Aalto University

UNIVERSITY OF WATERLOO

Extraction of Complex DNN Models: Real Threat or Boogeyman?

Buse Atli, Sebastian Szyller, Mika Juuti, Samuel Marchal, N. Asokan

Outline

Is model confidentiality important?

Can models be extracted via their prediction APIs?

What can be done to counter model extraction?

Is model confidentiality important?

Machine learning models: business advantage and intellectual property (IP)

Cost of

- gathering relevant data
- labeling data
- expertise required to choose the right model training method
- resources expended in training

Adversary who steals the model can avoid these costs

How to prevent model theft?

White box model theft can be countered by

- **Computation with encrypted models**
- **Protecting models using secure hardware**
- **Hosting models behind a firewalled cloud service**

Basic idea: hide the model itself, expose model functionality only via a prediction API

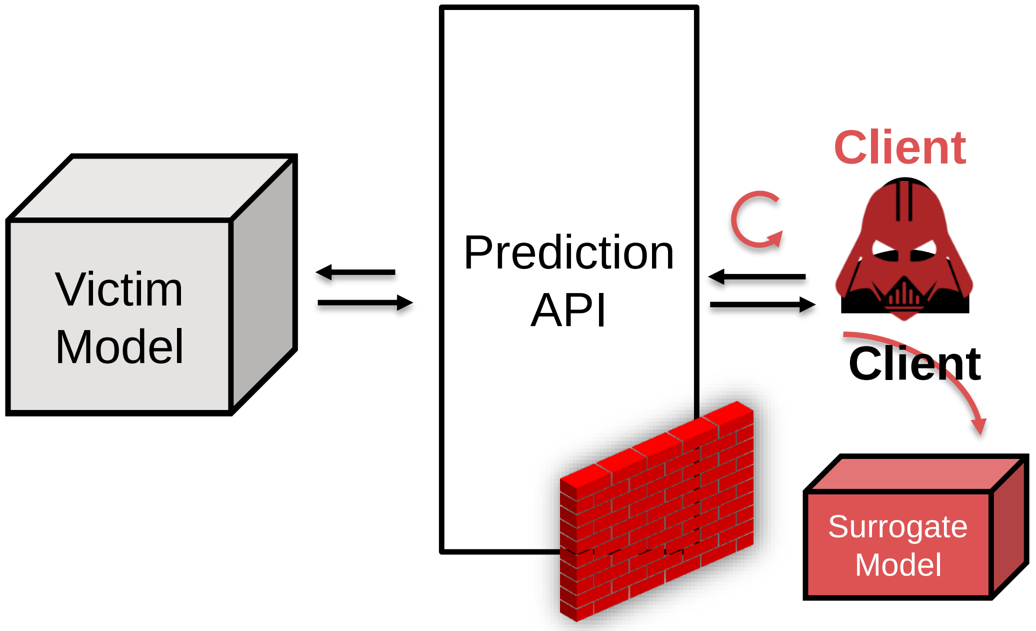
Is that enough to prevent model theft?

Extracting models via their prediction APIs

Prediction APIs are oracles that leak information

Adversary

- Malicious client
- Goal: rebuild a surrogate model for a victim model
- Capability: access to prediction API or model outputs



Prior work on extracting

- Logistic regression, decision trees[1]
- Simple CNN models[2]
- Querying API with synthetic samples

[1] Tramer et al. -*Stealing Machine Learning Models via Prediction APIs*. USENIX'16 (https://arxiv.org/abs/1609.02943)
[2] Papernot et al. -*Practical Black-Box Attacks against Machine Learning*. ASIA CCS '17(https://arxiv.org/abs/1602.02697)

