ZAMA

FULLY HOMOMORPHIC ENCRYPTION

End-to-End Encryption for Everyone

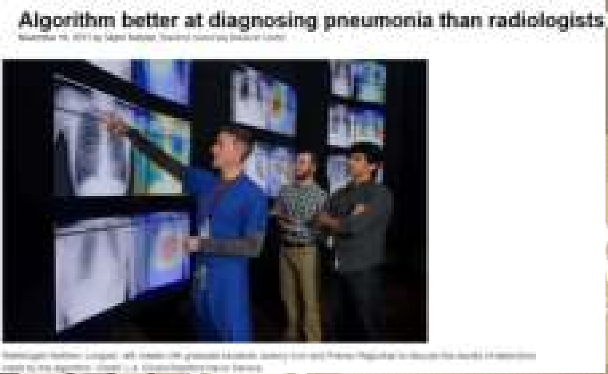
C&ESAR 2022 · Nov. 15–16, 2022

Marc Joye

People shouldn't care about privacy

THE MACHINE LEARNING REVOLUTION

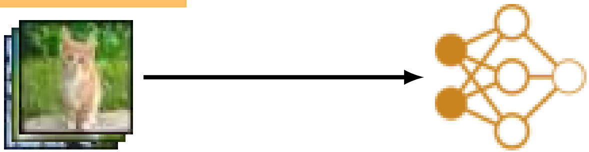

The newest AlphaGo mastered the game with no human input


Algorithm better at diagnosing pneumonia than radiologists

MACHINE LEARNING IN A NUTSHELL

Example: Image classifier

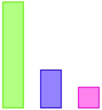
Training phase

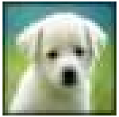


MACHINE LEARNING IN A NUTSHELL

Example: Image classifier

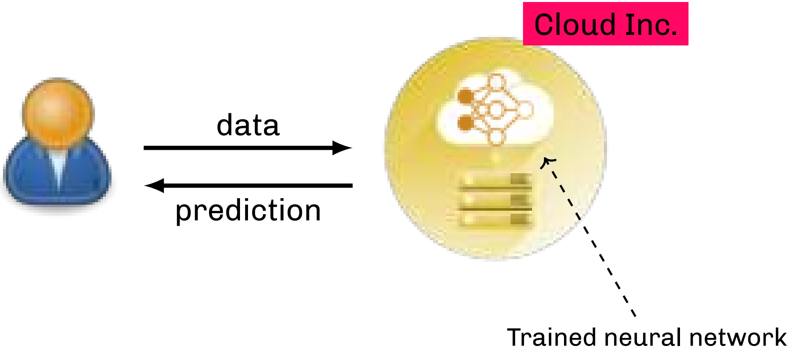
Training phase



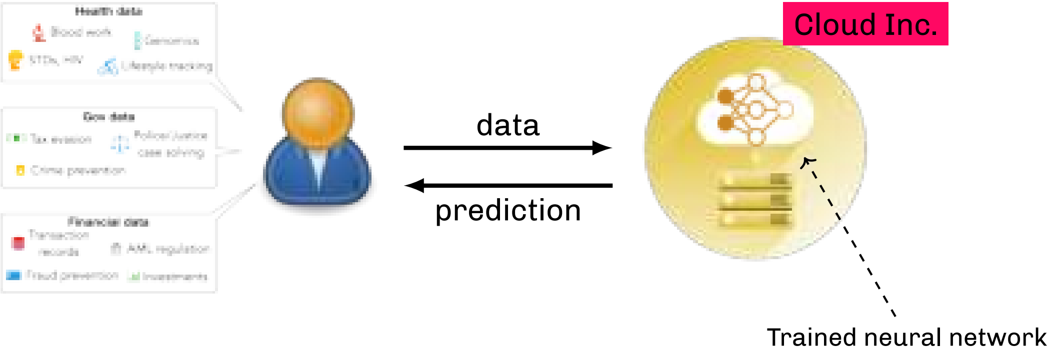


Inference phase

THE ELEPHANT IN THE ROOM



THE ELEPHANT IN THE ROOM



MACHINE LEARNING EXPOSES PERSONAL DATA

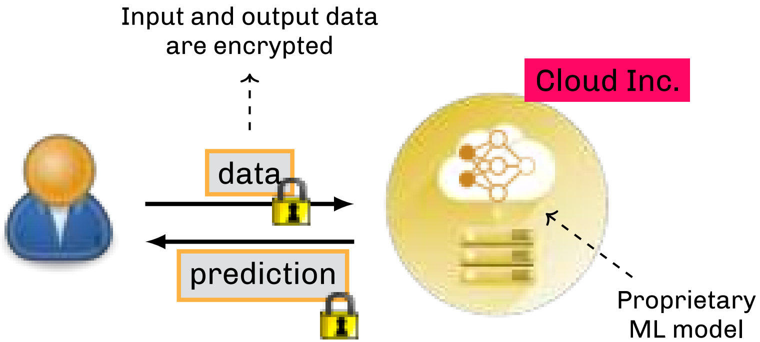
What Cryptography Can Do?

