Secure Multi-party Computation for Privacy-preserving Machine Learning

Alessandro Baccarini, PhD

abaccarini@proton.me
abaccarini.github.io

January 27, 2025

Outline

Motivation for PPML

Multi-party computation and PPML

- General-purpose secure computation
- Application: quantized neural networks

Security considerations and conclusions

AI is **everywhere**...

Apple Intelligence

Amazon Q

Meta AI

Grok

Gemini

ChatGPT

Copilot

Claude

AI is **everywhere**...

Apple Intelligence

Amazon Q

Meta AI

Grok

Gemini

ChatGPT

Copilot

Claude

... but what about "private" AI?

Forbes

FORBES > MONEY > FINTECH

Generative AI Under Attack: Flowbreaking Exploits Trigger Data Leaks

Nizan Geslevich Packin Contributor
I write about financial regulation tech policy and consumer protection

Follow

WIRED

DELL CAMERON ANDREW COUTS SECURITY MAY 25, 2024 6:30 AM

Security News This Week: Microsoft's New Recall AI Tool May Be a 'Privacy Nightmare'

Plus: US surveillance reportedly targets pro-Palestinian protesters, the FBI arrests a man for AI-generated CSAM, and stalkerware targets hotel computers.

Reuters

The privacy paradox with AI

By Gai Sher and Ariela Benchlouch

October 31, 2023 1:15 PM EDT · Updated a year ago

nature

nature > career guide > article

CAREER GUIDE | 04 September 2024

Intellectual property and data privacy: the hidden risks of AI

Generative artificial-intelligence tools have been widely adopted across academia, but users might not be aware of all their inherent risks.

By Amanda Heidt

Forbes

FORBES > INNOVATION > TRANSPORTATION

AI Risks Include Data Poisoning And Model Corruption

Steve Banker Contributor

nation

Business

Businesses warned not to use open AI to prevent data leaks

SUNDAY, DECEMBER 22, 2024

The Register

Slack AI can be tricked into leaking data from private channels via prompt injection

Whack yakety-yak app chaps rapped for security crack

Thomas Claburn Wed 21 Aug 2024 09:23 UTC

The Cyber Express

TRENDING

Google's Gemini AI Exposes User Chats in Search Results: Here's What Happened?

Reports surfaced on social media platforms indicating that certain chat pages linked to Gemini AI had leaked onto the internet.

by Ashish Khaitan — February 22, 2024 Reading Time: 5 mins read

DARKREADING

APPLICATION SECURITY DATA PRIVACY VULNERABILITIES & THREATS CYBER RISK

Privacy Anxiety Pushes Microsoft Recall AI Release Again

The Recall AI tool will be available to Copilot+ PC subscribers in December, and can be used to record images of every interaction on the device for review later. Critics say this introduces major privacy and security concerns along with useful functionality.

Becky Bracken, Senior Editor, Dark Reading
November 1, 2024
5 Min Read

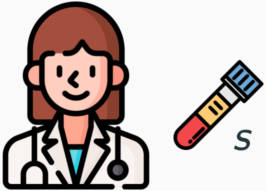
CYBER REPORT

From Gmail to Word, your privacy settings and AI are entering into a new relationship

PUBLISHED WED, JAN 15 2025·10:22 AM EST

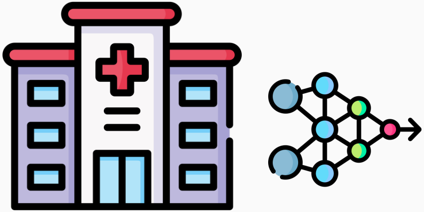
Kevin Williams

Core motivational example: healthcare



s

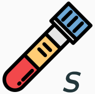
client



model owner M

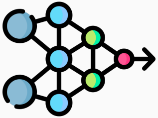
Core motivational example: healthcare

Use your model to classify this sample!