Model extraction: attacks and defenses

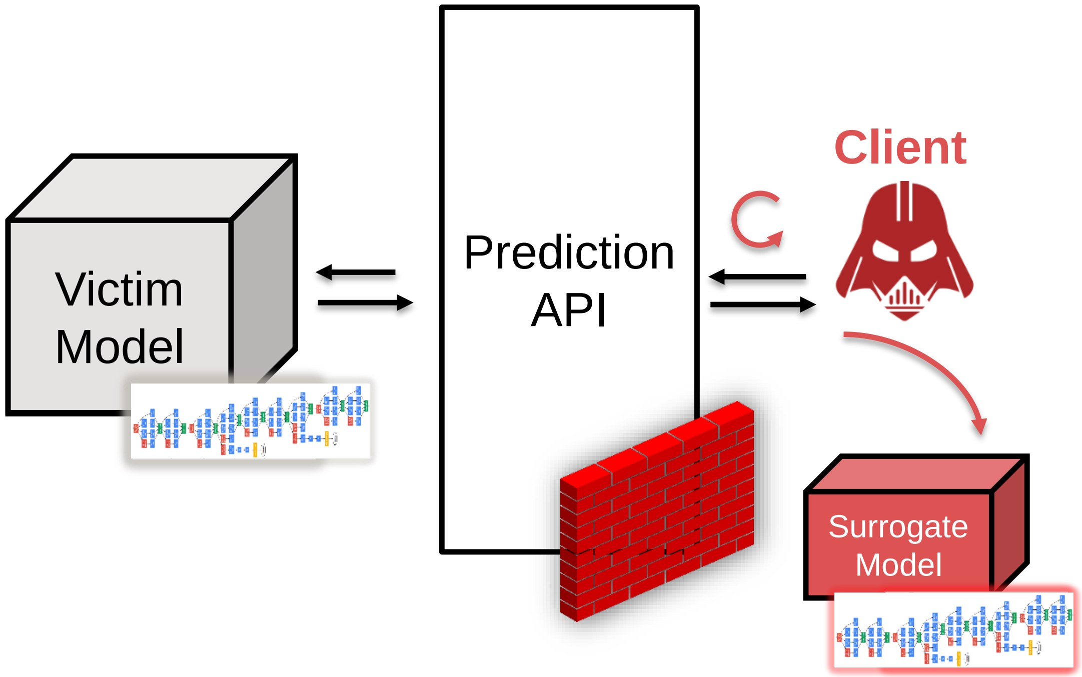
Are model extraction attacks realistic? Can they be detected effectively?

Against simple DNN models[1]

- E.g., MNIST, GTSRB
- Strategy for generating synthetic samples
- Hyperparameters CV-search
- **Defense:** detect abnormal query distribution

Against complex image classification models?

- Can adversaries extract complex DNNs successfully?
- Are common adversary models realistic?
- Are current defenses effective?



[1] Juuti et al. - *PRADA: Protecting against DNN Model Stealing Attacks*. EuroS&P'19 (https://arxiv.org/abs/1805.02628)

Extraction of Complex DNN Models: Knockoff nets[1]

Goal:

- Build a surrogate model that
  - steals model functionality of victim model
  - performs similarly on the same task with high classification accuracy

Adversary capabilities:

- Victim model knowledge:
  - None of train/test data, model internals, output semantics
  - Access to full prediction probability vector
- Access to natural samples, not (necessarily) from the same distribution as train/test data
- Access to pre-trained high-capacity model

[1] Orekondy et al. - *Knockoff Nets: Stealing Functionality of Black-Box Models.* CVPR’19 (https://arxiv.org/abs/1812.02766)

Knockoff nets: Our Goals and Contributions

Reproduce empirical evaluation of Knockoff nets [1] to confirm its effectiveness

Introduce a defense within the adversary model in [1] to detect attacker's queries

Revisit adversary model in [1]

- Explore impact of a more realistic adversary model on attack and defense effectiveness
  - Attack effectiveness decreases: Different surrogate-victim architectures, reduced granularity of victim's prediction API's output, reduced diversity of adversarial queries
  - Defense effectiveness decreases: Attacker has natural samples distributed like victim's training data

[1] Orekondy et al. - *Knockoff Nets: Stealing Functionality of Black-Box Models.* CVPR'19 (https://arxiv.org/abs/1812.02766)

Knockoff nets [1] : Experimental Setup

Victim model derived from public, pre-trained, high-capacity model (e.g., ResNet-34 on ImageNet)

