Stealing Machine Learning Models via Prediction APIs

Florian Tramèr, Fan Zhang, Ari Juels, Michael K. Reiter, Thomas Ristenpart

Usenix Security Symposium
Austin, Texas, USA
August, 11th 2016
Machine Learning (ML) Systems

1. Gather labeled data

\[ x^{(1)}, y^{(1)} \quad x^{(2)}, y^{(2)} \quad ... \]

- n-dimensional feature vector \( x \)
- Dependent variable \( y \)

2. Train ML model \( f \) from data

\[ f ( x ) = y \]

- Prediction
- Confidence

3. Use \( f \) in some application or publish it for others to use
Machine Learning as a Service (MLaaS)

Goal 1: Rich Prediction APIs
- Highly Available
- High-Precision Results

Goal 2: Model Confidentiality
- Model/Data Monetization
- Sensitive Data

Prediction API → Model f → Training API

$\text{input} \xrightarrow{\text{classification}} \text{Black Box} \xrightarrow{$$$ \text{per query}}$

Stealing Machine Learning Models via Prediction APIs
## Machine Learning as a Service (MLaaS)

<table>
<thead>
<tr>
<th>Service</th>
<th>Model types</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon</td>
<td>Logistic regressions</td>
</tr>
<tr>
<td>Google</td>
<td>??? (announced: logistic regressions, decision trees, neural networks, SVMs)</td>
</tr>
<tr>
<td>Microsoft</td>
<td>Logistic regressions, decision trees, neural networks, SVMs</td>
</tr>
<tr>
<td>PredictionIO</td>
<td>Logistic regressions, decision trees, SVMs (white-box)</td>
</tr>
<tr>
<td>BigML</td>
<td>Logistic regressions, decision trees</td>
</tr>
</tbody>
</table>

Sell Datasets – Models – Prediction Queries

$$ $$$

$$ $$$
Model Extraction Attacks

**Goal:** Adversarial client learns close approximation of \( f \) using as few queries as possible

---

Target: \( f(x) = f'(x) \) on \( \geq 99.9\% \) of inputs

---

**Applications:**

1) Undermine pay-for-prediction pricing model

2) Facilitate privacy attacks

3) Stepping stone to model-evasion
   
   [Lowd, Meek – 2005] [Srndic, Laskov – 2014]
Model Extraction Attacks (Prior Work)

**Goal:** Adversarial client learns close approximation of $f$ using as few queries as possible

If $f(x)$ is just a class label: *learning with membership queries*
- Boolean decision trees  [Kushilevitz, Mansour – 1993]
- Linear models (e.g., binary regression)  [Lowd, Meek – 2005]
Main Results

\[ f'(x) = f(x) \text{ on } 100\% \text{ of inputs} \]

100s-1000’s of online queries

- Logistic Regressions, Neural Networks, Decision Trees, SVMs
- Reverse-engineer model type & features

Improved Model-Inversion Attacks
[Fredrikson et al. 2015]
Model Extraction Example: Logistic Regression

Task: Facial Recognition of two people (binary classification)

$n+1$ parameters $w, b$ chosen using training set to minimize expected error

$$f(x) = \frac{1}{1+e^{-(w^T x + b)}}$$

$f$ maps features to predicted probability of being “Alice”
$\leq 0.5$ classify as “Bob”
$> 0.5$ classify as “Alice”

Generalize to $c > 2$ classes with multinomial logistic regression

$$f(x) = [p_1, p_2, ..., p_c]$$

predict label as $\text{argmax}_i p_i$

Feature vectors are pixel data e.g., $n = 92 \times 112 = 10,304$

Alice

Bob
**Model Extraction Example: Logistic Regression**

**Goal:** Adversarial client learns close approximation of $f$ using as few queries as possible

\[
f(x) = \frac{1}{1 + e^{-(w^T x + b)}}
\]

\[
\ln\left(\frac{f(x)}{1 - f(x)}\right) = w^T x + b
\]

Query $n+1$ random points $\Rightarrow$ solve a linear system of $n+1$ equations
Generic Equation-Solving Attacks

- **random inputs** $X$
- **MLaaS Service**
- **outputs** $Y$

- Solve **non-linear equation system** in the weights $W$
  - Optimization problem + gradient descent
  - “*Noiseless Machine Learning*”