PRIVACY-PRESERVING TECHNOLOGIES

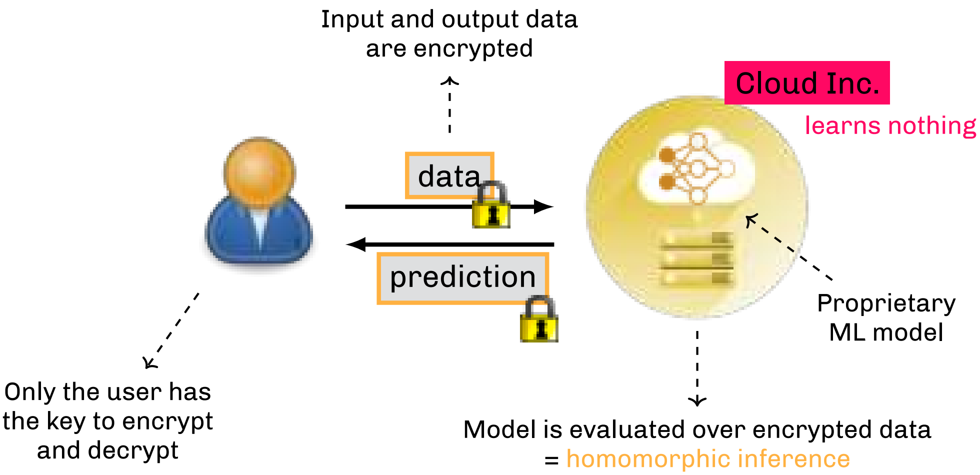
Same services without ever seeing the data!

- Multi-party computation
- Fully homomorphic encryption
- Edge computing
- ...

EMPOWERING MACHINE LEARNING WITH FHE



EMPOWERING MACHINE LEARNING WITH FHE



INTERNET SHOULD BE ENCRYPTED END-TO-END

People won't care about privacy anymore, not because it doesn't matter but because it will be guaranteed by design!

OUTLINE

Fully Homomorphic Encryption

Gentry's Recryption

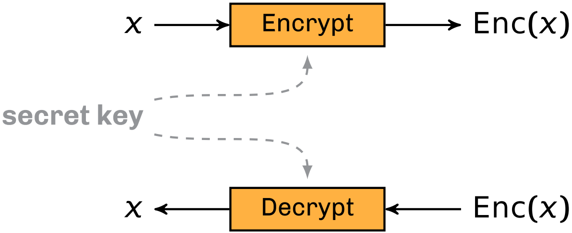
Programmable Bootstrapping

Functional Circuits

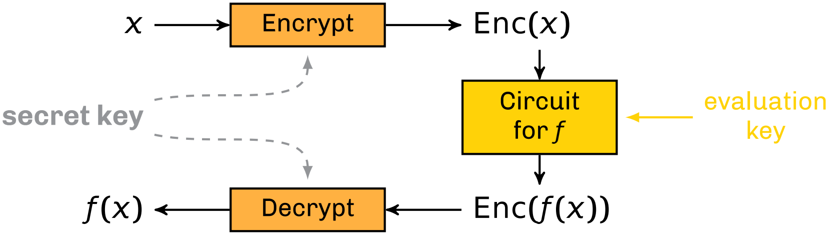
Numerical Experiments

Concluding Remarks

WHAT IS FULLY HOMOMORPHIC ENCRYPTION?



WHAT IS FULLY HOMOMORPHIC ENCRYPTION?



Remark: Any private-key FHE scheme can easily be turned into a public-key FHE scheme

FIRST GENERATION FHE (2009)

PERFORMANCE

$x, y \in \{0, 1\}$

$x, y \in \{0, 1\}$	$\mathrm{Enc}(x), \mathrm{Enc}(y) \rightsquigarrow \mathrm{Enc}(x \oplus y)$	pretty fast
	$\mathrm{Enc}(x), \mathrm{Enc}(y) \rightsquigarrow \mathrm{Enc}(x \wedge y)$	super slow

$\oplus$ and $\wedge$ = all operations

FIRST GENERATION FHE (2009)

PERFORMANCE

$x, y \in \{0, 1\}$

$\mathrm{Enc}(x), \mathrm{Enc}(y) \rightsquigarrow \mathrm{Enc}(x \oplus y)$ pretty fast

$\mathrm{Enc}(x), \mathrm{Enc}(y) \rightsquigarrow \mathrm{Enc}(x \wedge y)$ super slow

