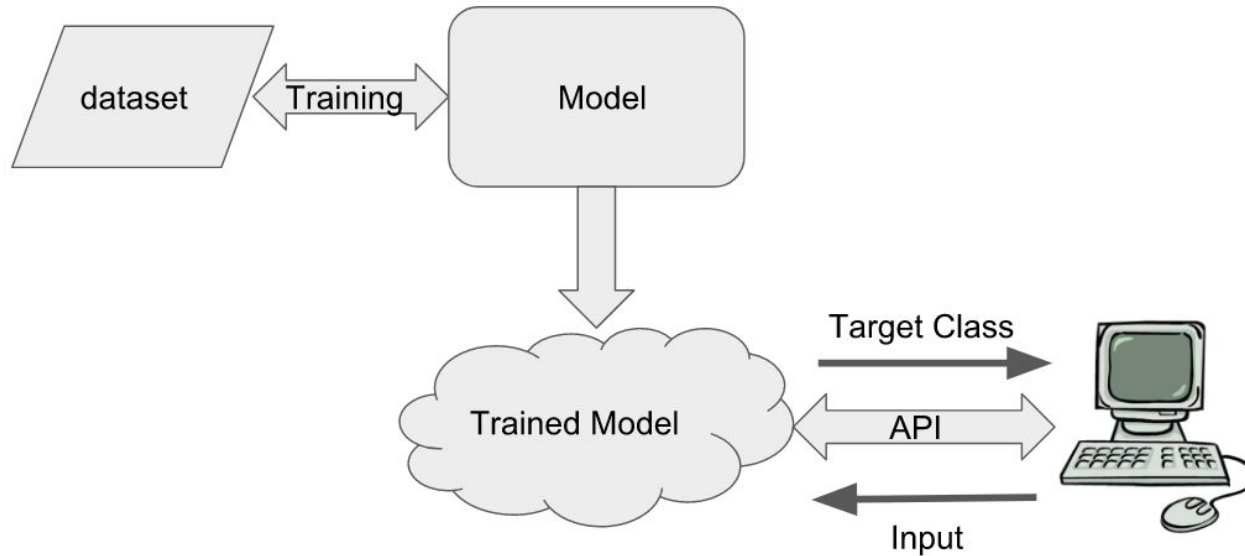
- How ML applications work
- Adversarial Learning Attack
- Model Stealing Attack
- Model Skewing Attack
- Model Inversion Attack
- Possible Mitigation (if any...)
- Strategic way to test ML applications

Applications in security

- WAF
- IDS/IPS
- Malware detection
- Antiviruses
- Spam filters
- ...



How it works in production

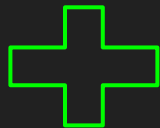


Adversarial Learning Attacks

Specially craft an input with an intention to produce desired prediction from target model.



Street Sign

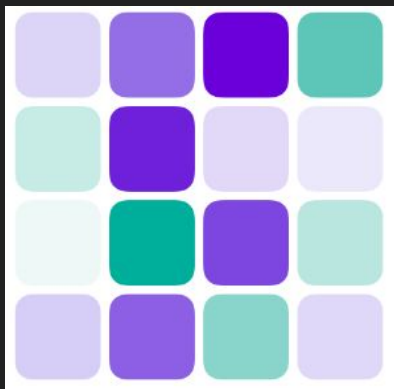


Perturbations



Mail Box

Adversarial Learning Attacks



Foolbox

Ref: <https://arxiv.org/abs/1710.08864>



Cleverhans

Ref: <https://arxiv.org/abs/1710.08864>

Adversarial Attacks Examples

One Pixel Attack for Fooling Deep Neural Networks

Jiawei Su*, Danilo Vasconcellos Vargas* and Kouichi Sakurai

AllConv



SHIP
CAR(99.7%)

NiN



HORSE
FROG(99.9%)

VGG



DEER
AIRPLANE(85.3%)

Ref: <https://arxiv.org/abs/1710.08864>

Adversarial Attacks: Face Recognition

A General Framework for Adversarial Examples with Objectives



Fig. 5. An example of digital dodging. Left: An image of actor Owen Wilson (from the PubFig dataset [40]), correctly classified by VGG143 with probability 1.00. Right: Dodging against VGG143 using AGN's output (probability assigned to the correct class < 0.01).