Strategy

Collect unlabeled natural data

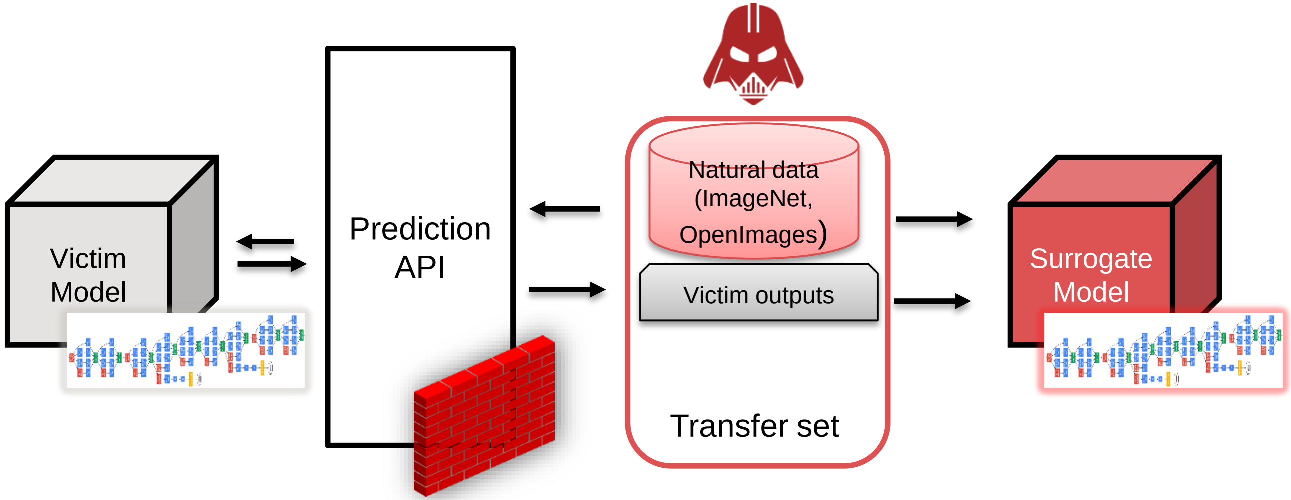
- From the same domain (e.g. images)
- Out of target train/test distribution

Query API to collect victim outputs

- Using ~ 100,000 queries
- API returns probability vector

Construct surrogate model

- Select a pre-trained model and retrain it with transfer set
- Takes ~ 3 days (Tesla V100 GPU, 10 GB; estimated cost $120-$170)



[1] Orekondy et al. - *Knockoff Nets: Stealing Functionality of Black-Box Models,* CVPR ‘19 (https://arxiv.org/abs/1812.02766)

Knockoff nets: Reproduction

Knockoff nets are effective against complex, pre-trained DNN models

Victim Model (Dataset-model)	Test Accuracy % (performance recovery)			
	Our reproduction		Reported in [1]	
	Victim Model	Surrogate Model	Victim Model	Surrogate Model
Caltech-RN34	74.1	72.2 (0.97x)	78.8	75.4 (0.96x)
CUBS-RN34	77.2	70.9 (0.91x)	76.5	68.0 (0.89x)
Diabetic-RN34	71.1	53.5 (0.75x)	58.1	47.7 (0.82x)
GTSRB-RN34	98.1	94.8 (0.96x)	-	-
CIFAR10-RN34	94.6	88.2 (0.93x)	-	-

[1] Orekondy et al. - *Knockoff Nets: Stealing Functionality of Black-Box Models.* CVPR'19 (https://arxiv.org/abs/1812.02766)

Revisiting the Adversary Model: Reduced Granularity of Prediction API's Output

Panda	99%
Mammal	99%
Vertebrate	99%
Terrestrial Animal	98%
Bear	94%
Nose	93%
Snout	92%
Nature Reserve	87%

Google Cloud Vision (top 20)

PREDICTED CONCEPT	PROBABILITY
wildlife	0.993
no person	0.988
zoo	0.974
panda	0.970
mammal	0.967
nature	0.964
animal	0.960
endangered species	0.958
cute	0.950
fur	0.948
outdoors	0.903
wild	0.901
portrait	0.885
endangered	0.842
frosty	0.840

Clarifai (top 20)

General Model –

Quickly understand objects, actions, scenes, and colors within an image.

mammal	0.99
animal	0.99
giant panda	0.99
carnivore	0.99
black color	0.91
coal black color	0.88

IBM Watson (top 10)

Revisiting the Adversary Model: Reduced Granularity of Prediction API's Output

Original adversary model in [1] expects a complete prediction vector for each query

Effectiveness degrades when prediction API gives truncated results (top label, rounded probabilities etc.)

