浙江大学 区块链与数据安全全国重点实验室
STATE KEY LABORATORY OF BLOCKCHAIN AND DATA SECURITY
ZHEJIANG UNIVERSITY

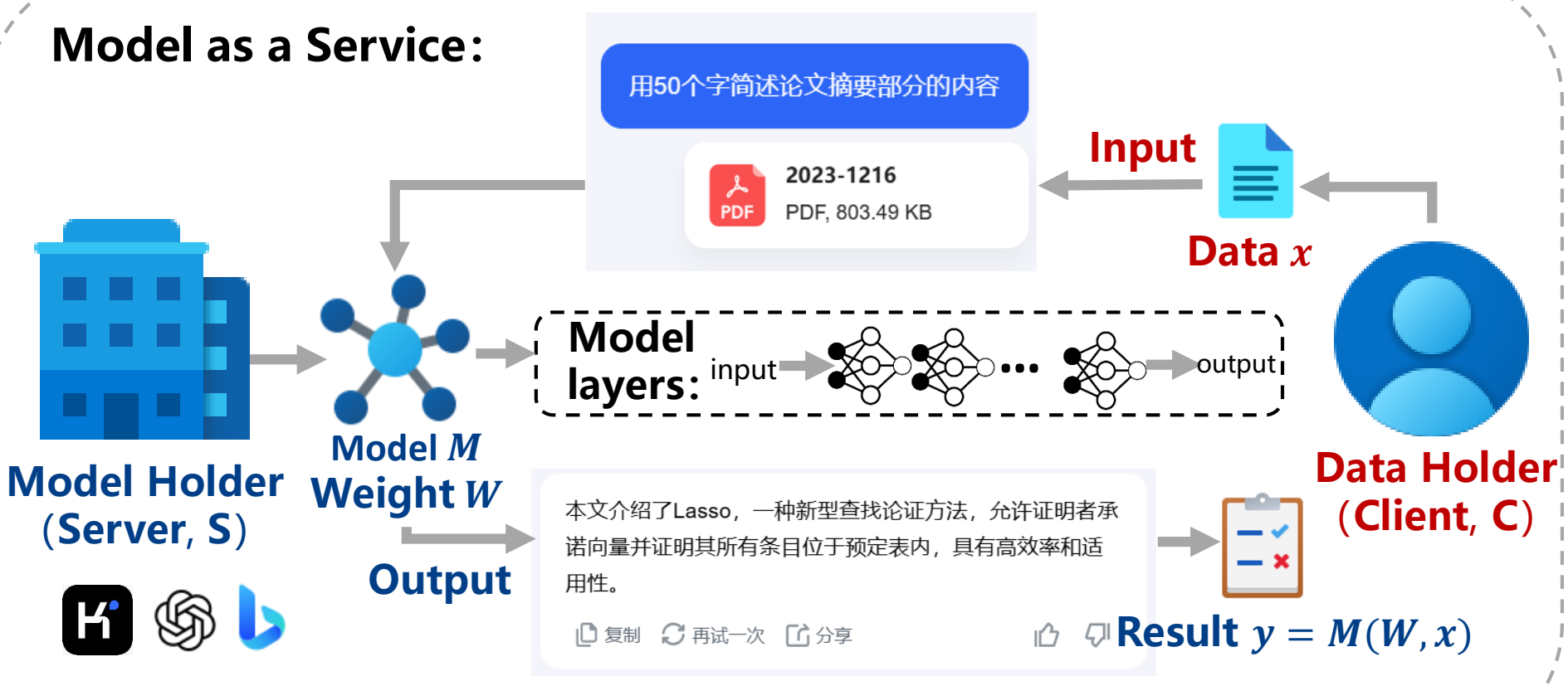
A New PPML Paradigm for Quantized Models

Tianpei Lu, Bingsheng Zhang*, Xiaoyuan Zhang, Kui Ren

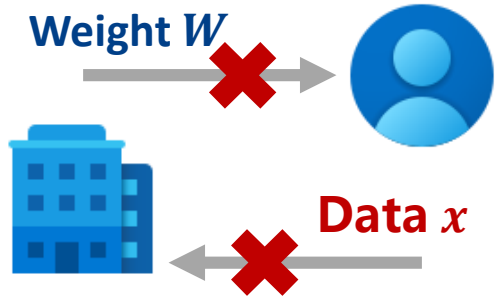
Zhejiang University

Background

Model as a Service:



Data Privacy:



S: keep W secret

C: keep x secret

Background

浙江大学 区块链与数据安全全国重点实验室
STATE KEY LABORATORY OF BLOCKCHAIN AND DATA SECURITY ZHEJIANG UNIVERSITY

Model as a Service:

Model Holder
(Server, S)

Model M
Weight W

Input

Data x

Secure Multiparty Computation

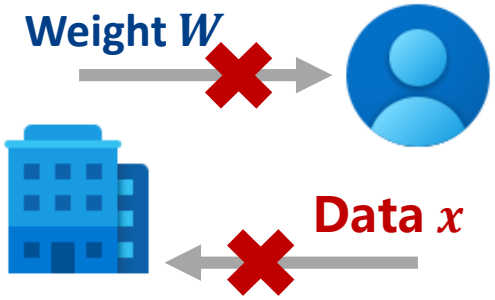
Evaluate $y = M(W, x)$

Output

Data Holder
(Client, C)

$M(W, x)$

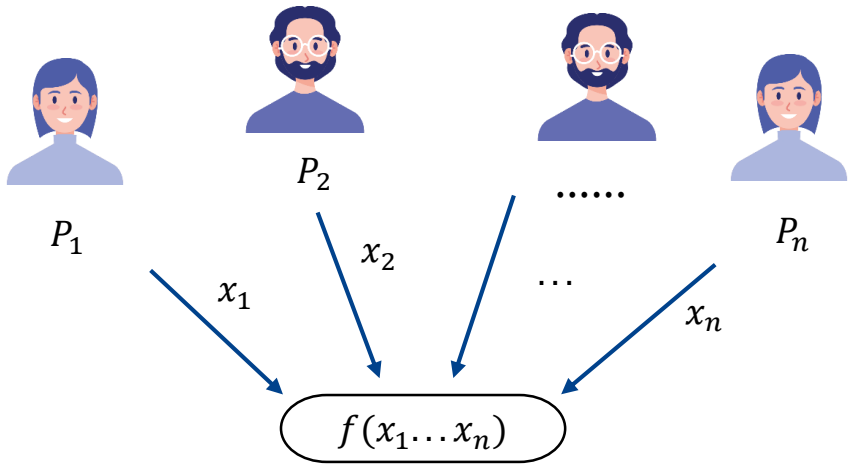
Data Privacy:



S: keep W secret

C: keep x secret

Secure Multiparty Computation



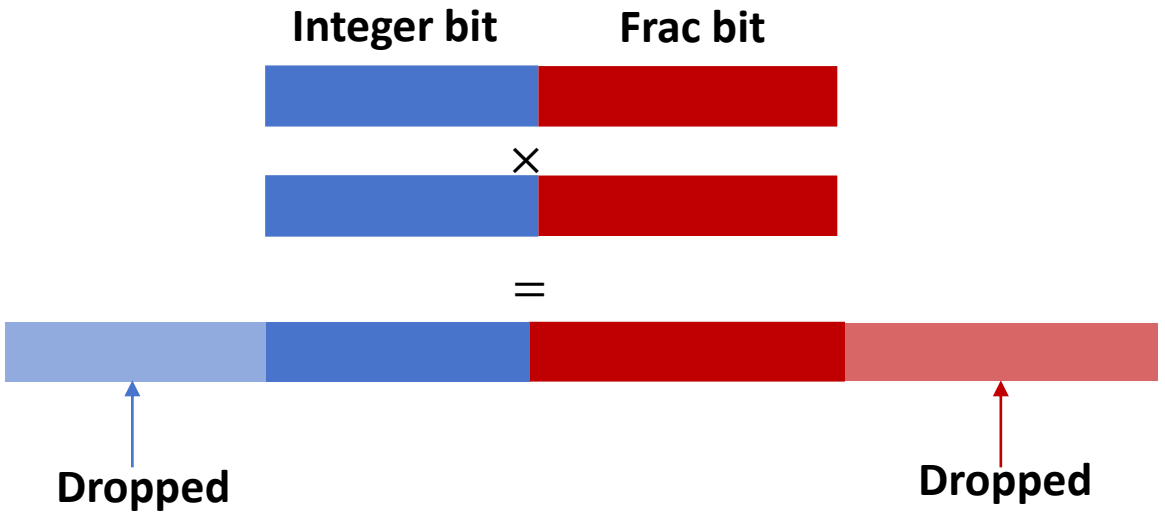
P_i input messages x_i and receive y_i

$$y_1 \ldots y_n = f(x_1 \ldots x_n)$$

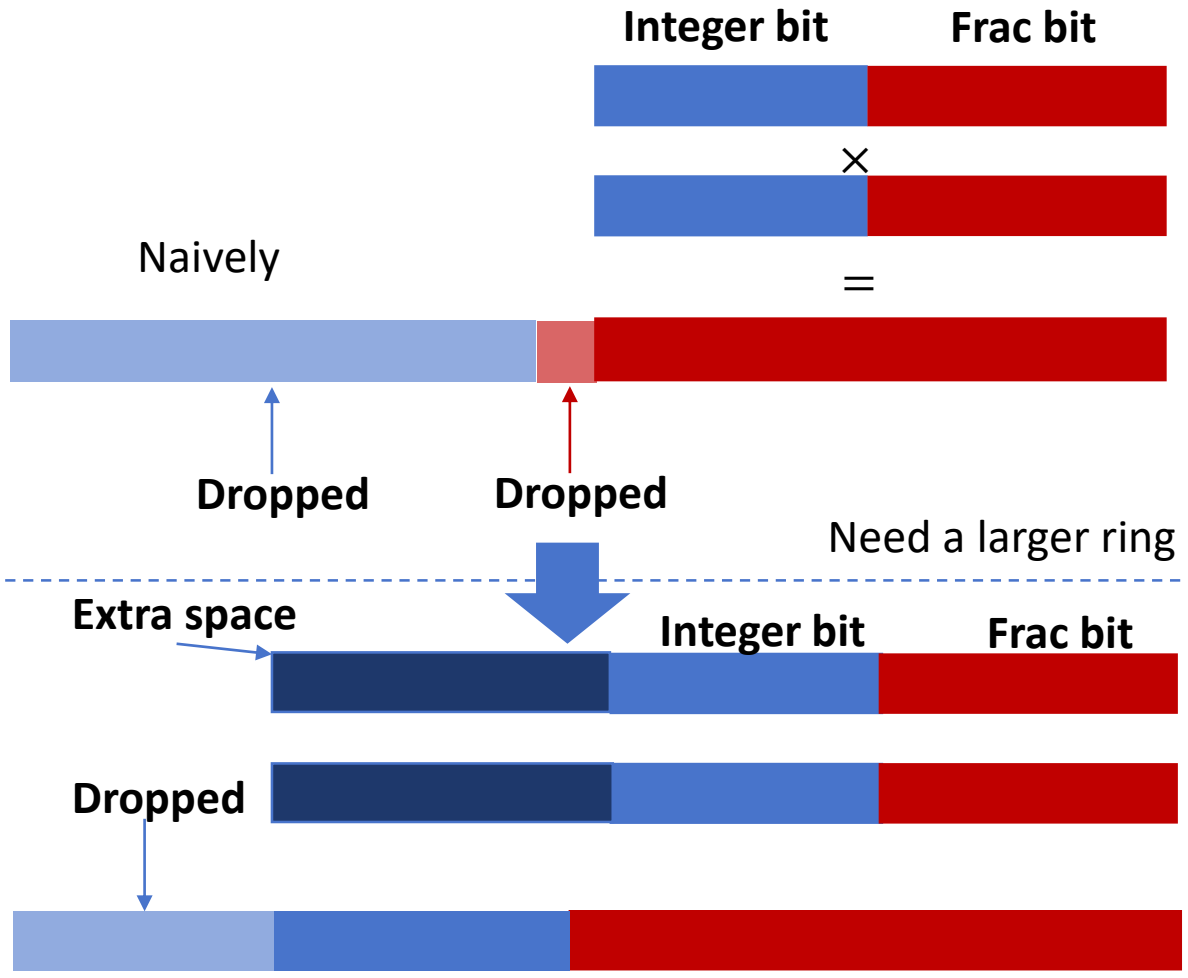
Typical Solution：Secure Multiparty Computation

Challenges of PPML (1)

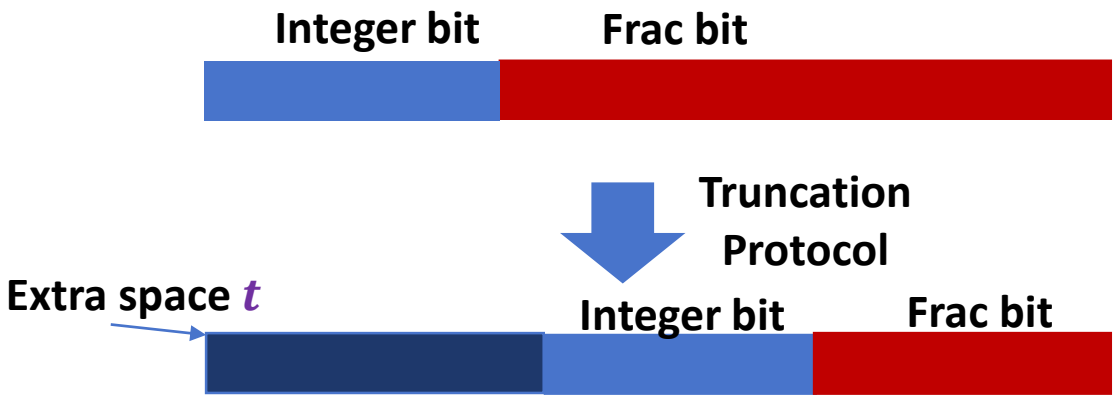
Fixed point multiplication



Multiplication over ring



Challenges of PPML (2)