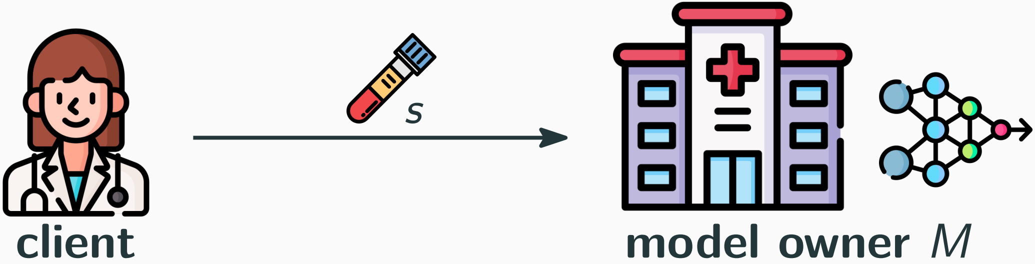
s

client

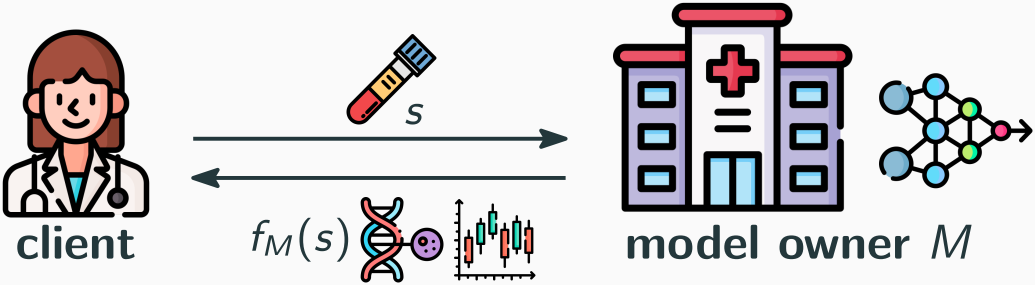


model owner M

Core motivational example: healthcare

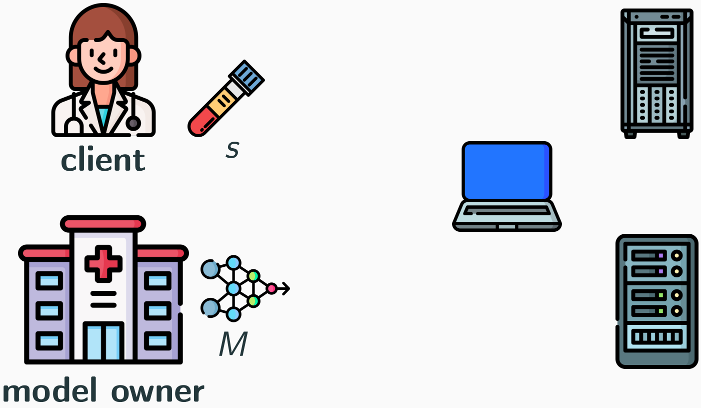


Core motivational example: healthcare

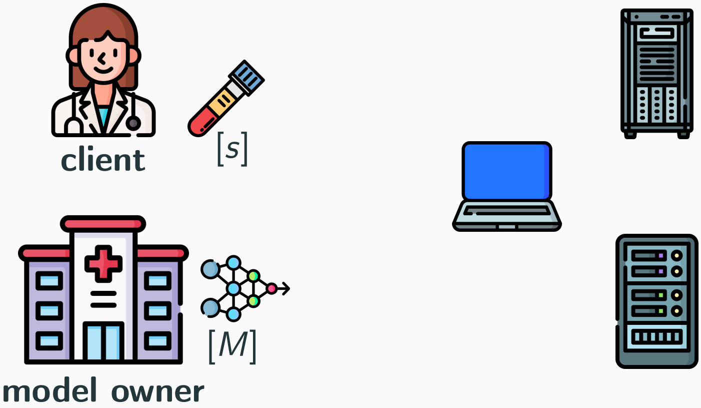


- No privacy for the **client** (data owner)
- No privacy for the **model owner** if roles are reversed
- How can we provide privacy for both parties?

Enter privacy-preserving machine learning (PPML)



Enter privacy-preserving machine learning (PPML)

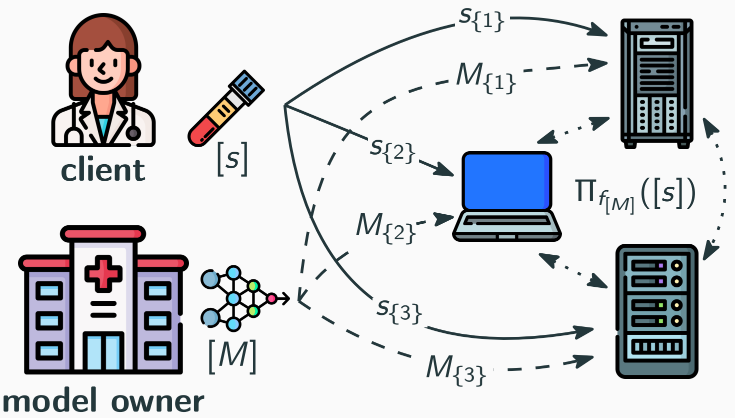


Multi-party computation

Multiple participants **jointly** evaluating an **arbitrary** function on private inputs.