$\oplus$ and $\wedge$ = all operations

NOISE PROPAGATION

$x, y \in \{0, 1\}$

$\mathrm{Enc}(x), \mathrm{Enc}(y) \rightsquigarrow \mathrm{Enc}(x \oplus y)$ noise size $\sim$ the same

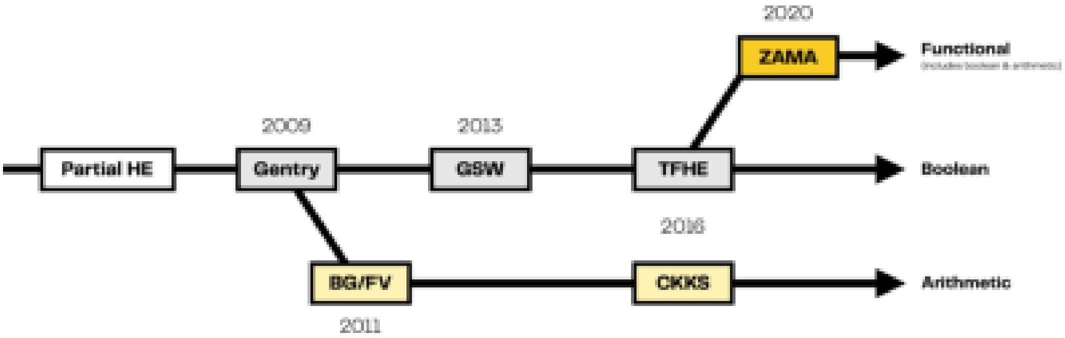
$\mathrm{Enc}(x), \mathrm{Enc}(y) \rightsquigarrow \mathrm{Enc}(x \wedge y)$ noise size doubles

If noise exceeds a threshold, the ciphertext looses “decryptability”

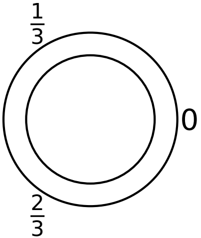
$\implies$ One must resort to bootstrapping, a very slow noise-cleaning operation

NEXT GENERATIONS

Two branches: leveled FHE and bootstrapped FHE



TORUS FHE a.k.a. TFHE

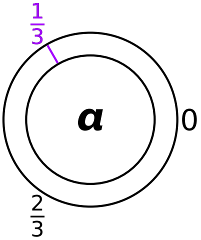


secret key: $\boldsymbol{s} \in \mathbb{B}^n$

ENCRYPTION

DECRYPTION

TORUS FHE a.k.a. TFHE



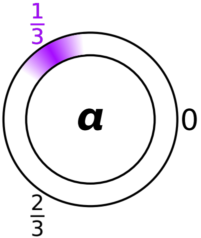
secret key: $\boldsymbol{s} \in \mathbb{B}^n$

ENCRYPTION

1 $\boldsymbol{a} \xleftarrow{\$} \mathbb{T}^n$ (mask)

DECRYPTION

TORUS FHE a.k.a. TFHE



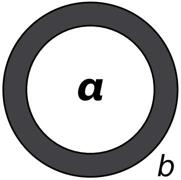
secret key: $s \in \mathbb{B}^n$

ENCRYPTION

1. $a \xleftarrow{\$} \mathbb{T}^n$ (mask)
2. $\mu^* := \mu + e$ with $e \leftarrow \mathcal{N}(0, \sigma^2)$

DECRYPTION

TORUS FHE a.k.a. TFHE



secret key: $\boldsymbol{s} \in \mathbb{B}^n$

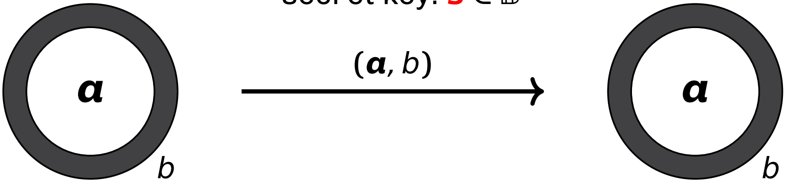
ENCRYPTION

1. $\boldsymbol{a} \xleftarrow{\$} \mathbb{T}^n$ (mask)
2. $\mu^* := \mu + e$ with $e \leftarrow \mathcal{N}(0, \sigma^2)$
3. $b \leftarrow \mu^* + \langle \boldsymbol{s}, \boldsymbol{a} \rangle$ (body)

DECRYPTION

TORUS FHE a.k.a. TFHE

secret key: $\boldsymbol{s} \in \mathbb{B}^n$

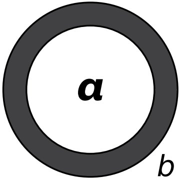


ENCRYPTION

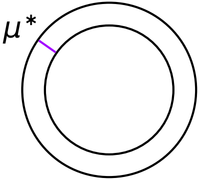
1. $\boldsymbol{a} \xleftarrow{\$} \mathbb{T}^n$ (mask)
2. $\mu^* := \mu + e$ with $e \leftarrow \mathcal{N}(0, \sigma^2)$
3. $b \leftarrow \mu^* + \langle \boldsymbol{s}, \boldsymbol{a} \rangle$ (body)

