On Learning and Recognition of Secure Patterns

Battista Biggio, Fabio Roli, Giorgio Fumera

Dept. Of Electrical and Electronic Engineering
University of Cagliari, Italy

MLDM.it @ AI*IA
Pisa, Italy, Dec. 11th, 2014
Secure Patterns in Nature

- Learning of secure patterns is a well-known problem in nature
  - Mimicry and camouflage
  - Arms race between predators and preys
Secure Patterns in Computer Security

- Similar phenomenon in machine learning and computer security
  - Obfuscation and polymorphism to hide malicious content

**Spam emails**

Start 2007 with a bang!
Make WBFS YOUR PORTFOLIO’s first winner of the year...

**Malware**

```
<script type="text/javascript"
src="http://palwas.servehttp.com//ml.php"></script>

... var PGuDO0uq19+PGuDO0uq20;
EbphZcei=PVqIW5sV.replace(/jTUZZ/g,"%");
var eWfleJqh=unescape;
Var NxfaGVHq="pqXdQ23KZri130";
q9124=this; var SkuyuppD=q9124["WYd1GoGYc2uG1mYGe2YntY".replace(/[Y12W12G:\]/g,"")];SkuyuppD.write(eWfleJqh(EbphZcei));
...```

http://pralab.diee.unica.it
Can Machine Learning Be Secure?

• Underlying assumption of machine learning techniques
  – Training and test data are sampled from the same distribution

• In practice
  – The classifier generalizes well from known examples (+ random noise)
  – ... but can not cope with carefully-crafted attacks!

Evasion attacks:
manipulation of test samples to evade detection

Poisoning attacks:
manipulation of training data to cause misclassifications at test time

(2) B. Biggio, G. Fumera, F. Roli. Security evaluation of pattern classifiers under attack. IEEE Trans. on Knowl. and Data Engineering, 2014
Evasion Attacks

A Simple Example

- **Problem:** how to evade a linear (trained) classifier?

\[
\begin{align*}
  f(x) &= \text{sign}(w^T x) \\
  +6 &> 0, \text{ SPAM (correctly classified)} \\
  f(x) &= \text{sign}(w^T x') \\
  +3 &- 4 < 0, \text{ HAM (misclassified email)}
\end{align*}
\]
Gradient-descent Evasion Attacks

- **Goal:** maximum-confidence evasion
- **Knowledge:** perfect
- **Attack strategy:**
  \[
  \min_{x'} g(x') \\
  \text{s.t. } d(x, x') \leq d_{\max}
  \]

- Non-linear, constrained optimization
  - **Gradient descent:** approximate solution for smooth functions

- Gradients of \( g(x) \) can be analytically computed in many cases
  - SVMs, Neural networks

\[
f(x) = \text{sign}(g(x)) = \begin{cases} 
+1, \text{ malicious} \\
-1, \text{ legitimate}
\end{cases}
\]
Computing Descent Directions

Support vector machines

\[ g(x) = \sum_i \alpha_i y_i k(x, x_i) + b, \quad \nabla g(x) = \sum_i \alpha_i y_i \nabla k(x, x_i) \]

**RBF kernel gradient:**

\[ \nabla k(x, x_i) = -2\gamma \exp\left\{-\gamma \|x - x_i\|^2\right\}(x - x_i) \]

Neural networks

\[ g(x) = \left[1 + \exp\left(-\sum_{k=1}^{m} w_k \delta_k(x)\right)\right]^{-1} \]

\[ \frac{\partial g(x)}{\partial x_f} = g(x)(1 - g(x)) \sum_{k=1}^{m} w_k \delta_k(x) (1 - \delta_k(x)) v_{k_f} \]
An Example on Handwritten Digits

- Nonlinear SVM (RBF kernel) to discriminate between ‘3’ and ‘7’

- **Features**: gray-level pixel values
  - 28 x 28 image = 784 features

Before attack (3 vs 7) | After attack, g(x)=0 | After attack, last iter. | g(x) | Number of modified gray-level values
---|---|---|---|---
![Image](http://pralab.diee.unica.it) | ![Image](http://pralab.diee.unica.it) | ![Image](http://pralab.diee.unica.it) | ![Image](http://pralab.diee.unica.it) |