Adversarial Attacks: Object Detection

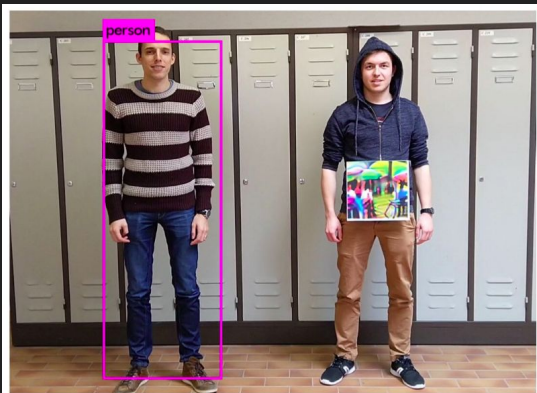
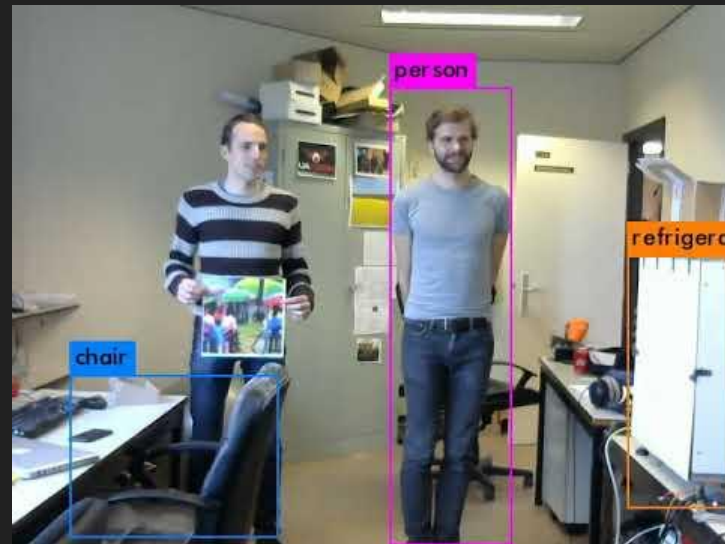
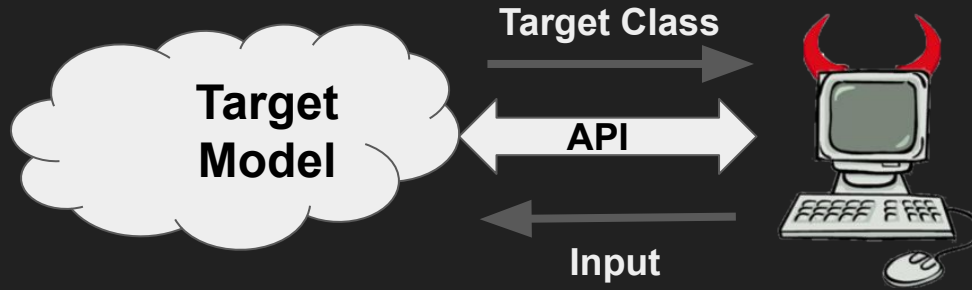


Figure 1: We create an adversarial patch that is successfully able to hide persons from a person detector. Left: The person without a patch is successfully detected. Right: The person holding the patch is ignored.

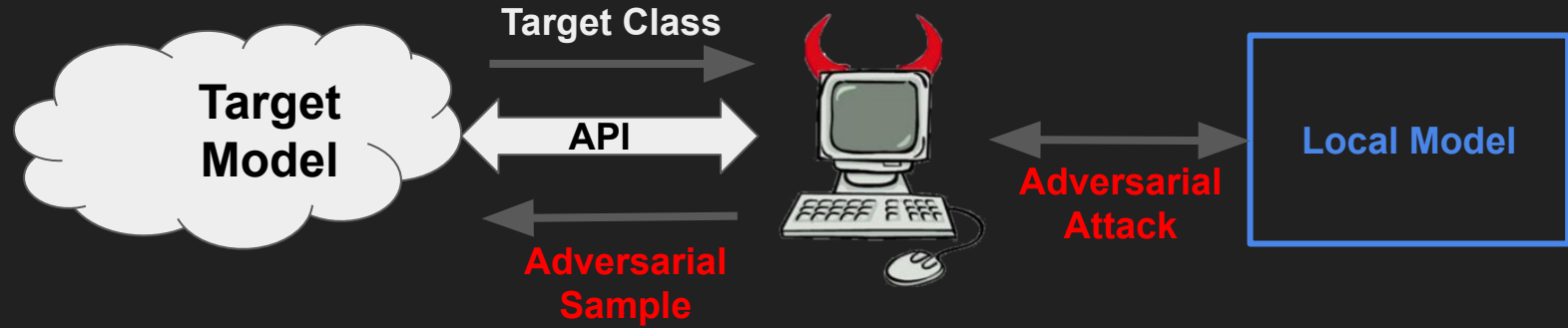


**Fooling automated surveillance cameras:
adversarial patches to attack person detection**

Adversarial Attacks on Black-Box Models



Adversarial Attacks on Black-Box Models



Adversarial Attacks are Transferable

Source Machine Learning Technique	DNN	LR	SVM	DT	kNN	Ens.
DNN	38.27	23.02	64.32	79.31	8.36	20.72
LR	6.31	91.64	91.43	87.42	11.29	44.14
SVM	2.51	36.56	100.0	80.03	5.19	15.67
DT	0.82	12.22	8.85	89.29	3.31	5.11
kNN	11.75	42.89	82.16	82.95	41.65	31.92
Target Machine Learning Technique	DNN	LR	SVM	DT	kNN	Ens.