DECRYPTION

TORUS FHE a.k.a. TFHE



secret key: $\boldsymbol{s} \in \mathbb{B}^n$



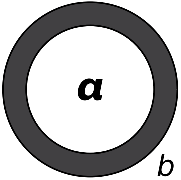
ENCRYPTION

1. $\boldsymbol{a} \xleftarrow{\$} \mathbb{T}^n$ (mask)
2. $\mu^* := \mu + e$ with $e \leftarrow \mathcal{N}(0, \sigma^2)$
3. $b \leftarrow \mu^* + \langle \boldsymbol{s}, \boldsymbol{a} \rangle$ (body)

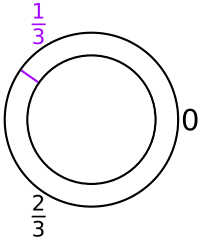
DECRYPTION

1. $\mu^* \leftarrow b - \langle \boldsymbol{s}, \boldsymbol{a} \rangle$

TORUS FHE a.k.a. TFHE



secret key: $\boldsymbol{s} \in \mathbb{B}^n$



ENCRYPTION

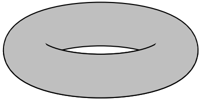
1. $\boldsymbol{a} \xleftarrow{\$} \mathbb{T}^n$ (mask)
2. $\mu^* := \mu + e$ with $e \leftarrow \mathcal{N}(0, \sigma^2)$
3. $b \leftarrow \mu^* + \langle \boldsymbol{s}, \boldsymbol{a} \rangle$ (body)

DECRYPTION

1. $\mu^* \leftarrow b - \langle \boldsymbol{s}, \boldsymbol{a} \rangle$
2. round μ^* to the closest value in $\mathcal{P}$ (plaintext space)

IN PRACTICE...

$$\mathbb{T} = \mathbb{R}/\mathbb{Z} = \{\text{real numbers modulo } 1\}$$



⇝ subset $\mathbb{T}_q := \frac{1}{q}\mathbb{Z}/\mathbb{Z}$

with representatives $\{0, \frac{1}{q}, \ldots, \frac{q-1}{q}\}$

IN THEORY

- $t \in \mathbb{T}$
  $= \sum_{i=1}^{\infty} t_i \, 2^{-i}$
  $= 0.t_1 t_2 t_3 t_4 \ldots$

FINITE PRECISION (ℓ BITS)

- $t = \sum_{i=1}^{\ell} t_i \, 2^{-i}$
  $= \dfrac{\sum_{i=0}^{\ell-1} t_{\ell-i} \, 2^i}{q}$ where $q = 2^\ell$

MESSAGE ENCODING & DECODING

ENCODING FUNCTION

Cleartexts $\mathcal{M} = \{0, \ldots, p-1\}$

Plaintexts $\mathcal{P} = \frac{1}{q}\mathbb{Z}/\mathbb{Z} = \left\{0, \frac{1}{q}, \ldots, \frac{q-1}{q}\right\}$

$$\text{Encode}: \mathcal{M} \to \mathcal{P}, m \mapsto \mu$$

where $\text{Encode}(m) = \dfrac{\lceil q\frac{m}{p} \rfloor \pmod q}{q}$

MESSAGE ENCODING & DECODING

ENCODING FUNCTION

Cleartexts $\mathcal{M} = \{0, \ldots, p-1\}$

Plaintexts $\mathcal{P} = \frac{1}{q}\mathbb{Z}/\mathbb{Z} = \{0, \frac{1}{q}, \ldots, \frac{q-1}{q}\}$

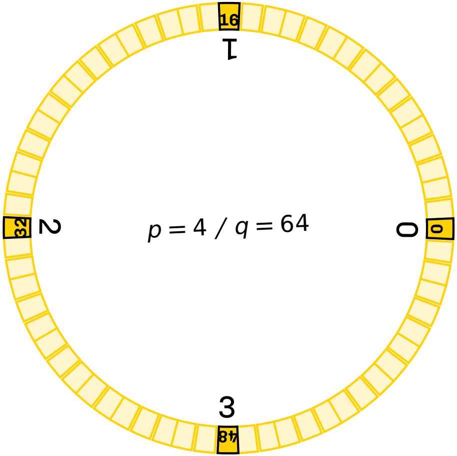
$$\text{Encode}: \mathcal{M} \to \mathcal{P}, m \mapsto \mu$$

where $\text{Encode}(m) = \dfrac{\lceil q\frac{m}{p} \rfloor \pmod q}{q}$

EXAMPLE

- Message space: $\mathcal{M} = \{0, 1, 2, 3\}$ $\rightsquigarrow p = 4$
- Ciphertext modulus: 6 bits $\rightsquigarrow q = 64$