- **No information disclosed other than the output**
- FHE, garbled circuits, **secret sharing**

Enter privacy-preserving machine learning (PPML)

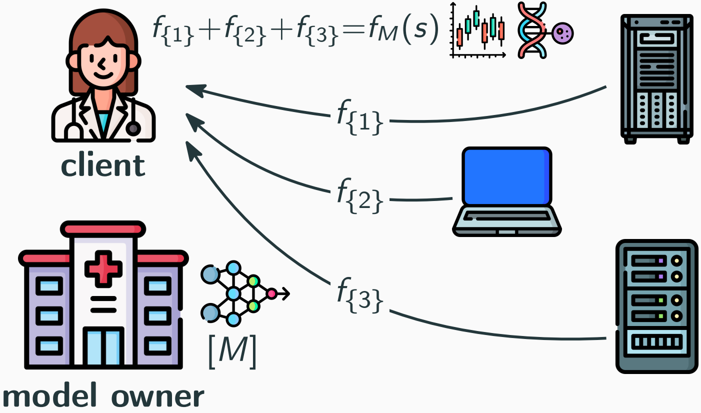


Multi-party computation

Multiple participants **jointly** evaluating an **arbitrary** function on private inputs.

- **No information disclosed other than the output**
- FHE, garbled circuits, **secret sharing**

Enter privacy-preserving machine learning (PPML)

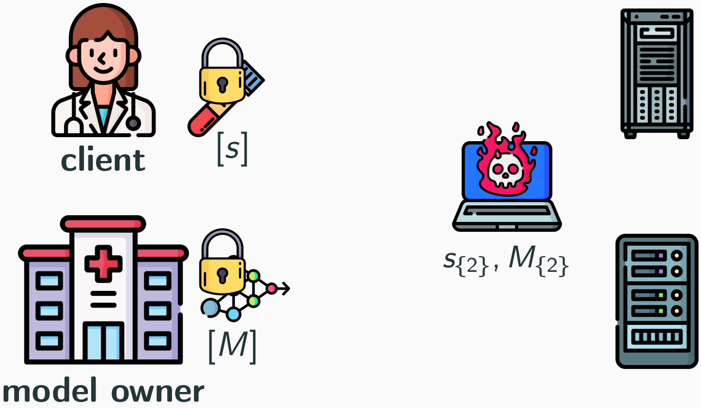


Multi-party computation

Multiple participants **jointly** evaluating an **arbitrary** function on private inputs.

- **No information disclosed other than the output**
- FHE, garbled circuits, **secret sharing**

Enter privacy-preserving machine learning (PPML)



Multi-party computation

Multiple participants **jointly** evaluating an **arbitrary** function on private inputs.

- **No information disclosed other than the output**
- FHE, garbled circuits, **secret sharing**
- (n, t)-threshold scheme
  - $\leq t$ **cannot** recover the secret
- **semi-honest (passive), honest majority**

Secret sharing (SS) techniques

Fields $\mathbb{F}_p$ (Shamir [Sha79])

Rings $\mathbb{Z}_{2^k}$ (Ito et al. [ISN87])

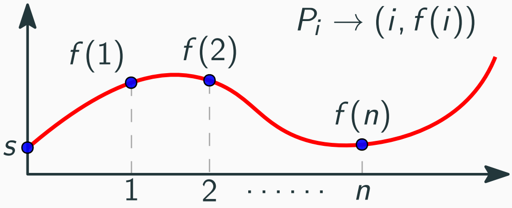
Secret sharing (SS) techniques

Fields $\mathbb{F}_p$ **(Shamir [Sha79])**

- Shares are points on a **polynomial**
- Reconstruction through **interpolation** (requires multiplicative inverses)
- Reliance on **large-number libraries**

