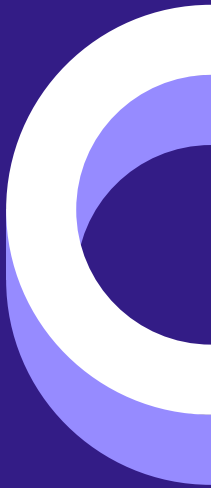
OneSpan
Innovation Centre

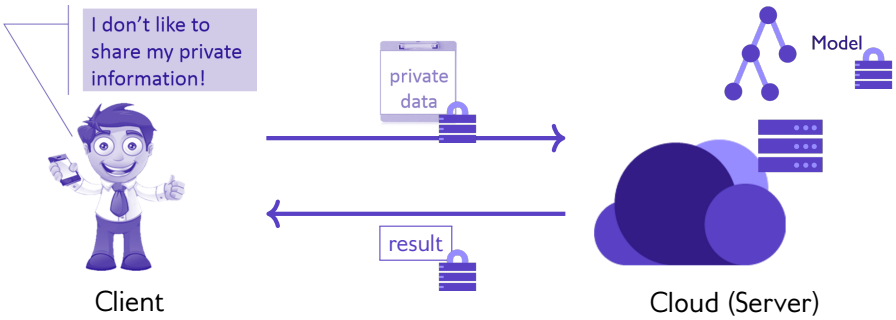
PINFER: PRIVACY-PRESERVING INFERENCE

DPM 2019
Luxembourg ▪ Sep. 26, 2019

Marc Joye Fabien Petitcolas



MACHINE LEARNING AS A SERVICE — GENERIC MODEL



1 exchange of messages

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

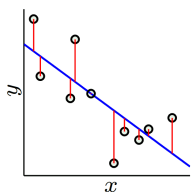
Private evaluation for:

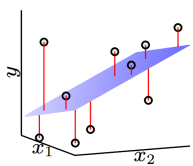
1. Linear regression
2. Logistic regression
3. Binary classification
   - Support Vector Machines (SVM)
   - requires a private comparison protocol (e.g., DGK+)
4. Neural networks
   - Sign or ReLU activation functions
   - 1 interaction per layer

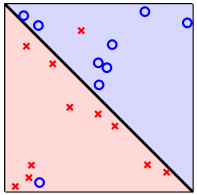
LINEAR PREDICTION MODEL

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

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = g(\boldsymbol{\theta}^\intercal \boldsymbol{x}) \quad \text{in many cases}$$







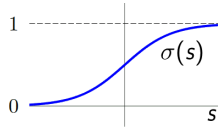
LINEAR PREDICTION MODEL — EVALUATION FUNCTION g

Linear Regression [real-valued output]

$$g = \mathrm{Id}$$

Logistic Regression [probability]

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



Linear Classification [binary decision]

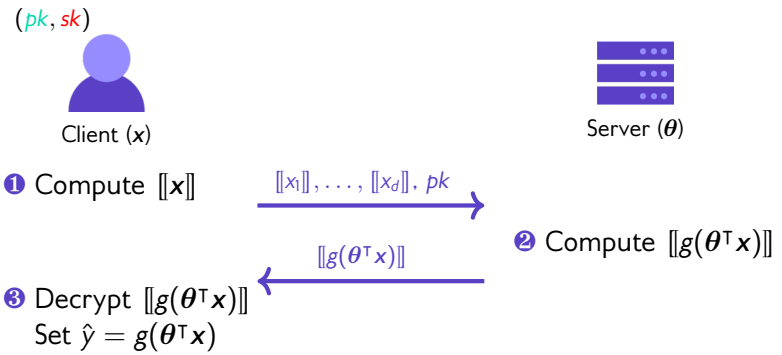
$$g = \mathrm{sign}$$

Rectified linear unit (ReLU) [neural networks]

$$g(s) = \begin{cases} 0 & \text{if } s < 0 \\ s & \text{otherwise} \end{cases}$$

LINEAR PREDICTION MODEL WITH ENCRYPTION

Model evaluation: $\hat{y} = g(\boldsymbol{\theta}^\top \boldsymbol{x})$



LINEARLY HOMOMORPHIC ENCRYPTION

- We only require linearly homomorphic encryption:

$$\mathrm{Enc}_{pk}(m_1) \boxplus \mathrm{Enc}_{pk}(m_2) = \mathrm{Enc}_{pk}(m_1 + m_2)$$