MESSAGE ENCODING & DECODING



ENCODING FUNCTION

Cleartexts $\mathcal{M} = \{0, \ldots, p-1\}$

Plaintexts $\mathcal{P} = \frac{1}{q}\mathbb{Z}/\mathbb{Z} = \{0, \frac{1}{q}, \ldots, \frac{q-1}{q}\}$

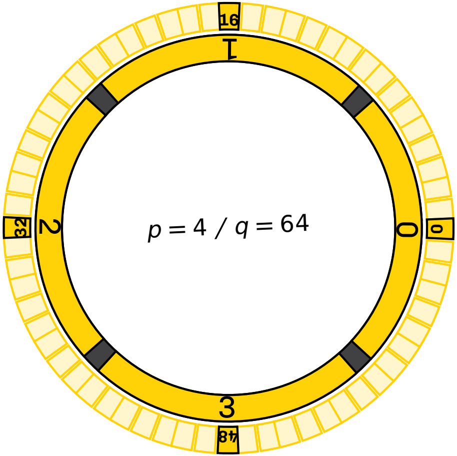
$$\text{Encode}: \mathcal{M} \to \mathcal{P}, m \mapsto \mu$$

where $\text{Encode}(m) = \dfrac{\lceil q\frac{m}{p} \rfloor \pmod{q}}{q}$

EXAMPLE

- Message space: $\mathcal{M} = \{0, 1, 2, 3\}$ $\rightsquigarrow p = 4$
- Ciphertext modulus: 6 bits $\rightsquigarrow q = 64$

MESSAGE ENCODING & DECODING



DECODING FUNCTION

Cleartexts $\mathcal{M} = \{0, \ldots, p-1\}$

Plaintexts $\mathcal{P} = \frac{1}{q}\mathbb{Z}/\mathbb{Z} = \{0, \frac{1}{q}, \ldots, \frac{q-1}{q}\}$

$$\text{Decode}: \mathcal{P} \to \mathcal{M}, \mu^* \mapsto m$$

where $\text{Decode}(\mu^*) = \lceil p\,\mu^* \rfloor \pmod{p}$

Ciphertexts will decrypt correctly provided that noise $|e| < \frac{q}{2p}$

THE EFFECT OF NOISE

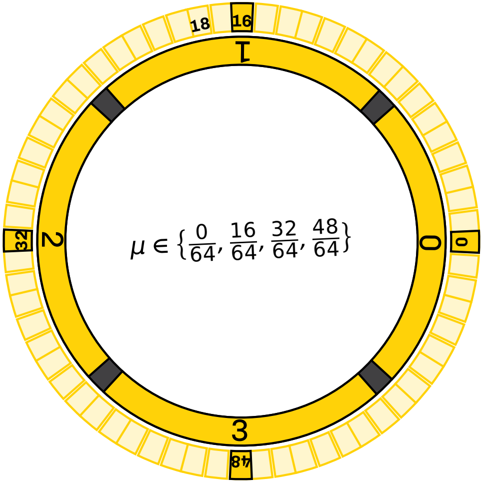
TLWE ENCRYPTION

$$\mathrm{Enc}(\mu \in \mathbb{T}_q) = (\boldsymbol{a}, b)$$

1. $\boldsymbol{a} \xleftarrow{\$} \mathbb{T}_q^n$
2. $\mu^* := \mu + e$ with $e \leftarrow \mathcal{N}(0, \sigma^2)$
3. $b \leftarrow \mu^* + \langle \boldsymbol{s}, \boldsymbol{a} \rangle$

TLWE DECRYPTION

1. recover μ^* as $b - \langle \boldsymbol{s}, \boldsymbol{a} \rangle$
2. round to get μ



THE EFFECT OF NOISE

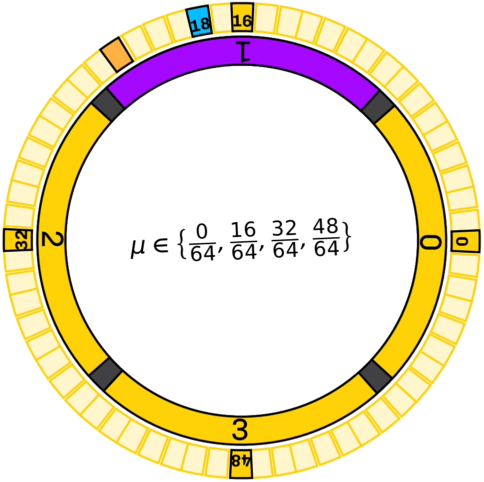
TLWE ENCRYPTION

$$\mathrm{Enc}(\mu \in \mathbb{T}_q) = (\boldsymbol{a}, b)$$

1. $\boldsymbol{a} \xleftarrow{\$} \mathbb{T}_q^n$
2. $\mu^* := \mu + e$ with $e \leftarrow \mathcal{N}(0, \sigma^2)$
3. $b \leftarrow \mu^* + \langle \boldsymbol{s}, \boldsymbol{a} \rangle$