$$f(x) = s + a_1 x + \cdots + a_t x^t \pmod p$$

$$P_i \rightarrow (i, f(i))$$



Rings $\mathbb{Z}_{2^k}$ **(Ito et al. [ISN87])**

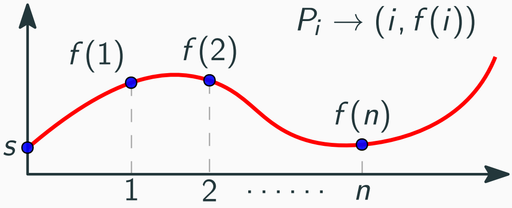
Secret sharing (SS) techniques

Fields $\mathbb{F}_p$ **(Shamir [Sha79])**

- Shares are points on a **polynomial**
- Reconstruction through **interpolation** (requires multiplicative inverses)
- Reliance on **large-number libraries**

$$f(x) = s + a_1 x + \cdots + a_t x^t \pmod{p}$$

$$P_i \rightarrow (i, f(i))$$



Rings $\mathbb{Z}_{2^k}$ **(Ito et al. [ISN87])**

- Each party maintains **replicated** shares
- Compatible with **native CPU instructions**
- Existing works **limited to $n = 3, 4$**

$$s = s_{\{1\}} + s_{\{2\}} + s_{\{3\}} \pmod{2^k}$$

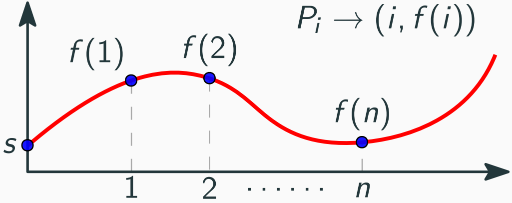
Secret sharing (SS) techniques

Fields $\mathbb{F}_p$ **(Shamir [Sha79])**

- Shares are points on a **polynomial**
- Reconstruction through **interpolation** (requires multiplicative inverses)
- Reliance on **large-number libraries**

$$f(x) = s + a_1 x + \cdots + a_t x^t \pmod{p}$$

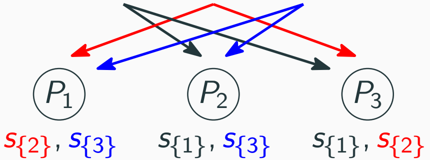
$$P_i \to (i, f(i))$$



Rings $\mathbb{Z}_{2^k}$ **(Ito et al. [ISN87])**

- Each party maintains **replicated** shares
- Compatible with **native CPU instructions**
- Existing works **limited to $n = 3, 4$**

$$s = s_{\{1\}} + s_{\{2\}} + s_{\{3\}} \pmod{2^k}$$

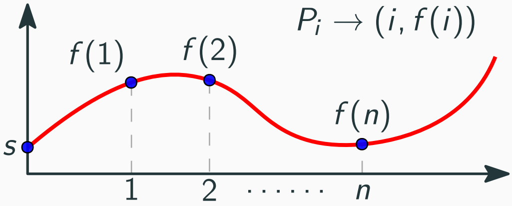


Secret sharing (SS) techniques

Fields $\mathbb{F}_p$ (Shamir [Sha79])

- Shares are points on a **polynomial**
- Reconstruction through **interpolation** (requires multiplicative inverses)
- Reliance on **large-number libraries**

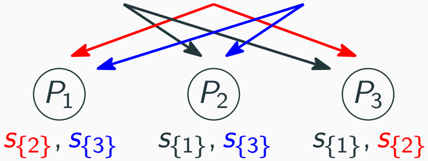
$$f(x) = s + a_1 x + \cdots + a_t x^t \pmod p$$



Rings $\mathbb{Z}_{2^k}$ (Ito et al. [ISN87])

- Each party maintains **replicated** shares
- Compatible with **native CPU instructions**
- Existing works **limited to $n = 3, 4$**

$$s = s_{\{1\}} + s_{\{2\}} + s_{\{3\}} \pmod{2^k}$$



RSS framework for any n [Bac24]

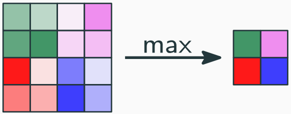
From ML to PPML: **neural networks**

Layer operations

convolution, transformer, ...

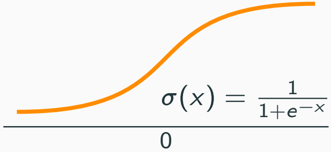
Pooling (optional)

max, average, ...