Ref: <https://arxiv.org/pdf/1605.07277.pdf>

Adversarial Attacks Examples

Researchers Trick Cylance Antivirus Into Thinking Malware Is Trusted Software

NICOLE LINDSEY · AUGUST 2, 2019

<https://www.cpomagazine.com/cyber-security/researchers-trick-cylance-antivirus-into-thinking-malware-is-trusted-software/>

How do we deal with copying content, or the first adversarial attack in prod

Avito company blog , Programming , algorithms , Image processing , Machine learning

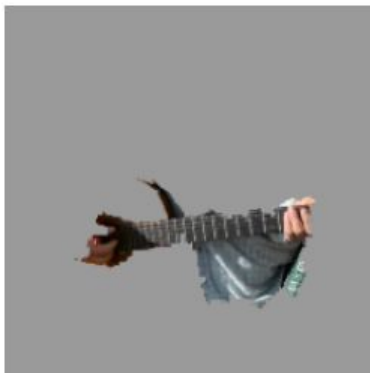
<https://habr.com/ru/company/avito/blog/452142/>

LIME for Model Interpretability

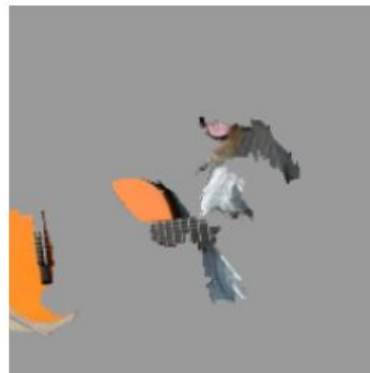
**“Why Should I Trust You?”
Explaining the Predictions of Any Classifier**



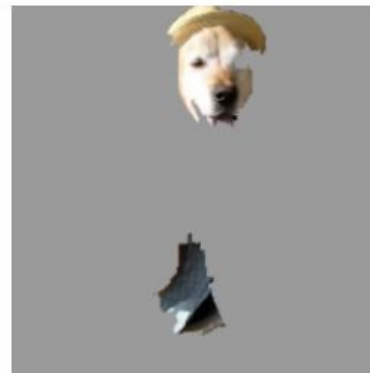
(a) Original Image



(b) Explaining *Electric guitar*



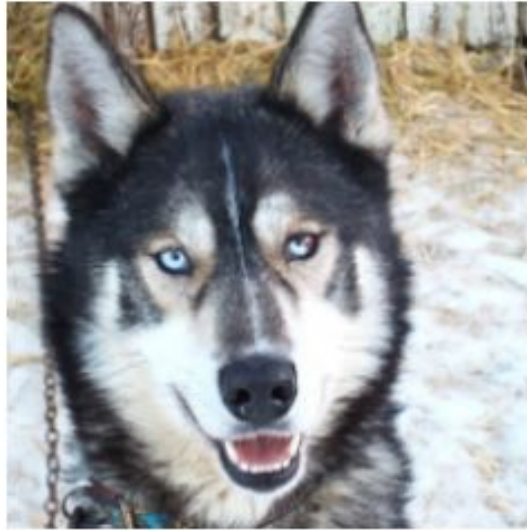
(c) Explaining *Acoustic guitar*



(d) Explaining *Labrador*

Figure 4: Explaining an image classification prediction made by Google’s Inception neural network. The top 3 classes predicted are “Electric Guitar” ($p = 0.32$), “Acoustic guitar” ($p = 0.24$) and “Labrador” ($p = 0.21$)

Explaining Predictions



(a) Husky classified as wolf



(b) Explanation

Ref: <https://arxiv.org/pdf/1602.04938.pdf>

LIME to bypass WAF

query: /index/lab/home.html

Text with highlighted words

/index/lab/home.html

Good query

home	0.05
index	0.05
lab	0.05
html	0.01

Bad query

query: /<script>alert(1)</script>/

Text with highlighted words

/script/alert(1)/script/

Good query

script	0.00
1	0.00
alert	0.00

Bad query

Model Stealing Attacks

- Offline Attacks
 - Stealing Locally deployed models
- Online Attacks
 - Stealing models deployed on cloud with black box access

Offline Model Stealing

- Analyse the serialized model
- Understand used Frameworks
- Leverage these frameworks to get predictions from model