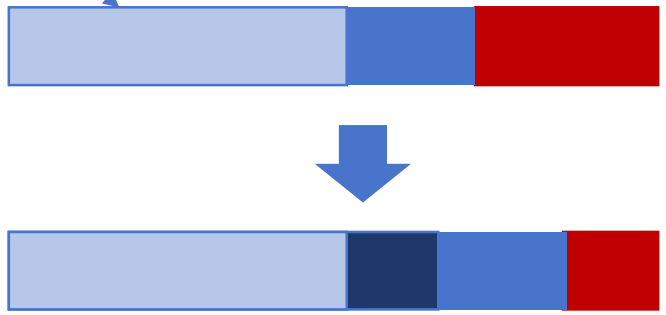
Secure evaluation

Deterministic truncation

Perform $y = f(x)$: $y = \mathbf{trunc}(x, t)$

Probabilistic truncation

l' **Zero bits**

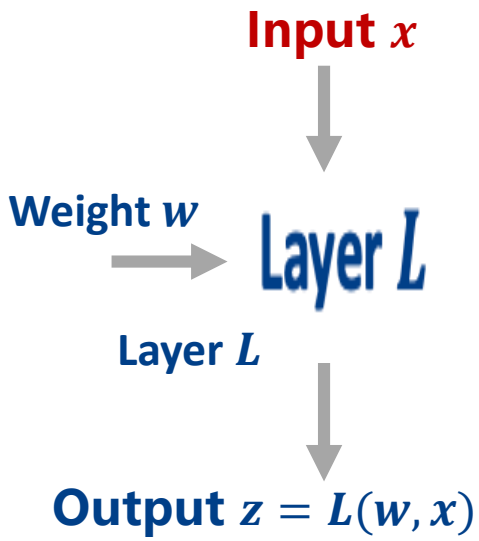


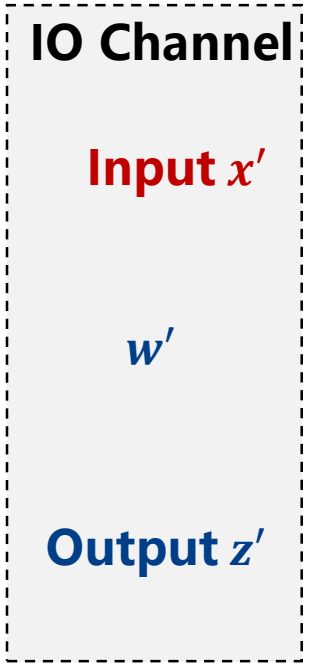
With $2^{1-l'}$ error rate

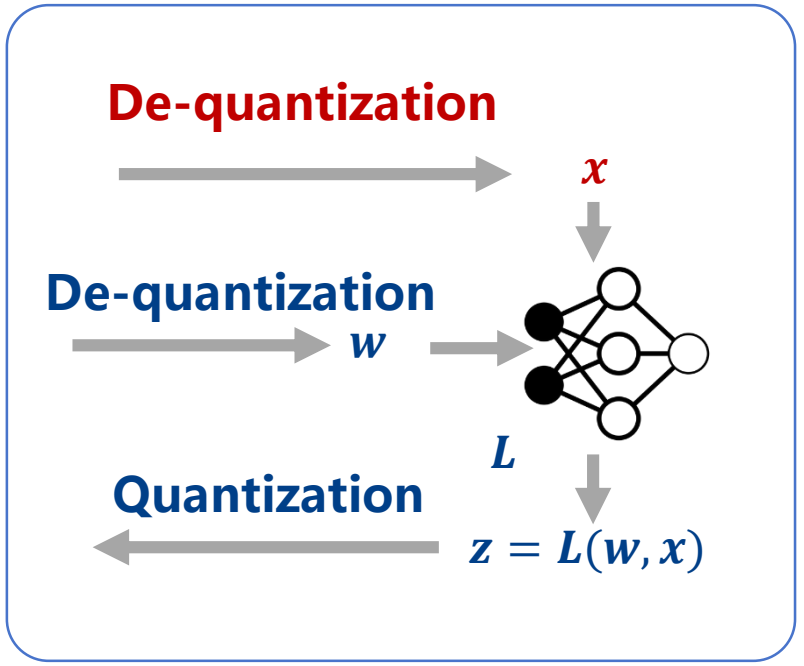
Need a larger ring

Truncation is expensive !!!