Activation functions

ReLU, sigmoid, ...



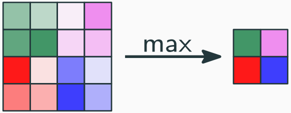
From ML to PPML: **neural networks**

Layer operations

convolution, transformer, . . .

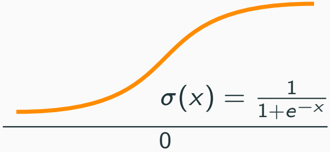
Pooling (optional)

max, average, . . .



Activation functions

ReLU, sigmoid, . . .

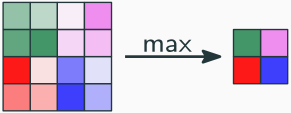


All distill to "simple" operations

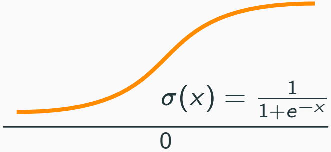
From ML to PPML: **neural networks**

Layer operations
convolution, transformer, . . .

Pooling (optional)
max, average, . . .



Activation functions
ReLU, sigmoid, . . .



All distill to "simple" operations

matrix multiplication

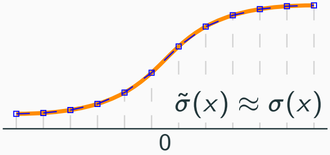
$$y = \sum_i x_i w_i + b$$

comparisons

$$(a \overset{?}{>} b) \rightarrow \mathrm{MSB}(a-b)$$

$$(a \overset{?}{=} b) \rightarrow \mathrm{EQZ}(a-b)$$

approximations



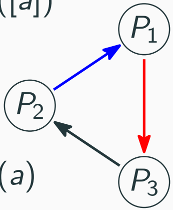
Towards general-purpose secure computation

Building Blocks
reconstruction, mult.,
inputting private values

$$\left.\begin{array}{r} c\cdot[a] \\ [a]+[b] \end{array}\right\} \text{local, “free”}$$

Open($[a]$)

$[a]\cdot[b]$



Input(a)

1 round,
$\mathcal{O}(t)$ comm.

Towards general-purpose secure computation

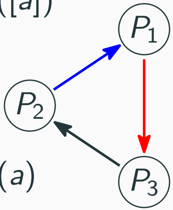
Building Blocks

reconstruction, mult., inputting private values

$\left.\begin{matrix} c\cdot[a] \\ [a]+[b] \end{matrix}\right\}$ local, “free”

$\mathsf{Open}([a])$

$[a]\cdot[b]$



$\mathsf{Input}(a)$

1 round, $\mathcal{O}(t)$ comm.

Composite Operations

share conversion, shared randomness generation, comparisons, shifts, division

$\mathbb{Z}_2 \longrightarrow \mathbb{Z}_{2^k}$

$\mathsf{RandBit}()$ $\mathsf{edaBit}(k)$

$\mathsf{MSB}([a])$ $\mathsf{EQZ}([a])$

$[a/2^m], [a\cdot 2^m]$

$[a]/[b]$

complexity

Poly(log) rounds/ comm. in k, t

Towards general-purpose secure computation

Building Blocks

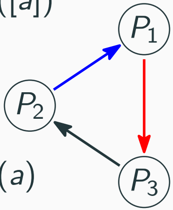
reconstruction, mult., inputting private values

$\left.\begin{array}{r} c\cdot[a] \\ [a]+[b] \end{array}\right\}$ local, "free"

Open($[a]$)

$[a]\cdot[b]$

Input(a)



1 round,
$\mathcal{O}(t)$ comm.

Composite Operations

share conversion, shared randomness generation, comparisons, shifts, division

$\mathbb{Z}_2 \longrightarrow \mathbb{Z}_{2^k}$ [Bac24, §4.2.1]

RandBit() edaBit(k)

MSB($[a]$) EQZ($[a]$)

$[a/2^m], [a\cdot 2^m]$

$[a]/[b]$

complexity

Poly(log) rounds/
comm. in k, t

Towards general-purpose secure computation

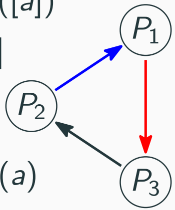
Building Blocks
reconstruction, mult., inputting private values

$$\left.\begin{array}{r} c\cdot[a] \\ [a]+[b] \end{array}\right\}\text{ local, “free”}$$