Victim Model (Dataset-model)	Test Accuracy % (performance recovery)		
	Victim Model	Surrogate Model (full probability vector)	Surrogate Model (only top label)
Caltech-RN34 (257 classes)	74.1	72.2 (0.97x)	57.2 (0.77x)
CUBS-RN34 (200 classes)	77.2	70.9 (0.91x)	42.5 (0.55x)
Diabetic-RN34 (5 classes)	71.1	53.5 (0.75x)	53.5 (0.75x)
GTSRB-RN34 (43 classes)	98.1	94.8 (0.96x)	91.9 (0.93x)
CIFAR10-RN34 (10 classes)	94.6	88.2 (0.93x)	84.4 (0.89x)

[1] Orekondy et al. - *Knockoff Nets: Stealing Functionality of Black-Box Models,* CVPR '19 (https://arxiv.org/abs/1812.02766)

Revisiting the Adversary Model: Different Surrogate-Victim Architectures

Adversary model in [1] : victim model uses publicly available, pre-trained DNN models.

Effectiveness degrades when both victim and surrogate models are not based on pre-trained ImageNet DNNs.

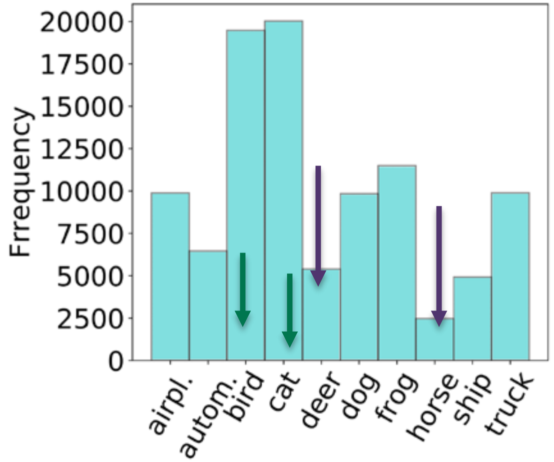
Victim Model (Dataset-model)	Test Accuracy % (performance recovery)		
	Victim Model	Surrogate Model (RN34)	Surrogate Model (VGG16)
GTSRB-RN34	98.1	94.8 (0.96x)	90.1 (0.91x)
CIFAR10-RN34	94.6	88.2 (0.93x)	82.9 (0.87x)
GTSRB-5L	91.5	54.5 (0.59x)	55.8 (0.60x)
CIFAR10-9L	84.5	67.5 (0.79x)	64.7(0.76x)

