Encrypted Learning

Carlos Aguilar Melchor
carlos.aguilar-melchor@isae-supaero.fr

Équipe RESCOM - DISC - ISAE SUPAERO

Preheating...

- Go to `https://colab.research.google.com`
- Create a Python3 notebook
- Execute `!pip3 install Pyfhel`

Vox populi

Voting instructions:

Open your smartphone browser and go to

live.voxvote.com

and enter the following code

PIN: 40025

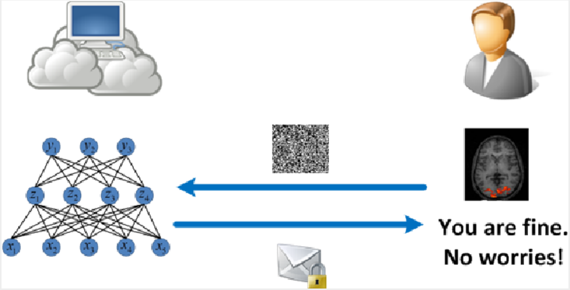
Alternative: Scan this QR-code and you are immediatly logged in

Alternative: Download the VoxVote app from

Outline

Encrypted Neural Networks

Cryptonets [Dowlin et al ICML2016]



Recognized MNIST encrypted digit with accurracy 97% in a second

Further works

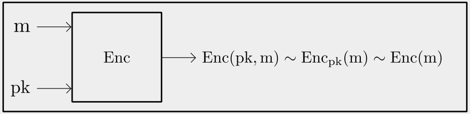
- Much faster
- CIFAR
- Learning phase

Many publications per year, evolving very fast

Outline

Deterministic Encryption

Deterministic Encryption : one ciphertext per plaintext



Example (Raw-RSA):

- $\mathtt{Enc}((\mathtt{N}, \mathtt{e}), \mathtt{m}) = \mathtt{m}^{\mathtt{e}} \bmod \mathtt{N}$ where $(\mathtt{N}, \mathtt{e}) = \mathtt{pk}$ is called a public key
- $\mathtt{Dec}((\mathtt{N}, \mathtt{d}), \mathtt{c}) = \mathtt{c}^{\mathtt{d}} \bmod \mathtt{N}$ where $(\mathtt{N}, \mathtt{d}) = \mathtt{sk}$ is called the private key
- Correct because $\mathtt{N}, \mathtt{e}, \mathtt{d}$ s.t. for all $\mathtt{m}$ we have $(\mathtt{m}^{\mathtt{e}})^{\mathtt{d}} \bmod \mathtt{N} = \mathtt{m}$

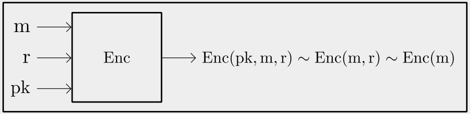
Indistinguishability issue

- Alice public key is (N, e) with $N = 9203904823049823098420393$ $e = 65537$
- Alice sends a ciphertext $c = 3448251181187896868804359$
- You know she just sent her (encrypted) pay rise which is either 300 euros or 500 euros

Can you decrypt ?

Randomized Encryption

Randomized Encryption : many ciphertexts for a plaintext



Exemple : Simplified RSA-PKCS1.5 $\texttt{Enc}((\mathrm{N}, \mathrm{e}), \mathrm{m}, \mathrm{r}) = (\mathrm{r}||\mathrm{m})^{\mathrm{e}} \bmod \mathrm{N}$, r having at least 8 random bytes

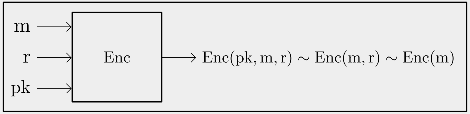
Indistinguishability issue

- Alice public key is (N, e) with $N = 9203904823049823098420393$ $e = 65537$
- Alice sends a ciphertext $c = 332492881708364402772076$
- You know she just sent her (encrypted) pay rise which is either 300 euros or 500 euros

Can you decrypt ?

Randomized Encryption

Randomized Encryption : many ciphertexts for a plaintext



Exemple : Simplified RSA-PKCS1.5 $\texttt{Enc}((\mathtt{N}, \mathtt{e}), \mathtt{m}, \mathtt{r}) = (\mathtt{r}||\mathtt{m})^{\mathtt{e}} \bmod \mathtt{N}$, r having at least 8 random bytes

Assumption: Indistinguishability for any pair of plaintext [GM 82]

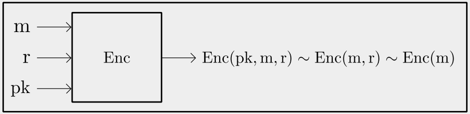
- For any two m_0, m_1, when given c a ciphertext of m_b the best one can do is a blind guess of b
- Formal proof $\Rightarrow$ Ciphertexts give no information about the associated plaintexts
- **Fun fact:** we cannot even decide if two ciphertexts are associated to the same plaintext

Two ciphertexts of $m = 0$ and one of $m = 1$ which is which?