Quantization in PPML







x、w、z are Fp32 or Fp64

x'、w'、z' are Int8

Quantization: $X = \frac{X'}{s_X} + b_X$

De-quantization: $X = s_X(X' - b_X)$

Quantization in PPML

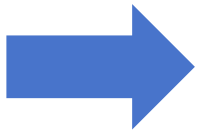
Challenges in Quantization PPML

$$X' = \frac{X}{s_X} + b_X$$

$$Y' = \frac{Y}{s_Y} + b_Y$$

$$\mathrm{Z} = XY$$

$$Z' = \frac{Z}{s_z} + b_z$$



$$Z' = (X' - b_X)(Y' - b_Y)\frac{s_X s_Y}{s_Z} + b_Z$$

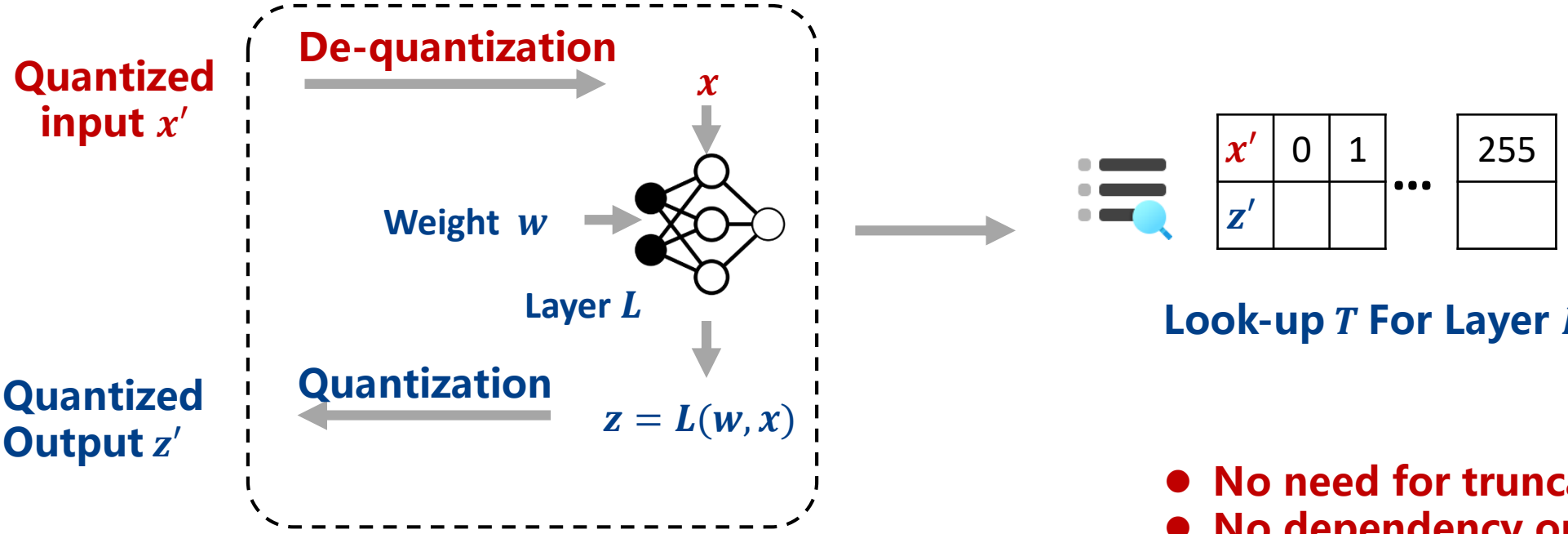
Fixed point scaler:
Require truncation

Int8 multiplication:
Calculated over Int16

Int8 offset

Quantization & Look-up table

Look-up table evaluation is all we need:



- No need for truncation
- No dependency on internal computational complexity.

Private look-up evaluation protocol

$x = [x]_0 + [x]_1$

Evaluate $[x] \to [f(x)]$ using Look-up table

Input $[x] \coloneqq \{[x]_0, [x]_1\}$

Shifted look-up table triple $\{[r], [f(r)], \ldots, [f(r+n-1)]\}$

S

$[r]_1$

$[f(x)]_1$

0	$[f(r)]_0$
1	$[f(r+1)]_0$
.....	
$x-r$	$[f(x)]_0$
.....	
$n-1$	$[f(r+n-1)]_0$

1. Open $x - r$

2. select $x - r$ th item

0	$[f(r)]_1$
1	$[f(r+1)]_1$
.....	
$x-r$	$[f(x)]_1$
.....	
$n-1$	$[f(r+n-1)]_1$

O(1) online communication

C

$[r]_0$

$[f(x)]_0$

Server knows f, but should not know r!!!