Offline Model Stealing

```
00000000: 0400 0000 0100 0000 0300 0000 5620 310d .....V 1
00000010: 0000 006e 6e2e 5365 7175 656e 7469 616c ...nn.Sequential
00000020: 0300 0000 0200 0000 0400 0000 0200 0000 .....
00000030: 0500 0000 7472 6169 6e05 0000 0000 0000 ....train.....
00000040: 0002 0000 0007 0000 006d 6f64 756c 6573 .....modules
00000050: 0300 0000 0300 0000 0d00 0000 0100 0000 .....
00000060: 0000 0000 0000 f03f 0400 0000 0400 0000 .....?.....
00000070: 0300 0000 5620 310e 0000 006e 6e2e 436f ....V 1....nn.Co
00000080: 6e63 6174 5461 626c 6503 0000 0005 0000 ncatTable.....
00000090: 0004 0000 0002 0000 0005 0000 005f 7479 ....._tv
000000a0: 7065 0200 0000 1100 0000 746f 7263 682e pe.....torch.
000000b0: 466c 6f61 7454 656e 736f 7202 0000 0007 FloatTensor.....
000000c0: 0000 006d 6f64 756c 6573 0300 0000 0600 ...modules.....
000000d0: 0000 0200 0000 0100 0000 0000 0000 0000 .....
000000e0: f03f 0400 0000 0700 0000 0300 0000 5620 .?.....V
000000f0: 3115 0000 006e 6e2e 5370 6174 6961 6c43 1....nn.SpatialC
00000100: 6f6e 766f 6c75 7469 6f6e 0300 0000 0800 onvolution.....
00000110: 0000 0d00 0000 0200 0000 0400 0000 7061 .....pa
00000120: 6457 0100 0000 0000 0000 0000 f03f 0200 dW.....?..
00000130: 0000 0200 0000 6457 0100 0000 0000 0000 .....dW.....
00000140: 0000 0040 0200 0000 0b00 0000 6e49 6e70 ...@.....nInp
00000150: 7574 506c 616e 6501 0000 0000 0000 0000 utPlane.....
00000160: 0008 4002 0000 0006 0000 006f 7574 7075 ..@.....outpu
00000170: 7404 0000 0009 0000 0003 0000 0056 2031 t.....V 1
00000180: 1100 0000 746f 7263 682e 466c 6f61 7454 ....torch.FloatT
00000190: 656e 736f 7200 0000 0001 0000 0000 0000 ensor.....
000001a0: 0000 0000 0002 0000 0002 0000 006b 4801 .....kH.
000001b0: 0000 0000 0000 0000 0008 4002 0000 000c .....@.....
000001c0: 0000 006e 4f75 7470 7574 506c 616e 6501 ...nOutputPlane.
```

Offline Model Stealing

```
public Model loadModel(String modelFolder) {  
    List<String> categories = loadCategories(modelFolder + "/categories.txt");  
    if (categories == null) {  
        Log.e(TAG, "Failed to load categories: " + modelFolder + "/categories.txt");  
        return null;  
    }  
    ByteBuffer enginePtr = loadModelFromAssets(modelFolder + "/model.net", modelFolder + "/stat.t7");  
    if (enginePtr != null) {  
        return new Model(enginePtr, categories, 224);  
    }  
    Log.e(TAG, "Failed to load model");  
    return null;  
}
```

```
# Loading model  
from torch.utils.serialization import load_lua  
model = load_lua(model_path)  
stat = load_lua(model_path[:-9]+'stat.t7')  
model_op = predict(IMAGE_PATH)
```

```

without bias
|
| (4): nn.SpatialBatchNormaliz
| (5): nn.SpatialDropout
| }
|`-> (1): nn.Identity
|. -> output
}
(1): nn.CAddTable
(2): nn.ReLU
}
}
(8): nn.Identity
(9): nn.SpatialAveragePooling(14x14, 1, 1)
(10): nn.View(128)
(11): nn.Linear(128 -> 696)
(12): nn.SoftMax
}

```

```

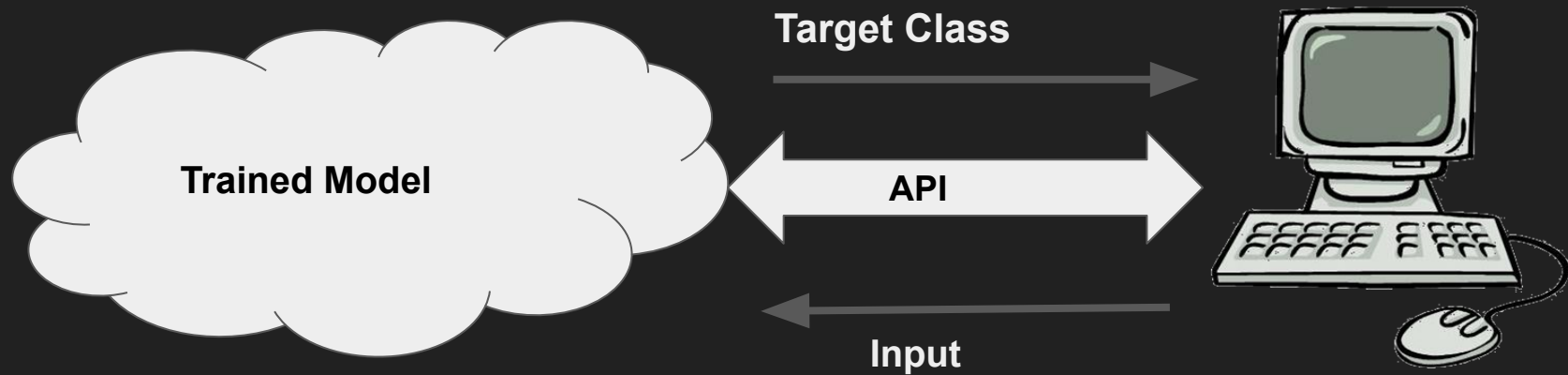
|`-> (1): nn.Identity
|. -> output
}
(1): nn.CAddTable
(2): nn.ReLU
}
}
(8): nn.Identity
(9): nn.SpatialAveragePooling(14x14, 1, 1)
(10): nn.View(1, 128)
(11): nn.Linear(128 -> 696)
(12): nn.SoftMax
}

