PINFER: PRIVACY-PRESERVING INFEERENCE

DPM 2019
Luxembourg • Sep. 26, 2019
Marc Joye  Fabien Petitcolas
MACHINE LEARNING AS A SERVICE — GENERIC MODEL

Client

Cloud (Server)

1 exchange of messages

I don’t like to share my private information!
REQUIREMENTS AND SOLUTIONS

Security requirements

- The server learns nothing about the client’s input
- The server does not learn the output of the calculation
- The client learns nothing about the ML model

Proposed solutions

Private evaluation for:

1. Linear regression
2. Logistic regression
3. Binary classification
   - Support Vector Machines (SVM)
   - requires a private comparison protocol (e.g., DGK+)
4. Neural networks
   - Sign or ReLU activation functions
   - 1 interaction per layer
LINEAR PREDICTION MODEL

• Input
  1. Server’s ML model: $\theta = (\theta_0, \ldots, \theta_d) \in \mathbb{R}^{d+1}$
  2. User’s feature vector: $x = (1, x_1, \ldots, x_d) \in \{1\} \times \mathbb{R}^d$

• Output

$$h_\theta(x) = g(\theta^T x) \quad \text{in many cases}$$
LINEAR PREDICTION MODEL — EVALUATION FUNCTION $g$

**Linear Regression** [real-valued output]

$$g = 1d$$

**Logistic Regression** [probability]

$$g = \sigma \quad \text{where} \quad \sigma(s) = \frac{\exp(s)}{1+\exp(s)}$$

**Linear Classification** [binary decision]

$$g = \text{sign}$$

**Rectified linear unit (ReLU) [neural networks]**

$$g(s) = \begin{cases} 
0 & \text{if } s < 0 \\
\text{s} & \text{otherwise}
\end{cases}$$
Model evaluation: \( \hat{y} = g(\theta^T x) \)

1. Compute \([x]\)
2. Compute \([g(\theta^T x)]\)
3. Decrypt \([g(\theta^T x)]\)
   Set \( \hat{y} = g(\theta^T x) \)
We only require linearly homomorphic encryption:

\[ \text{Enc}_{pk}(m_1) \boxplus \text{Enc}_{pk}(m_2) = \text{Enc}_{pk}(m_1 + m_2) \]

NOT fully homomorphic encryption:

\[ \text{Enc}_{pk}(m_1) \boxdot \text{Enc}_{pk}(m_2) = \text{Enc}_{pk}(m_1 \cdot m_2) \]

Benefits

- Simpler implementation
- Faster computation
Since $[\cdot]$ is homomorphic

$$[\theta^T x] = [\theta_0 + \sum_{i=1}^{d} \theta_i x_i] = [\theta_0] \boxplus [\theta_1 x_1] \boxplus \cdots \boxplus [\theta_d x_d]$$

and, for $1 \leq i \leq d$,

$$[\theta_i x_i] = [x_i] \boxplus \cdots \boxplus [x_i] := \theta_i \odot [x_i]$$

**Example (Paillier’s cryptosystem)**

- $[m] = (1 + N)^m r^N \mod N^2$
- $[m_1 + m_2] = [m_1] \cdot [m_2] \mod N^2$
- $[m_1 - m_2] = [m_1] / [m_2] \mod N^2$
- $a \odot [m] = [m]^a \mod N^2 \implies [\theta^T x]$ requires $d$ exponentiations modulo $N^2$
IF EVALUATION FUNCTION $g$ IS NON-LINEAR

- $g$ is non-linear but injective (e.g., $\sigma$)
  - Server computes $[[\theta^T x]]$
  - Client obtains $\theta^T x$ and simply applies $g$ and learns no more
    (by definition: $g(a) = g(b) \implies a = b$)

- $g$ is non-linear and non-injective (e.g., sign, ReLU)
  - Use set of tools and tricks
    - DGK+ comparison protocol
    - Simple masking with a random value
    - Masking and scaling of inner product
    - Variant of oblivious transfer (two possible ciphers sent)
  - Dual setup
    - Server publishes $pk_s$ and $[[\theta]]_s$
    - Still one round of messages!
Activation function $g_j^{(l)}$
NUMERICAL EXPERIMENTS

- Implementation (not much optimised)
  - Python
  - Intel i7-4770, 3.4GHz
  - GMP library (power exponentiation)
  - Fixed precision (53 bits)

- Parameters
  - Public datasets and randomly generated ones
  - Models with 30 to 7994 features
  - Key sizes: 1388 to 2440 bits

- Message overhead proportional to:
  - Key size
  - Number of features (or number of bits in DGK+)
  - Number of layers (FFNN)
<table>
<thead>
<tr>
<th>Protocol</th>
<th>Protocol step</th>
<th>Size</th>
<th>(kB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear regression (core)</td>
<td>Client sends:</td>
<td>( \ell_M + d \cdot 2\ell_M )</td>
<td>( \approx 15 )</td>
</tr>
<tr>
<td></td>
<td>( pk_C, [x_i], 1 \leq i \leq d )</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Server sends: ( t )</td>
<td>( \approx 2\ell_M )</td>
<td>(&lt; 1 )</td>
</tr>
<tr>
<td>SVM classification (core)</td>
<td>Client sends ( t^*, [\mu_i], 0 \leq i \leq \ell - 1 )</td>
<td>( 2\ell_M + \ell \cdot 2\ell_M )</td>
<td>( \approx 29 )</td>
</tr>
<tr>
<td></td>
<td>Server sends ( [h_i^*], -1 \leq i \leq \ell - 1 )</td>
<td>( (\ell + 1) \cdot 2\ell_M )</td>
<td>( \approx 30 )</td>
</tr>
<tr>
<td>FFNN sign act. (core)</td>
<td>Server sends ( t^*, {\mu_i}, 0 \leq i \leq \ell - 1 )</td>
<td>( L \cdot d \cdot (\ell + 1) \cdot 2\ell_M )</td>
<td>( 2,655 ) (885 per layer)</td>
</tr>
<tr>
<td></td>
<td>Client sends ( [\hat{y}^<em>], {h_i^</em>}, -1 \leq i \leq \ell - 1 )</td>
<td>( L \cdot d \cdot (\ell + 2) \cdot 2\ell_M )</td>
<td>( 2,700 ) (900 per layer)</td>
</tr>
</tbody>
</table>

1Features: \( d = 30 \); key-size \( \ell_M = 2048 \); \( \kappa = 95 \); layers \( L = 3 \); Precision \( P = 53 \); Inner-product bound: \( \ell = 58 \)
RESULTS: LINEAR REGRESSION

Private LR: 70 features

Private linear regression (core protocol)
Dataset: audiology, # features: 70

Average computing time (ms) over 1000 trials

Private LR: 7994 features

Private linear regression (core protocol)
Dataset: enron, # features: 7994

On Intel i7-4770, 3.4GHz
RESULTS: SUPPORT VECTOR MACHINE CLASSIFICATION

Private SVM: 70 features

Private SVM classification (core protocol)
Dataset: audiology, # features: 70

Private SVM: 7994 features

Private SVM classification (core protocol)
Dataset: enron, # features: 7994

On Intel i7-4770, 3.4GHz

DGK+ comparison is the main limiting factor
RESULTS: NEURAL NETWORKS

Private NNs: 10 features | 3 layers

- Simple FFNN with sign activation (heuristic solution)
- Dataset: random, # features: 10, # layers: 3

Average computing time (ms) over 100 trials

On Intel i7-4770, 3.4GHz

DGK+ comparison is the main limiting factor
THANK YOU!