$\mathrm{Open}([a])$

$[a]\cdot[b]$

$\mathrm{Input}(a)$



Composite Operations
share conversion, shared randomness generation, comparisons, shifts, division

$\mathbb{Z}_2 \longrightarrow \mathbb{Z}_{2^k}$ [Bac24, §4.2.1]

$\mathrm{RandBit}()$ $\mathrm{edaBit}(k)$

$\mathrm{MSB}([a])$ $\mathrm{EQZ}([a])$

$[a/2^m], [a\cdot 2^m]$

$[a]/[b]$

complexity

Floating-point Computation
floating-point arithmetic, function approximation

$[\tilde{a}]<[\tilde{b}]$

$[\tilde{a}]\cdot[\tilde{b}]$ $[\tilde{a}]/[\tilde{b}]$

$[\tilde{a}]+[\tilde{b}]$

$$f(x) \approx \begin{cases} \sum_i (a_i x + b_i) \\ \sum_i \frac{f^{(i)}(0)}{i!} x^i \\ \mathrm{LUT}\ldots \end{cases}$$

1 round, $\mathcal{O}(t)$ comm. → Poly(log) rounds/ comm. in k, t → many, *many* rounds, expensive comm.

Towards general-purpose secure computation

Building Blocks

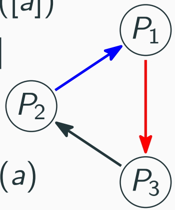
reconstruction, mult., inputting private values

$\left.\begin{matrix} c\cdot[a] \\ [a]+[b] \end{matrix}\right\}$ local, “free”

$\mathrm{Open}([a])$

$[a]\cdot[b]$

$\mathrm{Input}(a)$



Composite Operations

share conversion, shared randomness generation, comparisons, shifts, division

$\mathbb{Z}_2 \longrightarrow \mathbb{Z}_{2^k}$ [Bac24, §4.2.1]

$\mathrm{RandBit}()$ $\mathrm{edaBit}(k)$

$\mathrm{MSB}([a])$ $\mathrm{EQZ}([a])$

$[a/2^m], [a\cdot 2^m]$

$[a]/[b]$

complexity

Floating-point Computation

floating-point arithmetic, function approximation

$[\tilde{a}]<[\tilde{b}]$

$[\tilde{a}]\cdot[\tilde{b}]$ $[\tilde{a}]/[\tilde{b}]$

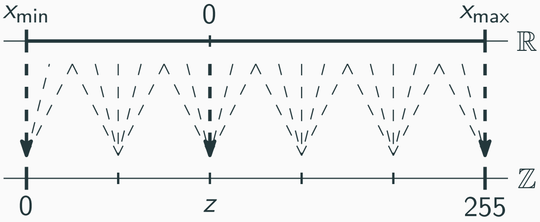
$[\tilde{a}]+[\tilde{b}]$

$$f(x) \approx \begin{cases} \sum_i (a_i x + b_i) \\ \sum_i \frac{f^{(i)}(0)}{i!} x^i \\ \mathrm{LUT}\ldots \end{cases}$$

1 round, $\mathcal{O}(t)$ comm. → Poly(log) rounds/ comm. in k, t → many, *many* rounds, expensive comm.

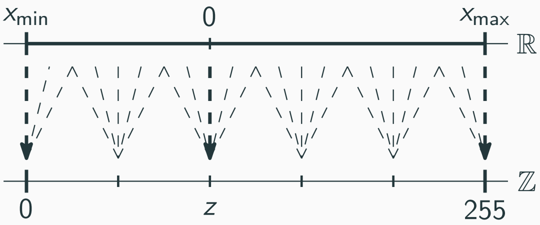
Application: **quantized** neural networks

- Neural network, but _{smaller}
- Values are mapped to the range $[0, 255]$ with *scale* $m \in \mathbb{R}$ and zero point z