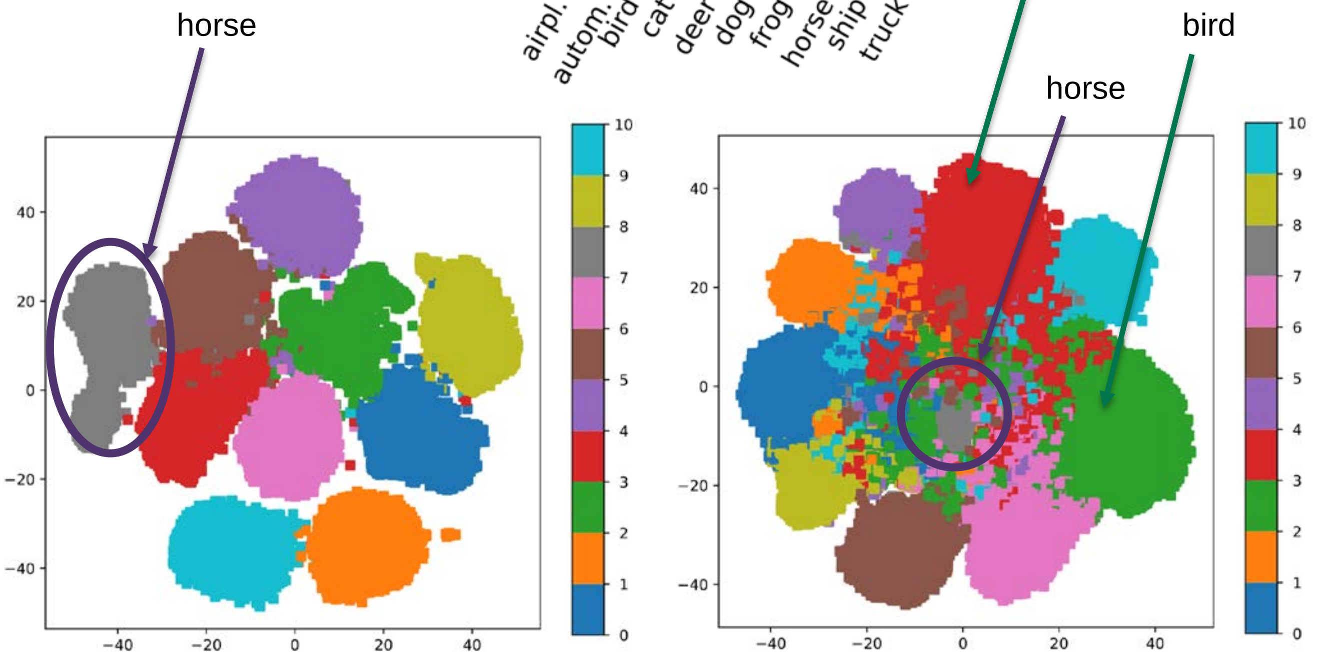
[1] Orekondy et al. - *Knockoff Nets: Stealing Functionality of Black-Box Models,* CVPR '19 (https://arxiv.org/abs/1812.02766)

Knockoff nets: Limitation

Knockoff nets cannot recover per-class performance of victim model

Class Name	Test accuracy % (performance recovery)	
	Victim Model (CIFAR-RN34) 94.6% on average	Surrogate Model 88.2% on average
Airplane (class 0)	95	88 (0.92x)
Automobile (class 1)	97	95 (0.97x)
Bird (class 2)	92	87 (0.94x)
Cat (class 3)	89	86 (0.96x)
Deer (class 4)	95	84 (0.88x)
Dog (class 5)	88	84 (0.95x)
Frog (class 6)	97	90 (0.92x)
Horse (class 7)	96	79 (0.82x)
Ship (class 8)	96	92 (0.95x)
Truck (class 9)	96	92 (0.95x)