`[31988785498412043126944724, 5293165912954825549052827, 3822143322044367965925822]`

Randomized Encryption

Randomized Encryption : many ciphertexts for a plaintext



Exemple : Simplified RSA-PKCS1.5 $\texttt{Enc}((\texttt{N}, \texttt{e}), \texttt{m}, \texttt{r}) = (\texttt{r}||\texttt{m})^{\texttt{e}} \bmod \texttt{N}$, r having at least 8 random bytes

Assumption: Indistinguishability for any pair of plaintext [GM 82]

- For any two m_0, m_1, when given c a ciphertext of m_b the best one can do is a blind guess of b
- Formal proof $\Rightarrow$ Ciphertexts give no information about the associated plaintexts
- **Fun fact:** we cannot even decide if two ciphertexts are associated to the same plaintext

Two ciphertexts of $m = 0$ and one of $m = 1$ which is which?
`[3198878549841204312694472, 5293165912954825549052827, 3822143322044367965925822]`
Did I lie?

Outline

The Fully Homomorphic Encryption Conjecture (1/3)

On Data Banks and Privacy Homomorphisms [RAD 1978]

There are secure cryptosystems with operations MUL and ADD such that:

- $\mathtt{MUL}(\mathtt{Enc}(\mathtt{x}_1, \mathtt{r}_1), \mathtt{Enc}(\mathtt{x}_2, \mathtt{r}_2)) = \mathtt{Enc}(\mathtt{x}_1 \mathtt{x}_2, \mathtt{r}_3)$
- $\mathtt{ADD}(\mathtt{Enc}(\mathtt{x}_1, \mathtt{r}_1), \mathtt{Enc}(\mathtt{x}_2, \mathtt{r}_2)) = \mathtt{Enc}(\mathtt{x}_1 + \mathtt{x}_2, \mathtt{r}_4)$

sums and products being computed modulo a given integer p.

Train yourself! (polynomials)

α is an encryption of an unknown number a and β an encryption of an unknown number b

- (Example) Compute an encryption of $a + b \bmod p$:
- Compute an encryption of $a^2 + b \bmod p$:
- Compute an encryption of $a^2 + b^2 + ab \bmod p$:

The Fully Homomorphic Encryption Conjecture (2/3)

Implications

f polynomial modulo p : $\mathtt{Enc}(\mathrm{x}_1, \mathrm{r}_1), \ldots, \mathtt{Enc}(\mathrm{x}_n, \mathrm{r}_n) \xrightarrow{f_H} \mathtt{Enc}(\mathrm{f}(\mathrm{x}_1, \ldots, \mathrm{x}_n), \mathrm{r})$

$\mathcal{C}$ circuit : $\mathtt{Enc}(\mathrm{b}_1, \mathrm{r}_1), \ldots, \mathtt{Enc}(\mathrm{b}_n, \mathrm{r}_n) \xrightarrow{\mathcal{C}_H} \mathtt{Enc}(\mathcal{C}(\mathrm{b}_1, \ldots, \mathrm{b}_n), \mathrm{r})$.
(working $\mathtt{mod}\ 2$, $a + b \leftrightarrow a \oplus b$ and $a * b \leftrightarrow a \,\&\, b$)

Train yourself! (circuits)

α is an encryption of an unknown bit a and β an encryption of an unknown bit b

- Assume computations are done $\mathrm{mod}\ 2$ and that you can call $\mathtt{Enc}$ (you know pk)
- Compute an encryption of $a \oplus b$:
- Compute an encryption of $a \,\&\, b$:
- Compute an encryption of $!a$:
- Compute an encryption of $a \mid b$:
- Compute an encryption of $a == 1$:
- Compute an encryption of $c \mid (a \,\&\, b) \mid (!c \,\&\, d)$:

The Fully Homomorphic Encryption Conjecture (2/3)