- Multinomial Regressions & Deep Neural Networks:
  - $>99.9\%$ agreement between $f$ and $f'$
  - $\approx 1$ query per model parameter of $f$
  - $100$s - $1,000$s of queries / seconds to minutes
MLaaS: A Closer Look

Feature Extraction: (automated and partially documented)

Prediction API

- Class labels and confidence scores
- Support for partial inputs

Model f

Training API

Data

ML Model Type Selection:
logistic or linear regression

Steeling Machine Learning Models via Prediction APIs
Online Attack: AWS Machine Learning

Feature Extraction: Quantile Binning + One-Hot-Encoding

Reverse-engineered with partial queries and confidence scores

Model Choice: Logistic Regression

“Extract-and-test”

<table>
<thead>
<tr>
<th>Model</th>
<th>Online Queries</th>
<th>Time (s)</th>
<th>Price ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Handwritten Digits</td>
<td>650</td>
<td>70</td>
<td>0.07</td>
</tr>
<tr>
<td>Adult Census</td>
<td>1,485</td>
<td>149</td>
<td>0.15</td>
</tr>
</tbody>
</table>

Extracted model $f'$ agrees with $f$ on 100% of tested inputs
Application: Model-Inversion Attacks

Infer training data from trained models [Fredrikson et al. – 2015]

```
Inversion Attack
```

```
White-Box Attack
```

```
Training samples of 40 individuals
```

```
Massachusetts Institute of Technology
```

```
Inversion Attack
```

```
Data
```

```
Inversion Attack
```

```
Extraction Attack
```

```
Multinomial LR Model f
```

```
f(x) = f′(x) for >99.9% of inputs
```

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Attack against 1 individual</th>
<th>Attack against all 40 individuals</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Online Queries</td>
<td>Attack Time</td>
</tr>
<tr>
<td>Black-Box Inversion</td>
<td>20,600</td>
<td>24 min</td>
</tr>
<tr>
<td>[Fredrikson et al.]</td>
<td>×40</td>
<td></td>
</tr>
<tr>
<td>Extract-and-Invert</td>
<td>41,000</td>
<td>10 hours</td>
</tr>
<tr>
<td>(our work)</td>
<td>×1</td>
<td></td>
</tr>
</tbody>
</table>

Stealing Machine Learning Models via Prediction APIs

Usenix Security’16
August 11th, 2016

13
Extracting a Decision Tree

Confidence value derived from class distribution in the training set

Kushilevitz-Mansour (1992)

- Poly-time algorithm with membership queries only
- Only for Boolean trees, impractical complexity

(Ab)using Confidence Values

- Assumption: all tree leaves have unique confidence values
- Reconstruct tree decisions with “differential testing”
- Online attacks on BigML

Inputs $x$ and $x'$ differ in a single feature

Different leaves are reached

Tree “splits” on this feature
Countermeasures

How to prevent extraction?

**API Minimization**

\[ f(x) = y \]

- Prediction = class label only
- *Learning with Membership Queries*

**Attack on Linear Classifiers [Lowd,Meek – 2005]**

classify as “+” if \( w^*x + b > 0 \)
and “-” otherwise

\[
f(x) = \text{sign}(w^*x + b)\]

1. Find points on **decision boundary** \( (w^*x + b = 0) \)
   - Find a “+” and a “-”
   - **Line search** between the two points
2. Reconstruct \( w \) and \( b \) (up to scaling factor)
Generic Model Retraining Attacks

- Extend the Lowd-Meek approach to non-linear models
- **Active Learning:**
  - Query points close to “decision boundary”
  - Update $f'$ to fit these points
- Multinomial Regressions, Neural Networks, SVMs:
  - $>99\%$ agreement between $f$ and $f'$
  - $\approx 100$ queries per model parameter of $f$

$\approx 100\times$ less efficient than equation-solving

query more points here
Conclusion

Rich prediction APIs ➡️ Model & data confidentiality

Efficient Model-Extraction Attacks

• Logistic Regressions, Neural Networks, Decision Trees, SVMs
• Reverse-engineering of model type, feature extractors
• Active learning attacks in membership-query setting

Applications

• Sidestep model monetization
• Boost other attacks: privacy breaches, model evasion

Thanks! Find out more: https://github.com/ftramer/Steal-ML