Few modifications are enough to evade detection! … without even mimicking the targeted class (‘7’)

http://pralab.diee.unica.it
Bounding the Adversary’s Knowledge
Limited knowledge attacks

- Only feature representation and learning algorithm are known
- Surrogate data sampled from the same distribution as the classifier’s training data
- Classifier’s feedback to label surrogate data
Experiments on PDF Malware Detection

- **PDF**: hierarchy of interconnected objects (keyword/value pairs)

  13 0 obj
  << /Kids [ 1 0 R 11 0 R ]
  /Type /Page
  ... >> end obj
  17 0 obj
  << /Type /Encoding
  /Differences [ 0 /C0032 ] >>
  endobj

- **Adversary’s capability**
  - adding up to \(d_{\text{max}}\) objects to the PDF
  - removing objects may compromise the PDF file (and embedded malware code)!

\[
\begin{align*}
\text{Features: } & \text{keyword count} \\
\end{align*}
\]

\[
\begin{align*}
/Type & \quad 2 \\
/Page & \quad 1 \\
/Encoding & \quad 1 \\
\end{align*}
\]

\[
\begin{align*}
\min & \ g(x') \\
\text{s.t.} & \quad d(x, x') \leq d_{\text{max}} \\
& \quad x \leq x'
\end{align*}
\]
Experiments on PDF Malware Detection

- **Dataset**: 500 malware samples (*Contagio*), 500 benign (Internet)
  - Targeted (surrogate) classifier trained on 500 (100) samples

- **Evasion rate** (FN) at FP=1% vs max. number of added keywords
  - Averaged on 5 repetitions
  - Perfect knowledge (PK); Limited knowledge (LK)
Poisoning Attacks against SVMs

Poisoning SVMs

• Poisoning attack
  – Goal: to maximize classification error by injecting samples into TR

• Attack strategy
  – optimal attack point $x_c$ in TR that maximizes classification error
Poisoning SVMs

- Poisoning attack
  - Goal: to maximize classification error by injecting samples into TR

- Attack strategy
  - optimal attack point $x_c$ in TR that maximizes classification error
Poisoning Attack Algorithm

• Max. classification error $L(x_c)$ w.r.t. $x_c$ through gradient ascent

• Gradient is not easy to compute
  – The training point affects the classification function
  – Details of the derivation are in the paper

**Input:** $\mathcal{D}_{tr}$, $\mathcal{D}_{val}$, $y_c$, $x_c^{(0)}$, t.

**Output:** $x_c$.

1: $\{\alpha_i, b\} \leftarrow$ learn an SVM on $\mathcal{D}_{tr}$.
2: $x_c \leftarrow x_c^{(0)}$
3: repeat
4: Update $\{\alpha_i, b\}$ on $\mathcal{D}_{tr} \cup \{x_c, y_c\}$
5: Compute $\nabla L(x_c)$ on $\mathcal{D}_{val}$.
6: $x'_c \leftarrow x_c + t\nabla L(x_c)$
7: until $L(x'_c) - L(x_c) < \epsilon$
8: return $x_c$

Experiments on the MNIST digits

Single-point attack

- Linear SVM; 784 features; TR: 100; VAL: 500; TS: about 2000
  - ‘0’ is the malicious (attacking) class
  - ‘4’ is the legitimate (attacked) one
Experiments on the MNIST digits

Multiple-point attack

- Linear SVM; 784 features; TR: 100; VAL: 500; TS: about 2000
  - ‘0’ is the malicious (attacking) class
  - ‘4’ is the legitimate (attacked) one
Conclusions and Future Work

• Learning-based systems can be vulnerable to well-crafted, sophisticated attacks devised by skilled attackers
  – … that exploit specific vulnerabilities of machine learning algorithms!

• Future (and ongoing) work
  – Secure Learning, Clustering and Feature Selection/Reduction
Thanks for your attention!

Any questions?