Private look-up evaluation protocol

(n-1, n)-ROT :

n l-bit messages $(m_0, \dots, m_n)$

(n-1, n)-ROT

Index j

$n-1$ l-bit messages $(m_0, \dots, m_n)\backslash m_j$

Sender

E.g.: can be constructed from log n ROT via GGM tree

Receiver

$M = (m_0, \dots, m_j, \dots, m_N)$

Left shift by r_0

$n \times n$

$+ + + = -u$

u Left shift by r_0 u' $+$ v $= w$

v

C

C randomly generates j column, $r_0 = j = 2$

(N-1, N)-ROT

i-th row Shift by i

$n \times n$

S

$M = (m_0, \dots, m_N)$

u $+$ $\hat{T}$ 1 2 3 0 $= s$

Left shift by r_0 $s' - w = [T]_0$

v $= [T]_1$

T 0 1 2 3

Lookup table

S Pick r_1, Left shift T by $r_1 = 1$ positions

C Holds

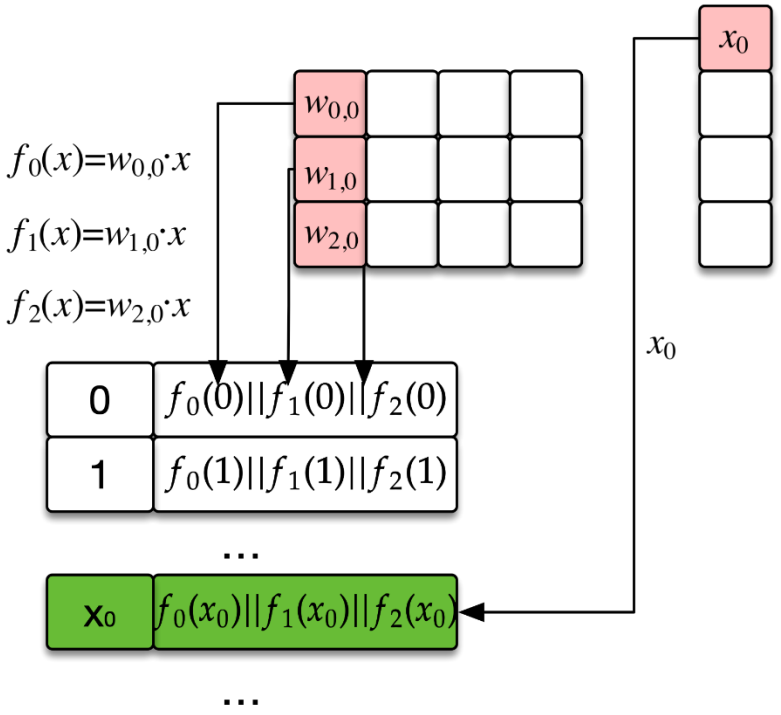
$[r]_0 = r_0, [T]_0$

S Holds

$[r]_1 = r_1, [T]_1$

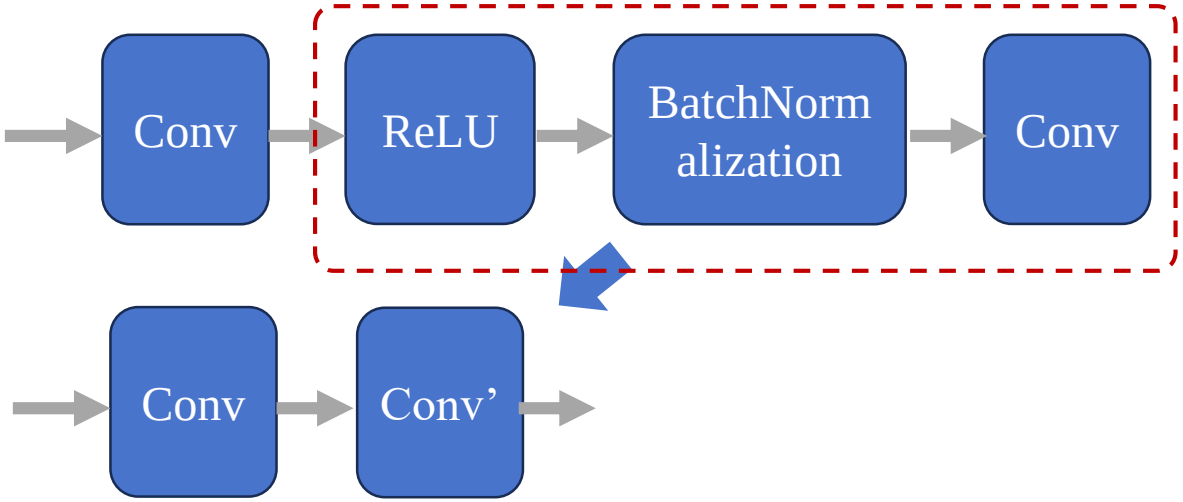
Optimization

Batch Scalar multiplications



Only need to reveal $x_0 - r$, for all $w_{0,i}$

Operator Fusion

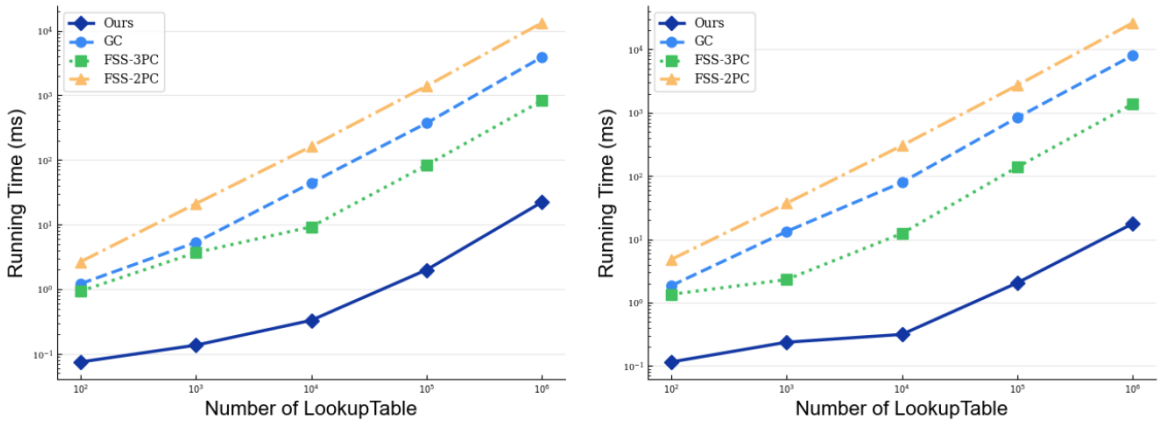


$$z = g(y), \mathrm{y} = f(\mathrm{x})$$

$$\mathrm{z} = F(\mathrm{x})$$

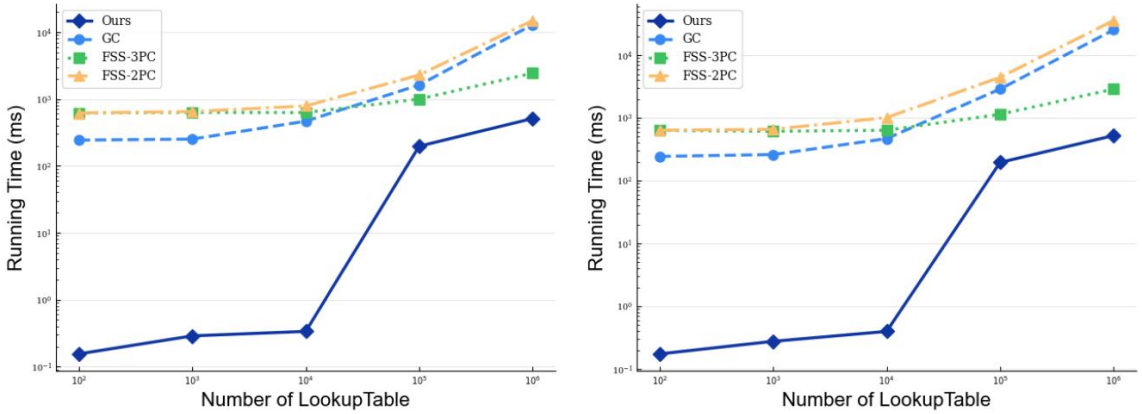
$$T := \{F(0), F(1), \ldots, F(n-1)\}$$

Performance



(a) 4-bit Quantization. (b) 8-bit Quantization.