Victim

Surrogate

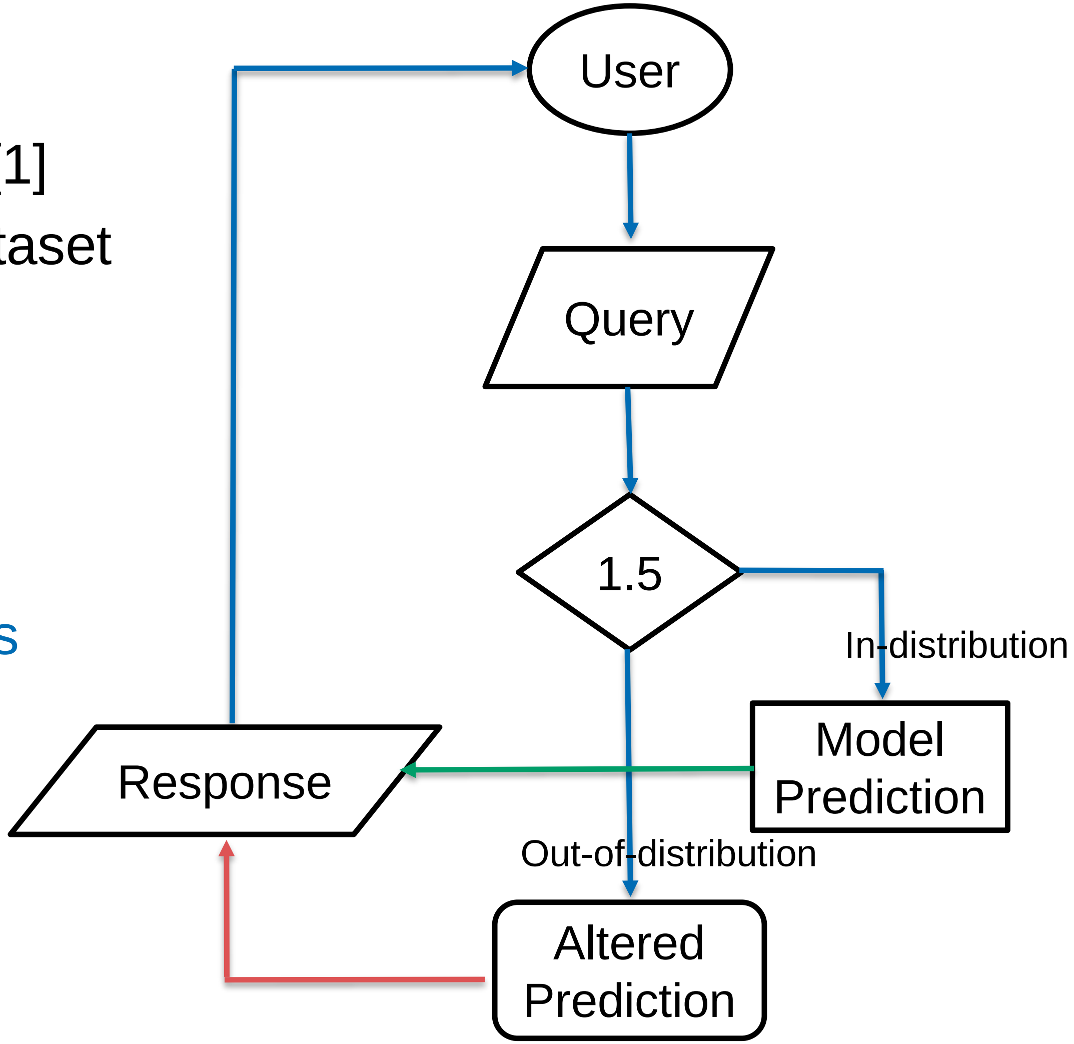
Knockoff nets: Detecting Attacker's Queries

Motivation

- Adversary is unaware of target distribution or task [1]
- Queries API with a random subset of public dataset used for a general task

Design

- Binary pre-classifier for incoming queries (1.5)
- Detect images from distribution other than victim's
- Give proper prediction only to in-distribution queries



[1] Orekondy et al. - *Knockoff Nets: Stealing Functionality of Black-Box Models.* CVPR'19 (https://arxiv.org/abs/1812.02766)

Knockoff nets: Detecting Attacker's Queries

Evaluation

- Trained ResNet classifiers to detect in and out-of-distribution queries
- High TPR/TNR on all datasets but Caltech (strong overlap with ImageNet, OpenImages)
- Performs better than state-of-the-art out-of-distribution methods (ODIN[1] , Mahal[2])

Victim Model (Dataset-model)	ImageNet		OpenImages	
	In-distribution (TPR%)	Out-of-distribution (TNR%)	In-distribution (TPR%)	Out-of-distribution (TNR%)
Caltech-RN34	63	56	61	59
CUBS-RN34	93	93	93	93
Diabetic-RN34	99	99	99	99
GTSRB-RN34	99	99	99	99
CIFAR10-RN34	96	96	96	96

[1] Liang et al. – *Enhancing the Reliability of Out-of-Distribution Image Detection in Neural Networks*. ICLR'18 (https://arxiv.org/abs/1706.02690)
[2] Lee et al. - *A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks. NIPS'18* (https://arxiv.org/abs/1807.03888)

Revisiting the Adversary Model: Access to In-distribution Data

The larger the overlap between attacker's transfer set and victim's training data, the less effective the detection.

A more realistic adversary

- Has access to more (unlimited) data (public databases, search engines)
- Has approximate knowledge of prediction APIs task (food, faces, birds etc.)
- Can evade detection mechanisms identifying out-of-distribution queries

Are there any prevention mechanisms?