```

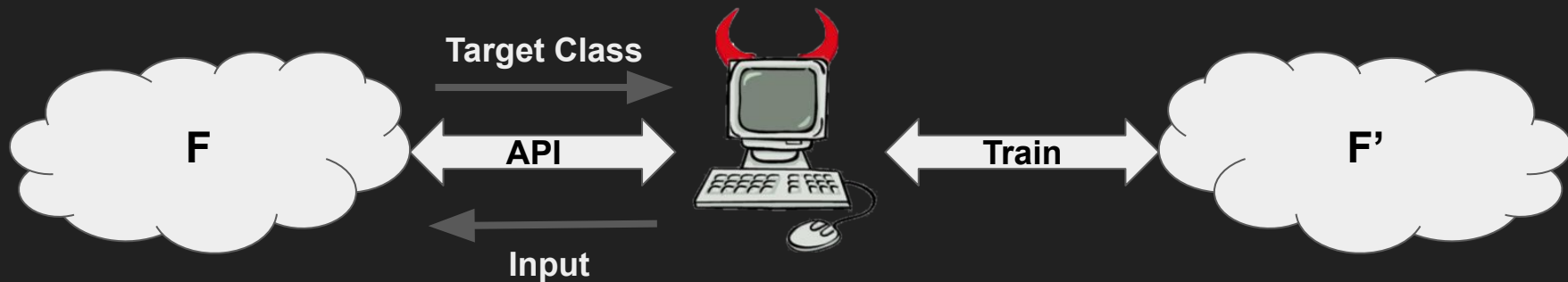
`torch.nn.View(1, 128)`

Black Box model stealing

MLaaS



Online Attacks



Input: $[f_1, f_2, f_3, \dots, f_n]$

F output: $P(\text{class1}), P(\text{class2}), P(\text{class3}), \dots, P(\text{classN})$

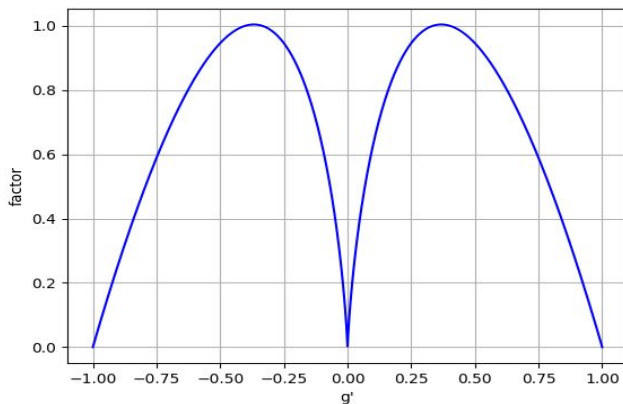
API output: $\max(P(\text{class1}), P(\text{class2}), P(\text{class3}), \dots, P(\text{classN}))$