Application: **quantized** neural networks

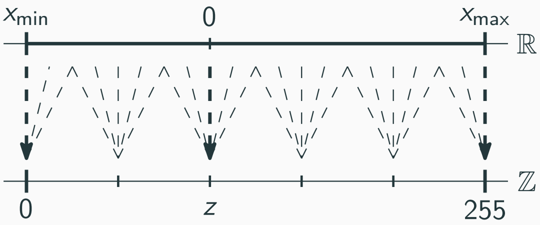
- Neural network, but smaller
- Values are mapped to the range $[0, 255]$ with *scale* $m \in \mathbb{R}$ and zero point z



$$\mathrm{ReLU6}\Big(\underbrace{\sum\nolimits_i x_i w_i + b}_{y}\Big) \implies 0 \leq \underbrace{z_y + \frac{m_x m_w}{m_y} \sum\nolimits_i \big((\bar{x}_i - z_x)(\bar{w}_i - z_w) + \bar{b}\big)}_{\bar{y}} \leq 255$$

- Certain activations (like ReLU6) become **free** by careful selection of m_y, z_y
- Prior works [DEK20]: fixed-point mult., followed by truncation and clamping

Application: **quantized** neural networks



- Neural network, but smaller
- Values are mapped to the range $[0, 255]$ with *scale* $m \in \mathbb{R}$ and zero point z

$$\mathrm{ReLU6}\Big(\underbrace{\textstyle\sum_i x_i w_i + b}_{y}\Big) \implies 0 \le \underbrace{z_y + \frac{m_x m_w}{m_y} \textstyle\sum_i \left((\bar{x}_i - z_x)(\bar{w}_i - z_w) + \bar{b}\right)}_{\bar{y}} \le 255$$

- Certain activations (like ReLU6) become **free** by careful selection of m_y, z_y
- Prior works [DEK20]: fixed-point mult., followed by truncation and clamping

Bottleneck

Uses $k = 72$ to accommodate for the 63-bit truncation.

Solution (Baccarini et al. [BBY23])

Fold scales into clamping operation, and compute a much smaller truncation at the end of each layer.

Application: **quantized** neural networks

Solution (Baccarini et al. [BBY23])

Fold scales into clamping operation, and compute a much smaller truncation at the end of each layer.

$$
\begin{array}{ccccccc}
0 & \leq & z_y & + & \frac{m_x m_w}{m_y} \sum_i \left((\bar{x}_i - z_x)(\bar{w}_i - z_w) + \bar{b} \right) & \leq & 255 \\
 & & \Downarrow & & & & \Downarrow \\
0 & \leq & \frac{m_y z_y}{m_x m_w} & + & \sum_i \left((\bar{x}_i - z_x)(\bar{w}_i - z_w) + \bar{b} \right) & \leq & \frac{255 m_y}{m_x m_w}
\end{array}
$$

Application: **quantized** neural networks

Solution (Baccarini et al. [BBY23])

Fold scales into clamping operation, and compute a much smaller truncation at the end of each layer.

$$\begin{array}{ccccccc} 0 & \leq & z_y & + & \frac{m_x m_w}{m_y} \sum_i \left((\bar{x}_i - z_x)(\bar{w}_i - z_w) + \bar{b} \right) & \leq & 255 \\ & & \Downarrow & & & & \Downarrow \\ 0 & \leq & \frac{m_y z_y}{m_x m_w} & + & \sum_i \left((\bar{x}_i - z_x)(\bar{w}_i - z_w) + \bar{b} \right) & \leq & \frac{255 m_y}{m_x m_w} \end{array}$$

- **Over 2× reduction in ring size!** ($72 \longrightarrow 32$)
- Updated parameters become part of the model, distributed by model owner
- **No impact on accuracy**

Conclusion: the solution to AI privacy concerns?

- Can achieve PPML by applying MPC, yielding robust security guarantees
- Does MPC alleviate *all* AI privacy concerns?

Conclusion: the solution to AI privacy concerns?

- Can achieve PPML by applying MPC, yielding robust security guarantees
- Does MPC alleviate *all* AI privacy concerns?

Adversarial (vanilla) ML

Black box computation (oracle)

- Membership inference attacks
- Model poisoning/inversion, . . .

Typically involves training a “shadow model”

Conclusion: the solution to AI privacy concerns?

- Can achieve PPML by applying MPC, yielding robust security guarantees
- Does MPC alleviate *all* AI privacy concerns?

Adversarial (vanilla) ML

Black box computation (oracle)

- Membership inference attacks
- Model poisoning/inversion, . . .

Typically involves training a “shadow model”

“Other” MPC threats?

- But by definition, MPC is perfectly secure!

Conclusion: the solution to AI privacy concerns?