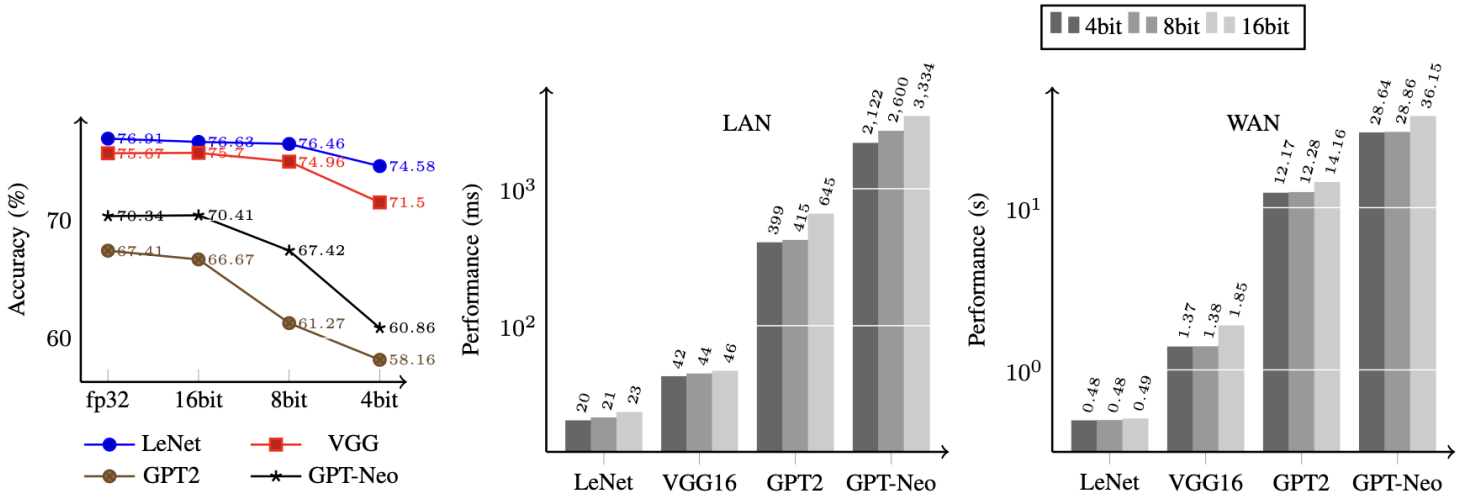
Online Performance comparison of Look-up table evaluation in the LAN setting



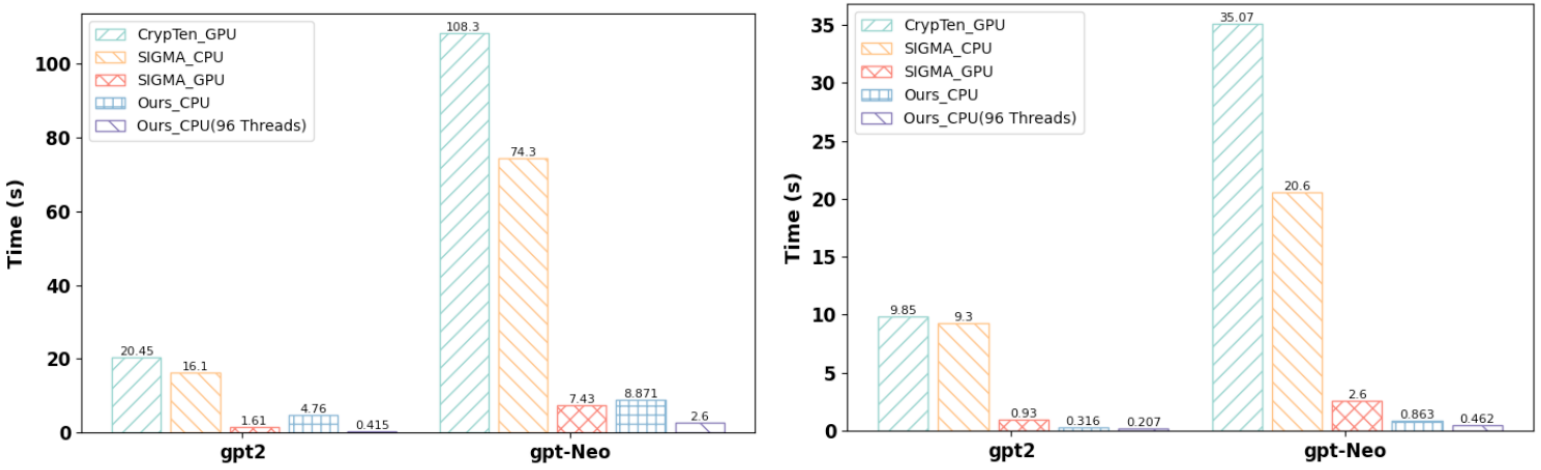
(a) 4bit Quantization. (b) 8bit Quantization.

Online Performance comparison of Look-up table evaluation in the WAN setting

Performance



Online Performance comparison of 4-bit/8-bit/16-bit quantization



(a) LAN setting (b) WAN setting

Online performance comparison for LLM with Sigma and CrypTen

Takeaway

- **Fixed-point truncation in PPML introduces heavy overhead, hindering its use in quantized evaluations.**
- **Our Private Lookup Table Evaluation Scheme bypasses truncation, supporting both linear and nonlinear computations.**
- **Our approach achieves 20–80× speedups over non-quantized methods.**

Thank You