- Stateful analysis → Sybil attacks
- Charging customers upfront → Reduced utility for benign users
- Restrict access to the API → Reduced utility for benign users

Outline: recap

Is model confidentiality important? Yes

Can models be extracted via their prediction APIs? Yes

What can be done to counter model extraction?

Atli et al. -*Extraction of Complex DNN Models: Real Threat or Boogeyman?* (https://arxiv.org/pdf/1910.05429.pdf, to appear in AAAI-EDSMLS 2020)

Existing Watermarking of DNNs[1]

Watermark embedding:

- Embed the watermark in the model **during the training phase**:
  - Choose specific labels to a set of samples (*trigger set)*
  - Train using training data + *trigger set*

Verification of ownership:

- Requires adversary to publicly expose the stolen model
- Query the model with the *trigger set*
- Verify watermark - predictions correspond to chosen labels

Limitations:

- Protects only against **physical theft** of the model
- *Model extraction* attacks steal the model **without the watermark**

[1] Yadi et al. - *Watermarking Deep Neural Networks by Backdooring*. USENIX '18 (https://www.usenix.org/node/217594)

Dynamic Adversarial Watermarking of DNNs

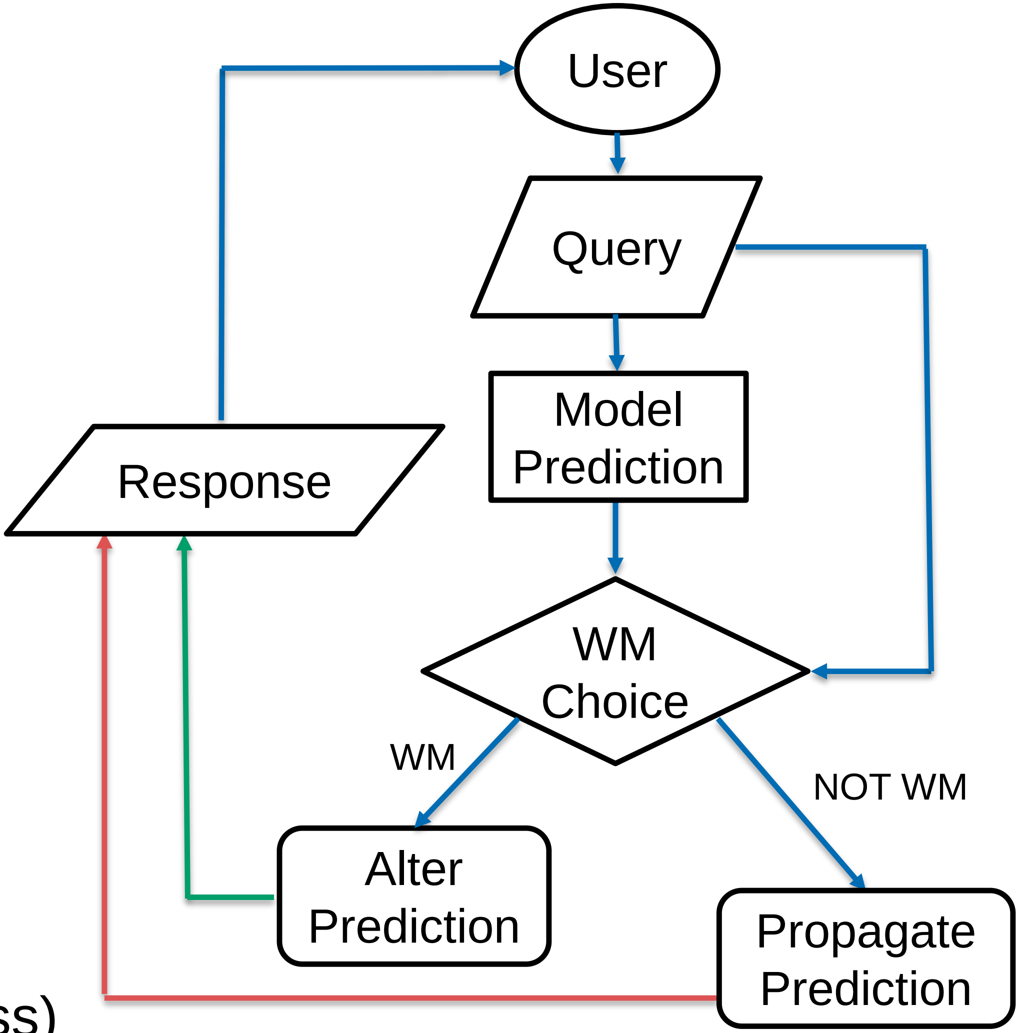
Goal: Watermark models obtained via *model extraction*