- Can achieve PPML by applying MPC, yielding robust security guarantees
- Does MPC alleviate *all* AI privacy concerns?

Adversarial (vanilla) ML

Black box computation (oracle)

- Membership inference attacks
- Model poisoning/inversion, ...

Typically involves training a “shadow model”

“Other” MPC threats?

- But by definition, MPC is perfectly secure!

Information disclosure analysis [Bac24, Part II]

Thank you!

Questions?

References

[Ali+13] M. Aliasgari, M. Blanton, Y. Zhang, and A. Steele. "Secure Computation on Floating Point Numbers". In: *Network and Distributed System Security Symposium (NDSS)*. 2013.

[Bac24] A. Baccarini. "New Directions in Secure Multi-Party Computation: Techniques and Information Disclosure Analysis". PhD Thesis. University at Buffalo, 2024.

[BBY23] A. Baccarini, M. Blanton, and C. Yuan. "Multi-Party Replicated Secret Sharing over a Ring with Applications to Privacy-Preserving Machine Learning". In: *Proceedings on Privacy Enhancing Technologies (PoPETs)* 2023.1 (2023), pp. 608–626.

[BGY23] M. Blanton, M. T. Goodrich, and C. Yuan. "Secure and Accurate Summation of Many Floating-Point Numbers". In: *Proceedings on Privacy Enhancing Technologies (PoPETs)* 2023.3 (2023), pp. 432–445.

[Dam+19] I. Damgård, D. Escudero, T. Frederiksen, M. Keller, P. Scholl, and N. Volgushev. "New Primitives for Actively-Secure MPC over Rings with Applications to Private Machine Learning". In: *IEEE Symposium on Security and Privacy (S&P)*. 2019, pp. 1102–1120.

[DEK20] A. Dalskov, D. Escudero, and M. Keller. "Secure Evaluation of Quantized Neural Networks". In: *Proceedings on Privacy Enhancing Technologies (PoPETs)* 2020.4 (2020), pp. 355–375.

[ISN87] M. Ito, A. Saito, and T. Nishizeki. "Secret Sharing Schemes Realizing General Access Structures". In: *IEEE Global Telecommunication Conference (GLOBECOM)*. 1987, pp. 99–102.

[Rat+22] D. Rathee, A. Bhattacharya, R. Sharma, D. Gupta, N. Chandran, and A. Rastogi. "SecFloat: Accurate Floating-Point meets Secure 2-Party Computation". In: *IEEE Symposium on Security and Privacy (S&P)*. 2022, pp. 1553–1553.

[Sha79] A. Shamir. "How to Share a Secret". In: *Communications of the ACM* 22.11 (1979), pp. 612–613.

Binary-to-arithmetic conversion (B2A)

- Often operate on **individual bits** of secrets, requiring conversion from $\mathbb{Z}_2 \to \mathbb{Z}_{2^k}$
- Prior works use **RandBit** [Dam+19], requires temporary computation in $\mathbb{Z}_{2^{k+2}}$
  - E.g., $k = 8$ requires 16-bit integers, **doubling** the communication
- Blanton et al. [BGY23] **eliminated** this requirement for 3-party RSS

Generalization of [BGY23] to any n **[Bac24]**

1. t parties locally XOR a subset of their shares, enter result into computation
2. Remaining $t + 1$ parties "locally reshare" last share (all but one share is nonzero)
3. Compute XOR (in $\mathbb{Z}_{2^k}$) of local XOR(s) and the last share as a tree

- Can use approach to generate shared random bits (RandBit) **without** $\mathbb{Z}_{2^{k+2}}$
- Up to 6.5× faster for 3 parties, 2× faster for 5 parties

Floating-point protocols

– Prior protocols designed for computation on **integer**[1] inputs. . .

– But what about **floating-point**?

$$\tilde{a} = \underbrace{\boxed{\text{sign } s}}_{1}\ \underbrace{\boxed{\text{exponent } e}}_{p}\ \underbrace{\boxed{\text{mantissa (significand) } m}}_{q(+\,1)}$$

$$\tilde{a} = (1 - z) \cdot (1 - 2s) \cdot 2^{e} \cdot m = (z, s, e, m)$$

$$z = \begin{cases} 1 & \text{if } \tilde{a} = 0 \\ 0 & \text{otherwise} \end{cases}$$

[Ali+13,Rat+22]

[1] Fixed-point computation directly follows from our integer constructions.