TLWE DECRYPTION

1. recover μ^* as $b - \langle \boldsymbol{s}, \boldsymbol{a} \rangle$
2. round to get μ



THE EFFECT OF NOISE

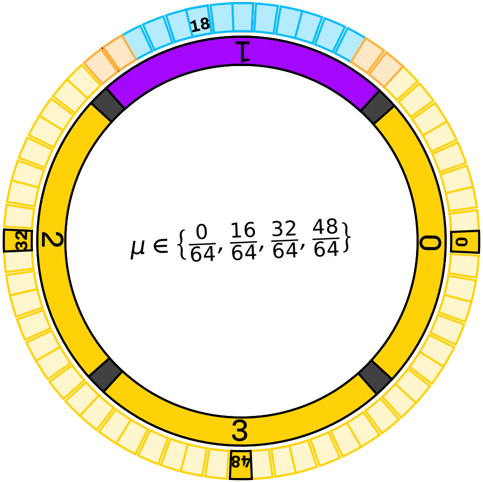
TLWE ENCRYPTION

$$\mathrm{Enc}(\mu \in \mathbb{T}_q) = (\boldsymbol{a}, b)$$

1. $\boldsymbol{a} \xleftarrow{\$} \mathbb{T}_q^n$
2. $\mu^* := \mu + e$ with $e \leftarrow \mathcal{N}(0, \sigma^2)$
3. $b \leftarrow \mu^* + \langle \boldsymbol{s}, \boldsymbol{a} \rangle$

TLWE DECRYPTION

1. recover μ^* as $b - \langle \boldsymbol{s}, \boldsymbol{a} \rangle$
2. round to get μ



OUTLINE

Fully Homomorphic Encryption

Gentry's Recryption

Programmable Bootstrapping

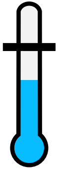
Functional Circuits

Numerical Experiments

Concluding Remarks

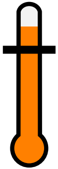
CONTROLLING THE NOISE

Enc(x)



✔ valid ciphertext

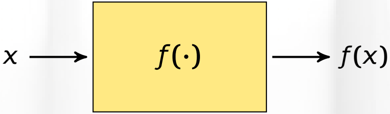
Enc(x)



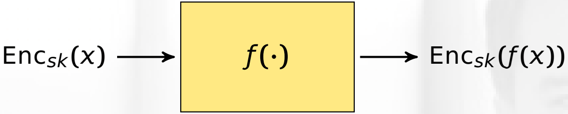
✖ incorrect decryption

Noise accumulates over time

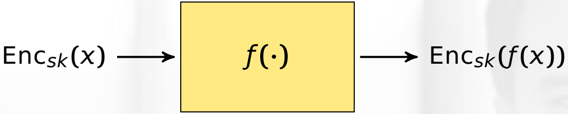
GENTRY'S RECRYPTION



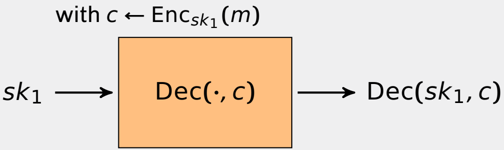
GENTRY'S RECRYPTION



GENTRY'S RECRYPTION



APPLICATION: RECRYPTION



GENTRY'S RECRYPTION

$$\mathrm{Enc}_{sk}(x) \longrightarrow \boxed{f(\cdot)} \longrightarrow \mathrm{Enc}_{sk}(f(x))$$

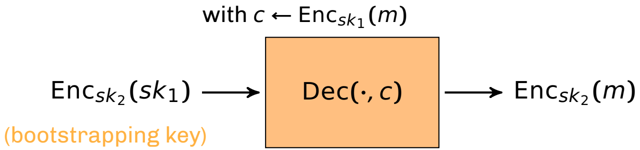
APPLICATION: RECRYPTION

with $c \leftarrow \mathrm{Enc}_{sk_1}(m)$

$$\mathrm{Enc}_{sk_2}(sk_1) \longrightarrow \boxed{\mathrm{Dec}(\cdot, c)} \longrightarrow \mathrm{Enc}_{sk_2}(\mathrm{Dec}(sk_1, c))$$