- NOT fully homomorphic encryption:

$$\mathrm{Enc}_{pk}(m_1) \boxplus \mathrm{Enc}_{pk}(m_2) = \mathrm{Enc}_{pk}(m_1 + m_2)$$
$$\mathrm{Enc}_{pk}(m_1) \boxdot \mathrm{Enc}_{pk}(m_2) = \mathrm{Enc}_{pk}(m_1 \cdot m_2)$$

- Benefits
  - Simpler implementation
  - Faster computation

- Since $[\![\cdot]\!]$ is homomorphic

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

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

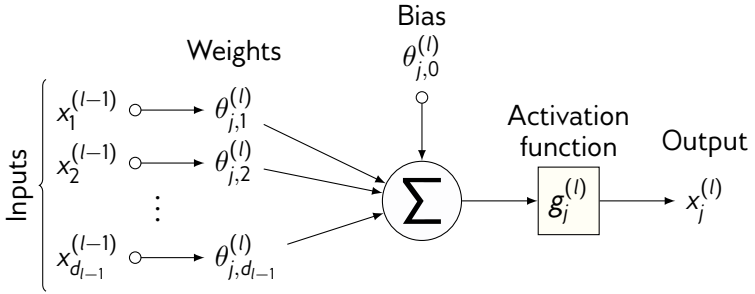
$$[\![\theta_i x_i]\!] = \underbrace{[\![x_i]\!] \boxplus \cdots \boxplus [\![x_i]\!]}_{\theta_i \text{ times}} := \theta_i \odot [\![x_i]\!]$$

Example (Paillier's cryptosystem)

- $[\![m]\!] = (1+N)^m r^N \bmod N^2$
- $[\![m_1 + m_2]\!] = [\![m_1]\!] \cdot [\![m_2]\!] \mod N^2$
- $[\![m_1 - m_2]\!] = [\![m_1]\!] / [\![m_2]\!] \mod N^2$
- $a \odot [\![m]\!] = [\![m]\!]^a \bmod N^2 \implies [\![\boldsymbol{\theta}^\mathsf{T}\boldsymbol{x}]\!]$ requires d exponentiations modulo N^2

IF EVALUATION FUNCTION g IS NON-LINEAR

- g is non-linear but injective (e.g., σ)
  - Server computes $[\![\boldsymbol{\theta}^\intercal \boldsymbol{x}]\!]$
  - Client obtains $\boldsymbol{\theta}^\intercal \boldsymbol{x}$ 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 $\{\![\boldsymbol{\theta}]\!\}_s$
    - Still one round of messages!



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)

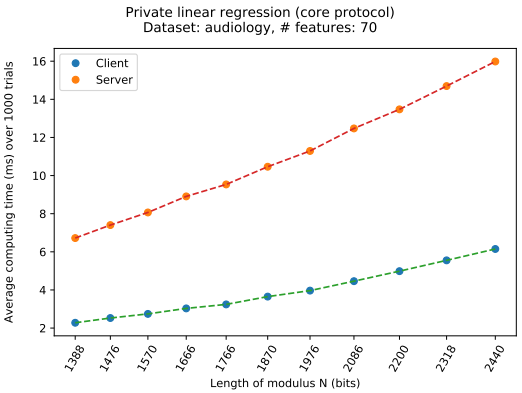
MESSAGE OVERHEAD

Protocol	**Protocol step**	**Size**	(kB) [1]
Linear regression (core)	Client sends: pk_C, $[\![x_i]\!]$, $1 \le i \le d$	$\ell_M + d \cdot 2\ell_M$	≈ 15
	Server sends: $\mathfrak{t}$	$\approx 2\ell_M$	< 1
SVM classification (core)	Client sends $\mathfrak{t}^*$, $[\![\mu_i]\!]$, $0 \le i \le \ell - 1$	$2\ell_M + \ell \cdot 2\ell_M$	≈ 29
	Server sends $[\![h_i^*]\!]$, $-1 \le i \le \ell - 1$	$(\ell + 1) \cdot 2\ell_M$	≈ 30
FFNN sign act. (core)	Server sends $\mathfrak{t}^*$, $\{\!\{\mu_i\}\!\}_s$, $0 \le i \le \ell - 1$	$L \cdot d \cdot (\ell + 1) \cdot 2\ell_M$	2,655 (885 per layer)
	Client sends $[\![\hat{y}^*]\!]$, $\{\!\{h_i^*\}\!\}_s$, $-1 \le i \le \ell - 1$	$L \cdot d \cdot (\ell + 2) \cdot 2\ell_M$	2,700 (900 per layer)