GDALR: Gradient Driven Adaptive Learning Rate

$$g'_i = \tanh(g_i) \quad (7)$$

$$fact_i = \text{abs}(g'_i 2\pi \log_{10}(\text{abs}(g'_i))) \quad (8)$$

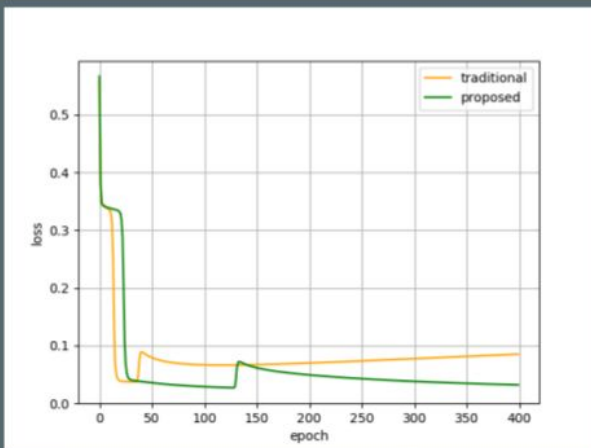
$$l'_i = l_i \cdot fact_i \quad (9)$$



Ref: <https://ieeexplore.ieee.org/document/8878726>

Results: Logistic Regression

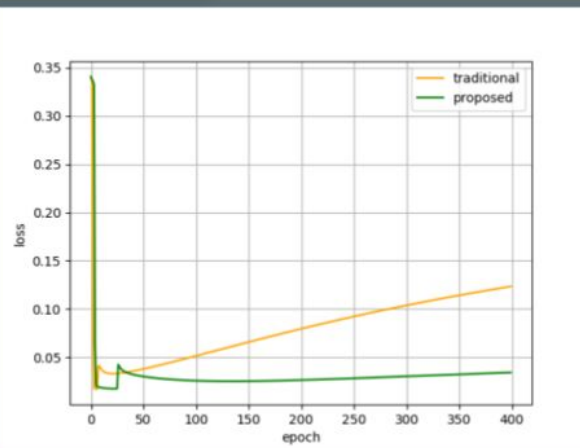
$$l = 0.01$$



$$T_{Loss} = 0.0849$$

$$P_{Loss} = 0.0317$$

$$l = 0.05$$

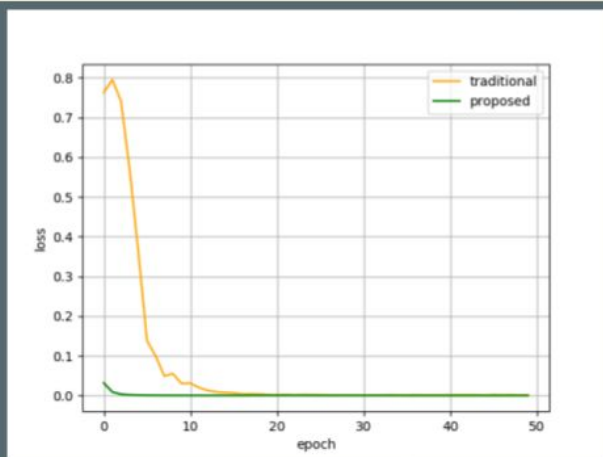


$$T_{Loss} = 0.1233$$

$$P_{Loss} = 0.0342$$

Results: Multi Layer Perceptron

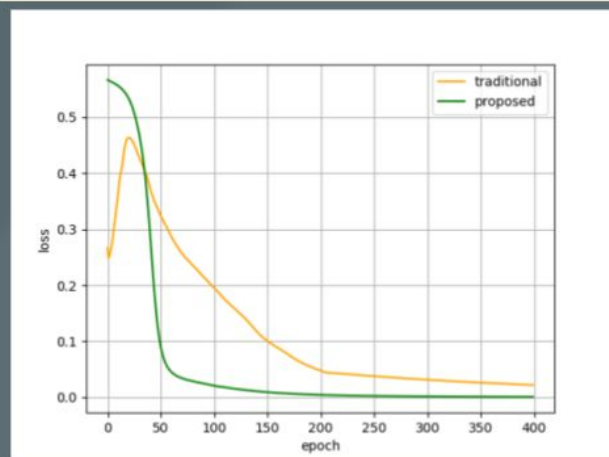
$$l = 10^{-3}$$