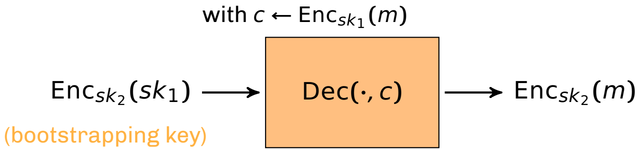
$$= \mathrm{Enc}_{sk_2}(m)$$

PROBLEM TO SOLVE



- Only known way to bootstrap is Gentry's recryption technique

PROBLEM TO SOLVE

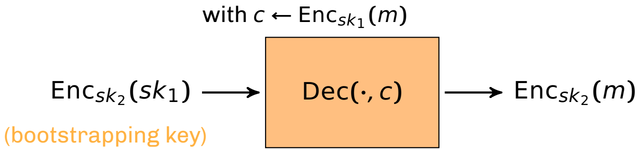


- Only known way to bootstrap is Gentry's recryption technique

TLWE DECRYPTION

1. $\mu^* \leftarrow b - \langle \boldsymbol{s}, \boldsymbol{a} \rangle$
2. round μ^*

PROBLEM TO SOLVE

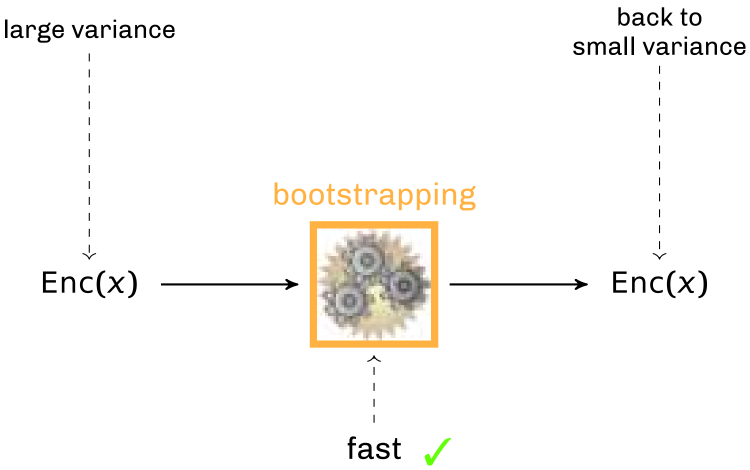


- Only known way to bootstrap is Gentry's recryption technique
- **How to round over encrypted data?**

TLWE DECRYPTION

1. $\mu^* \leftarrow b - \langle \boldsymbol{s}, \boldsymbol{a} \rangle$
2. round μ^*

TFHE BOOTSTRAPPING



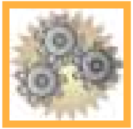
OUTLINE

PROGRAMMABLE BOOTSTRAPPING

large variance

back to
small variance

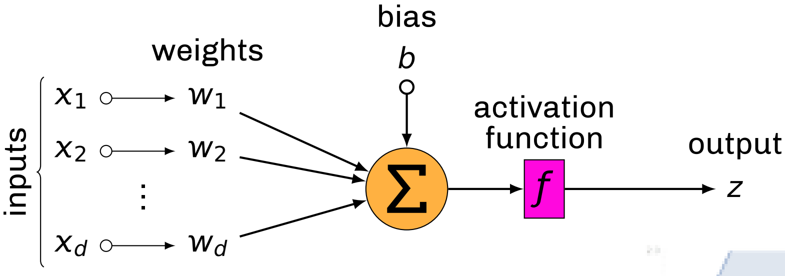
Enc(x) → **programmable** bootstrapping → Enc($f(x)$)



$f(\cdot)$

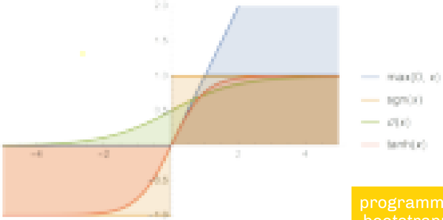
- function f is programmed during the bootstrapping
- essentially no extra cost

ARTIFICIAL NEURON



$$y = \sum_{i=1}^{d} w_i x_i + b$$

$$z = f(y)$$



programmable bootstrapping

PERFORMANCE

Programmable bootstrapping in milliseconds*

	$N = 1024$		$N = 2048$		$N = 4096$	
# bits	32	64	32	64	32	64
$n = 630$	15.49	18.08	33.28	39.54	73.22	84.01
$n = 800$	19.23	22.98	42.33	50.53	93.12	107.3
$n = 1024$	24.54	29.16	54.14	64.18	117.9	135.2