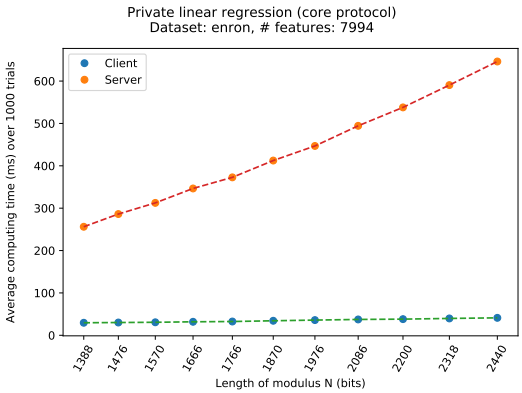
[1] Features: $d = 30$; key-size $\ell_M = 2048$; $\kappa = 95$; layers $L = 3$; Precision $P = 53$; Inner-product bound: $\ell = 58$

RESULTS: LINEAR REGRESSION

Private LR: 70 features



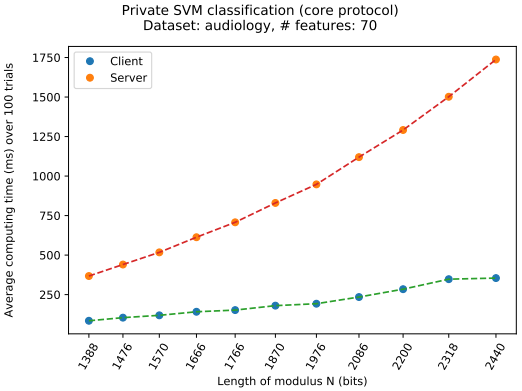
Private LR: 7994 features



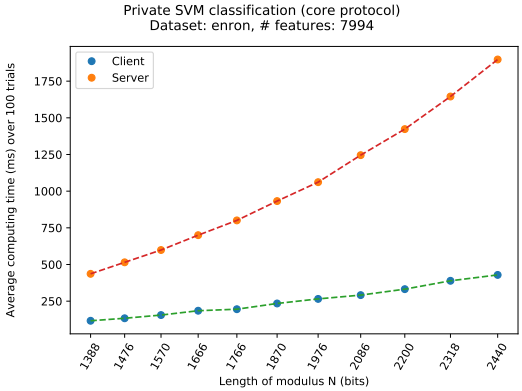
On Intel i7-4770, 3.4GHz

RESULTS: SUPPORT VECTOR MACHINE CLASSIFICATION

Private SVM: 70 features



Private SVM: 7994 features

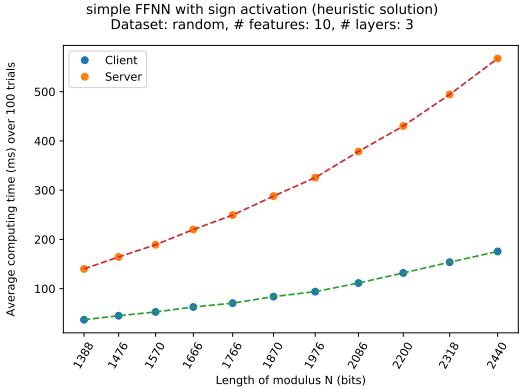


On Intel i7-4770, 3.4GHz

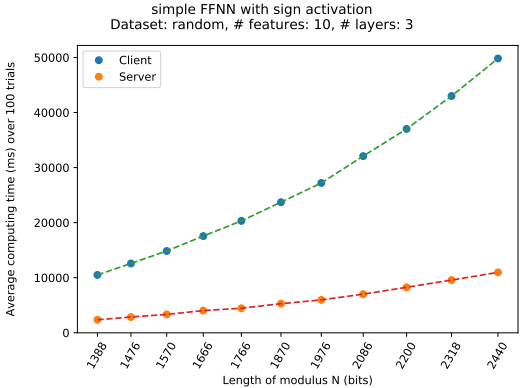
DGK+ comparison is the main limiting factor

RESULTS: NEURAL NETWORKS

Private NNs: 10 features | 3 layers



Private NNs: 10 features | 3 layers



On Intel i7-4770, 3.4GHz

DGK+ comparison is the main limiting factor

COMMENTS/QUESTIONS?