Numbers and circuit logic

- How to compute ($\texttt{var} >= \texttt{somenumber}$) (return 1 if so or 0 if not) for α an encryption of $\texttt{age}$?
- For any test, there is a polynomial of degree $\texttt{MAXVALUE}$ that is exactly as we want (by interpolation)
- The problem is much simpler if we have ciphertexts $(\alpha_0, \ldots, \alpha_{n-1})$ of the bits $(a_0, \ldots, a_{n-1})$ of $\texttt{var}$

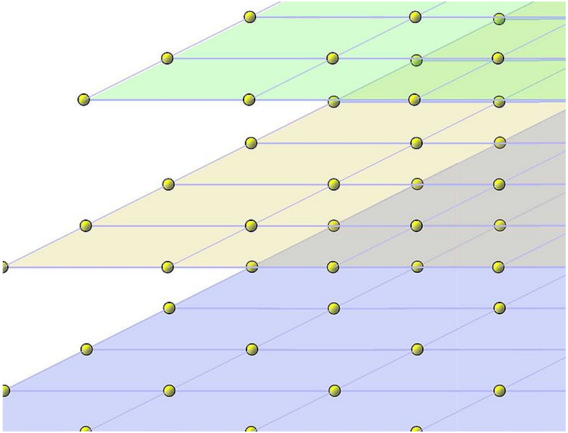
Train yourself! (comparisons)

Suppose $(\alpha_0, \ldots, \alpha_6)$ are ciphertexts of the bits $(a_0, \ldots, a_6)$ of a variable $\texttt{age} = \sum_{i \in [0..6]} a_i 2^i$ ($\texttt{age}$ is in [0..127])

- (EXAMPLE) Give a logical circuit for ($\texttt{age} \neq 0$):
- Give a logical circuit for ($\texttt{age} >= 64$):
- Give a logical circuit for ($\texttt{age} == 32$):
- Give a logical circuit for ($\texttt{age} >= 18$):

Interlude: Lattices

$$\begin{pmatrix} z_1 & z_2 & z_3 \end{pmatrix} \times \begin{pmatrix} 1 & 1 & 3 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{pmatrix}$$

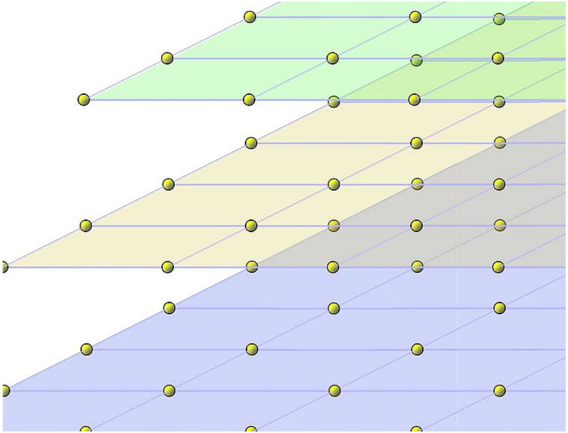


Lattice-based Encryption

Strong security proofs
Fully Homomorphic constructions

Interlude: Lattices

$$\begin{pmatrix} z_1 & z_2 & z_3 \end{pmatrix} \times \begin{pmatrix} 1 & 1 & 3 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{pmatrix}$$

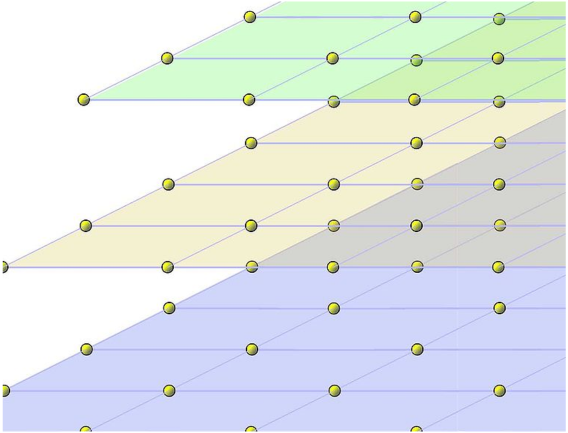


Usual Sizes

Vectors of 100 to 1000 coordinates
With scalars modulo an integer, typically of 32 bits.