* 2.6 GHz 6-Core Intel® Core™ i7 processor

OUTLINE

Fully Homomorphic Encryption

Gentry's Recryption

Programmable Bootstrapping

Functional Circuits

Numerical Experiments

Concluding Remarks

PROGRAMMABLE BOOTSTRAPPING IS POWERFUL

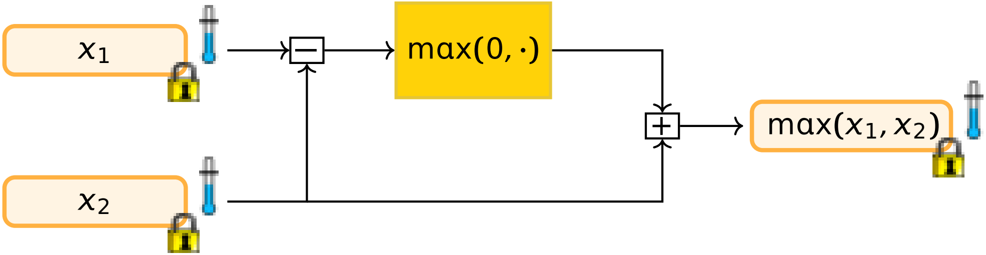
COMPUTING A MAXIMUM: $\max(x_1, x_2, \ldots, x_n)$

- $\max(x_1, x_2) = \max(0, x_1 - x_2) + x_2$
- $\max(x_1, x_2, x_3) = \max\big(\max(x_1, x_2), x_3\big)$

PROGRAMMABLE BOOTSTRAPPING IS POWERFUL

COMPUTING A MAXIMUM: $\max(x_1, x_2, \ldots, x_n)$

- $\max(x_1, x_2) = \max(0, x_1 - x_2) + x_2$
- $\max(x_1, x_2, x_3) = \max\big(\max(x_1, x_2), x_3\big)$



ALL YOU NEED: ADDITIONS AND PBS's

Kolmogorov Superposition Theorem (KST) 1957

$$f(x_1, \ldots, x_n) = \sum_i g_i\left(\sum_j f_{i,j}(x_j)\right)$$

univariate

Ridge decomposition or approximation

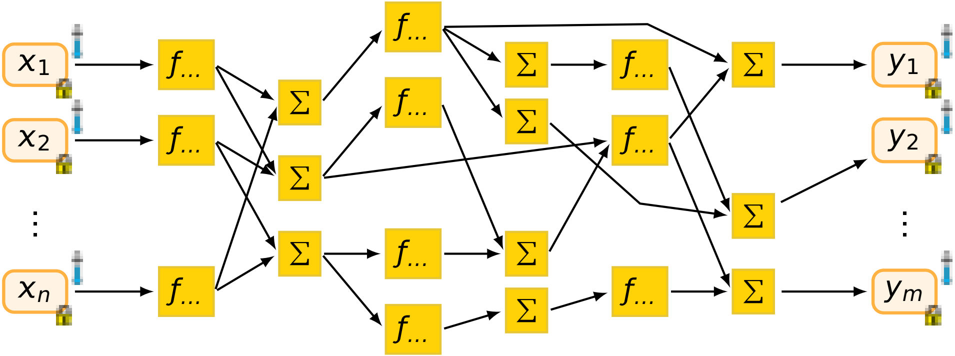
$$f(x_1, \ldots, x_n) \approx \sum_i g_i\left(\sum_j a_{i,j}\, x_j\right)$$

univariate

$a_{i,j} \in \mathbb{Z}$

A NEW COMPUTATIONAL PARADIGM

Circuit of univariate functions



Graph mixing univariate functions and linear combinations

OUTLINE

Fully Homomorphic Encryption

Gentry's Recryption

Programmable Bootstrapping

Functional Circuits

Numerical Experiments

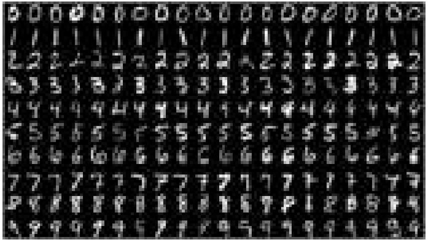
Concluding Remarks

Let’s be Concrete

https://github.com/zama-ai

NUMERICAL EXPERIMENTS

- MNIST dataset



- Three neural networks:
  - NN-x where x is the number of layers with $x \in \{20, 50, 100\}$
  - networks all include dense and convolution layers with activation functions
  - every hidden layer possesses at least 92 active neurons

NUMERICAL EXPERIMENTS

	In the clear	Encrypted
NN-20	0.17 ms	115.52 s
NN-50	0.20 ms	233.55 s
NN-100	0.33 ms	481.61 s

* 2.6 GHz 6-Core Intel® Core™ i7 processor

OUTLINE

Fully Homomorphic Encryption

Gentry's Recryption

Programmable Bootstrapping

Functional Circuits

Numerical Experiments

Concluding Remarks

SUMMARY

- Programmable bootstrapping is a powerful tool
  - enables evaluation of any function
  - runs relatively fast
  - accommodates every use-case

- Try out the Concrete library!

SOME PERSPECTIVES

PROVABLE FHE

Generate a proof of correctness for an FHE execution

HW ACCELERATION

Make FHE faster to execute for addressing more use-cases

ZAMA

FULLY HOMOMORPHIC ENCRYPTION

End-to-End Encryption for Everyone

C&ESAR 2022 · Nov. 15–16, 2022

Marc Joye