Our approach:

- Implemented as part of the **prediction API**
- Return **incorrect predictions** for several samples
- Adversary embeds the watermark during its training

Watermarking evaluation:

- *Unremovable* and *indistinguishable*
- Reliable demonstration of ownership
  - with confidence *$1- 2^{-64}$*
- **Defend against** *PRADA* [1] and *KnockOff* [2]
- Preserve victim *model utility* (*0.03-0.5%* accuracy loss)



[1] Juuti et al. - *PRADA: Protecting against DNN Model Stealing Attacks*. EuroS&P'19 (https://arxiv.org/abs/1805.02628)
[2] Orekondy et al. - *Knockoff Nets: Stealing Functionality of Black-Box Models*. CVPR'19 (https://arxiv.org/abs/1812.02766)
[3] Szyller et. al. - *DAWN: Dynamic Adversarial Watermarking of Neural Networks.* In submission. (https://arxiv.org/abs/1906.00830)

Takeaways

Is model confidentiality important? Yes

models constitute business advantage to model owners

Can models be extracted via their prediction APIs? Yes

Protecting model data via cryptography or hardware security is insufficient

What can be done to counter model extraction? Watermarking as a deterrence

Watermarking at the prediction API is feasible

Deserves to be considered as a deterrence against model stealing

More on our security + ML research at https://ssg.aalto.fi/research/projects/mlsec/

Backup slides

Knockoff nets: Detecting Attacker's Queries

- Adversary in [1] has no prior information or expectation about the output vector
- Prediction API gives shuffled prediction vector for detected out-of-distribution queries

Victim Model (Dataset-model)	Test Accuracy % (performance recovery)		
	Victim Model	Surrogate Model (correct probability list)	Surrogate Model (shuffled probability list)
Caltech-RN34 (257 classes)	74.1	72.2 (0.97x)	29.5 (0.39x)
CUBS-RN34 (200 classes)	77.2	70.9 (0.91x)	20.1 (0.26x)
Diabetic-RN34 (5 classes)	71.1	53.5 (0.75x)	28.0 (0.39x)
GTSRB-RN34 (43 classes)	98.1	94.8 (0.96x)	14.8 (0.15x)
CIFAR10-RN34 (10 classes)	94.6	88.2 (0.93x)	2.8 (0.02x)

[1] Orekondy et al. - *Knockoff Nets: Stealing Functionality of Black-Box Models.* CVPR'19 (https://arxiv.org/abs/1812.02766)

Revisiting the Adversary Model: Reduced Diversity of Adversarial Queries

- Original adversary model in [1] uses public dataset for general tasks. expects a complete prediction vector for each query
- Effectiveness degrades if attacker uses a dataset with low diversity

Victim Model (Dataset-model)	Test Accuracy % (performance recovery)		
	Victim Model	Surrogate Model (ImageNet subset)	Surrogate Model (Diabetic5)
Caltech-RN34 (257 classes)	74.1	72.2 (0.97x)	5.8 (0.07x)
CUBS-RN34 (200 classes)	77.2	70.9 (0.91x)	3.9 (0.05x)
Diabetic-RN34 (5 classes)	71.1	53.5 (0.75x)	71.2 (1.00x)
GTSRB-RN34 (43 classes)	98.1	94.8 (0.96x)	41.9 (0.44x)
CIFAR10-RN34 (10 classes)	94.6	88.2 (0.93x)	28.6 (0.32x)

[1] Orekondy et al. - *Knockoff Nets: Stealing Functionality of Black-Box Models.* CVPR’19 (https://arxiv.org/abs/1812.02766)

What can be done to counter model extraction?

A powerful (but realistic) adversary can extract complex real-life models

Detecting such an adversary is difficult/impossible

Can we deter such adversaries?