$$T_{Loss} = 0.0014$$

$$P_{Loss} = 5.444 \times 10^{-5}$$

$$l = 10^{-5}$$

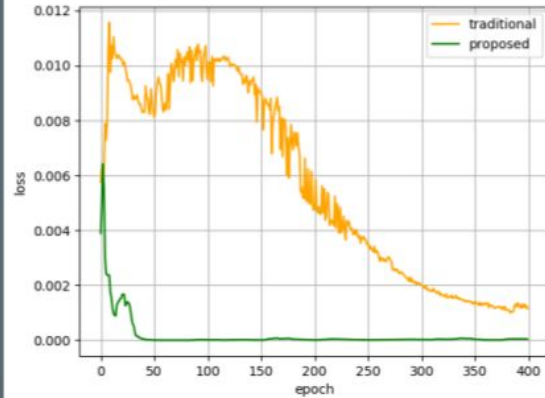


$$T_{Loss} = 0.0219$$

$$P_{Loss} = 0.0007$$

Results: Convolutional Neural Network

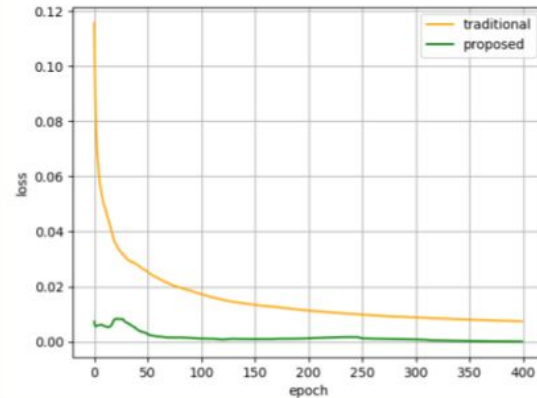
$$l = 10^{-4}$$



$$T_{Loss} = 0.0011$$

$$P_{Loss} = 3.993 \times 10^{-5}$$

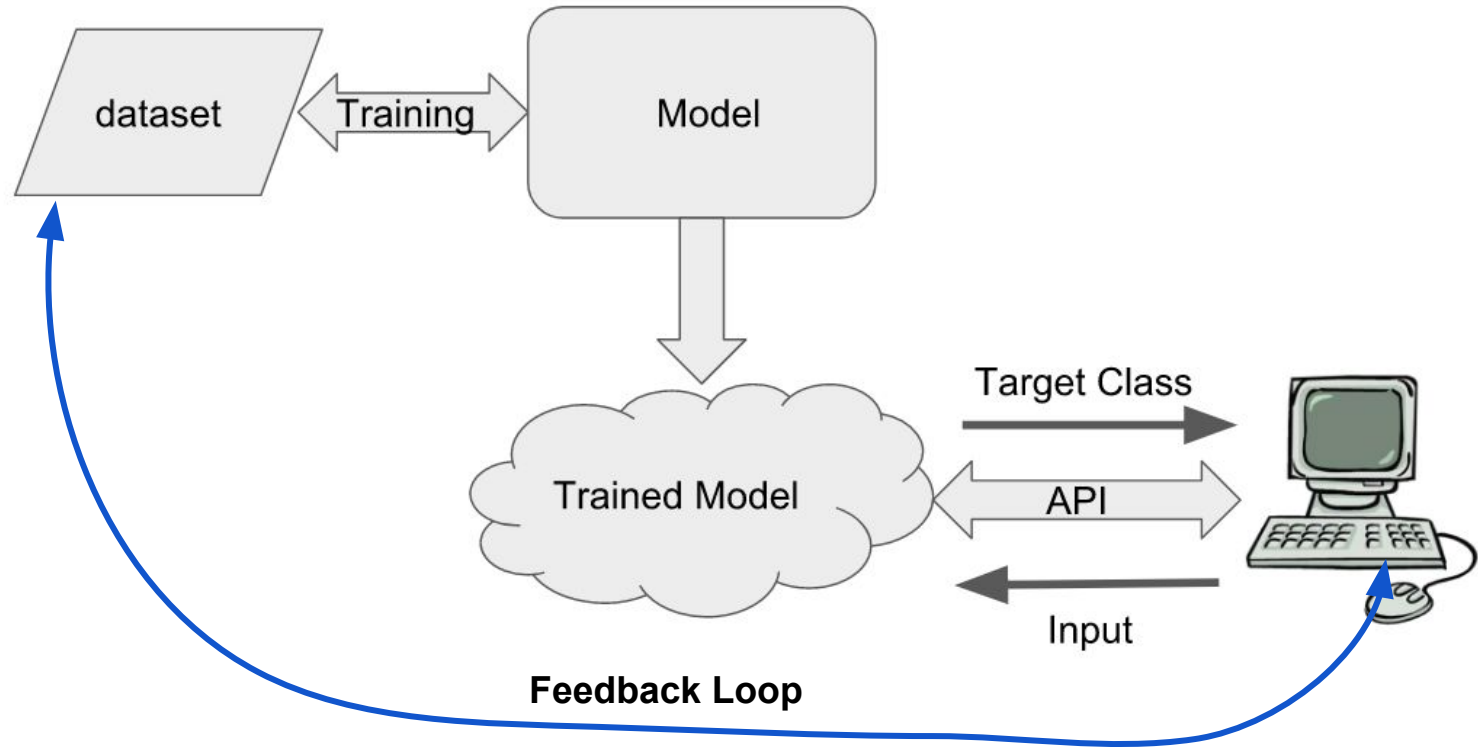
$$l = 10^{-5}$$



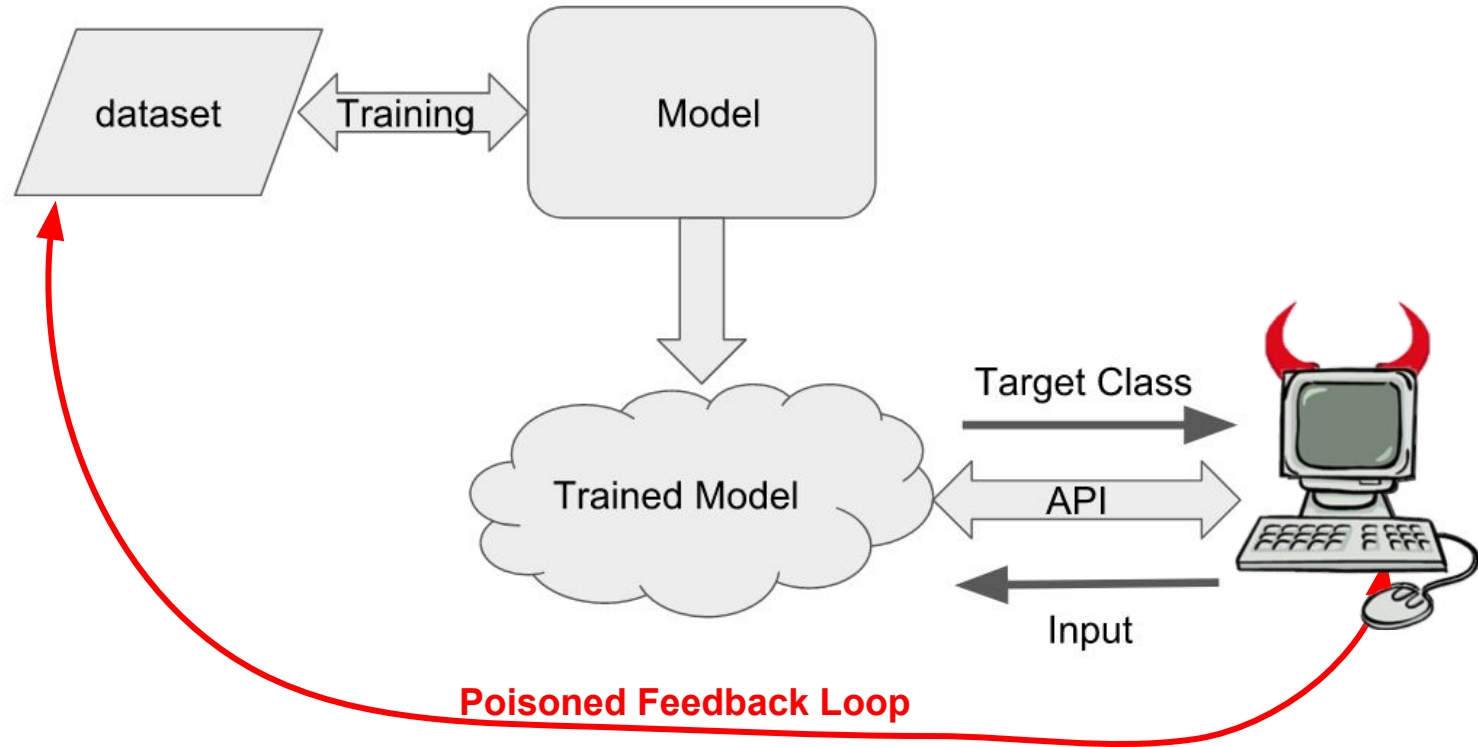
$$T_{Loss} = 0.0073$$

$$P_{Loss} = 4.184 \times 10^{-5}$$

Model skewing Attacks



Model skewing Attacks

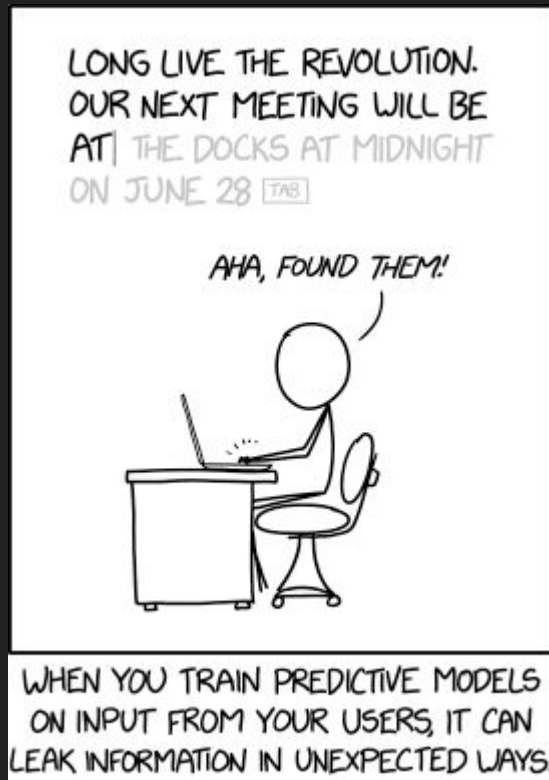


Model Inversion Attack



Figure 1: An image recovered using a new model inversion attack (left) and a training set image of the victim (right). The attacker is given only the person's name and access to a facial recognition system that returns a class confidence score.

Unintended Memorization in Neural Networks



Ref: <https://www.usenix.org/system/files/sec19-carlini.pdf>

Final Notes

- ML application learn patterns from “given” dataset
- They are different from signature / rule based applications
- Identify the use-case
- White box or Black box
- Craft Threat Model
- Start with simple Methods

Thank You!

nikhilj@payatu.com

@adversarial_nik