Interlude: Lattices

$$\begin{pmatrix} z_1 & z_2 & z_3 \end{pmatrix} \times \begin{pmatrix} 1 & 1 & 3 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{pmatrix}$$



High level description

Ciphertext: Vector close to a random point in the lattice
Plaintext: Correction needed to get back to the lattice (vector with smaller coordinates)

Somewhat Fully Homomorphic Encryption

What

There are secure cryptosystems with operations $\mathtt{MUL}$ and $\mathtt{ADD}$ such that:

- $\mathtt{MUL}(\mathtt{Enc}(\mathtt{x}_1, \mathtt{r}_1), \mathtt{Enc}(\mathtt{x}_2, \mathtt{r}_2)) = \mathtt{Enc}(\mathtt{x}_1 \mathtt{x}_2, \mathtt{r}_3)$
- $\mathtt{ADD}(\mathtt{Enc}(\mathtt{x}_1, \mathtt{r}_1), \mathtt{Enc}(\mathtt{x}_2, \mathtt{r}_2)) = \mathtt{Enc}(\mathtt{x}_1 + \mathtt{x}_2, \mathtt{r}_4)$

with $r_3 = r_1 + r_2$ and $r_4 \sim r_1 * r_2$. Think of polynomial multiplication and addition modulo an integer and modulo $X^n + 1$.

The growing noise issue

- Initial noise r_1, r_2 bounded by a small integer δ
- If it grows above a limit Δ decryption may be incorrect

$\rightarrow$ We have a computational budget
  - Logarithmic in the multiplicative depth
  - Relinearization/key-switching (out of scope) trick can increase it to linear

Easy/hard examples

- Hard: $f(x) = x^d \; mod \; N$, circuits with 64-bit arithmetic mixed with bit operations
- Easy: $f(x) = 480 + 120x - 480x^3$, `(age>45)&((weight>100)&(sex=male)|(weight>70)&(sex=female))`

Cryptonets: Paper ID Card

Reference

- Title: CryptoNets: Applying Neural Networks to Encrypted Datawith High Throughput and Accuracy
- Authors:
  - Nathan Downlin, Princeton
  - Ran Gilad-Bachrach, Microsoft
  - Kim Laine, Microsoft
  - Kristin Lauter, Microsoft
  - Michael Naehrig, Microsoft
  - John Wernsing, Microsoft
- Publication: ICML 2016


CryptoNets: Applying Neural Networks to Encrypted Data
with High Throughput and Accuracy

Abstract

1. Introduction

Contents

- Neural network using ADD/MUL operations
- Takes as input MNIST digits encrypted pixel by pixel
- Returns an encrypted set of ten outputs values
- The owner of data decrypts them, the max value is the guessed digit

Cryptonets: Paper ID Card

Reference

- Title: CryptoNets: Applying Neural Networks to Encrypted Datawith High Throughput and Accuracy
- Authors:
  - ▶ Nathan Downlin, Princeton
  - ▶ Ran Gilad-Bachrach, Microsoft
  - ▶ Kim Laine, Microsoft
  - ▶ Kristin Lauter, Microsoft
  - ▶ Michael Naehrig, Microsoft
  - ▶ John Wernsing, Microsoft
- Publication: ICML 2016


CryptoNets: Applying Neural Networks to Encrypted Data with High Throughput and Accuracy

Nathan Dowlin
Department of Mathematics, Princeton University

Ran Gilad-Bachrach
Kim Laine
Kristin Lauter
Michael Naehrig
John Wernsing
Microsoft Research, Redmond

Abstract

1. Introduction

Advantages

- Completely private
  - ▶ The cloud knows nothing about the input or the guess
  - ▶ The user knows nothing about the NN
- 99% precision
- 59k predictions per hour

Limitations

- Restricted to the inference stage
  - ▶ Model trained over unencrypted data
  - ▶ Simplified neurons (polynomials and no floating point)
  - ▶ No sigmoids
  - → Less precision and less complex tasks
- Important latency for applying the network: 250s
- Owner sends 400MB of encrypted data
- Malicious model not really taken into account

Hands-on: Pyfhel

Tutorials

Follow the Pyfhel tutorials: `https://pyfhel.readthedocs.io` → Tutorials

- Follow the HelloWorld tutorial (you should understand almost everything)
- Follow the MultDepth and Relinearization (focus on learning how to relinearize)

Arithmetic tasks

Create an encryption instance as in the tutorials and

- Encrypt/Decrypt and print an integer
- Encrypt/Add/Decrypt two integers
- Encrypt a few integers, compute a linear combination (e.g. neuron) and Decrypt
- Encrypt an float x and compute $3x^2 - 2x^3$ (e.g. sigmoid neuron between 0 and 1) and Decrypt

Circuit tasks

Set $p = 2$ and define a half adder, a full adder and an 8 bit adder
(See `https://en.wikipedia.org/wiki/Adder_(electronics)` if you do not know about adders)
Use relinearization after each multiplication

Get the real stuff

Go check and try `https://github.com/microsoft/CryptoNets`