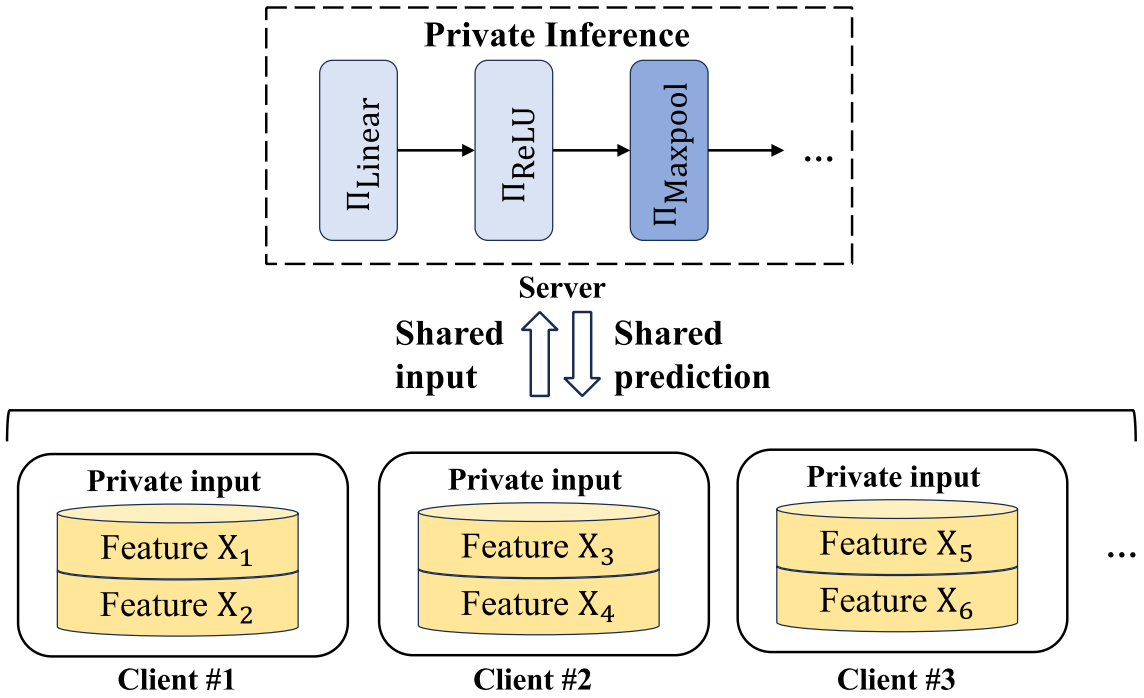
SecInfer: Secure and Effcient Model Inference on Vertically Partitioned Data

Haotian Deng, Hongwei Li, Hanxiao Chen, Meng Hao, Pengzhi Xing,
Jia Hu, Rui Zhang, Wenbo Jiang

Overview of Secure Inference in SecInfer



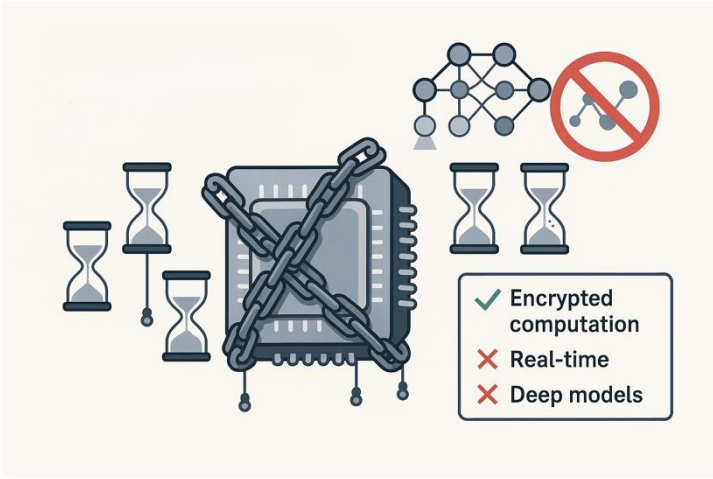
Privacy Concerns:

- Model ***parameters*** can be sensitive business secrets.
- Involved ***data*** is often privacy-sensitive (e.g., personal data like income tax, medical details, or non-personal corporate data).

Existing Solutions are not Sufficient

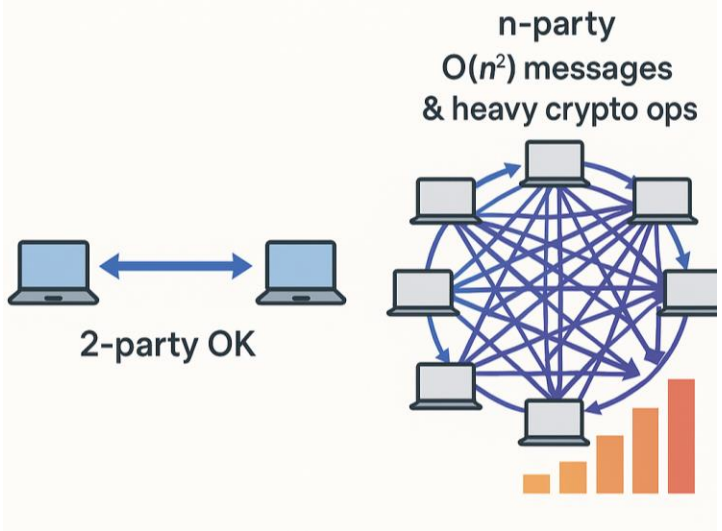
Homomorphic Encryption:

- ***High*** computational overhead, ***limited*** operation support …



Secure Multi-Party Computation:

- Often designed for ***two-party*** settings and ***unable to scale well*** to multi-party scenarios …



Drawbacks in existing non-linear protocols

Subtraction operations under Boolean shares will produce ***high costs*** in both execution time and communication.

Protocol $\Pi_{\mathsf{basicReLU}}$

Inputs: All parties hold an arithmetic sharing $\langle x \rangle^A$, and a module number p.

Processing:

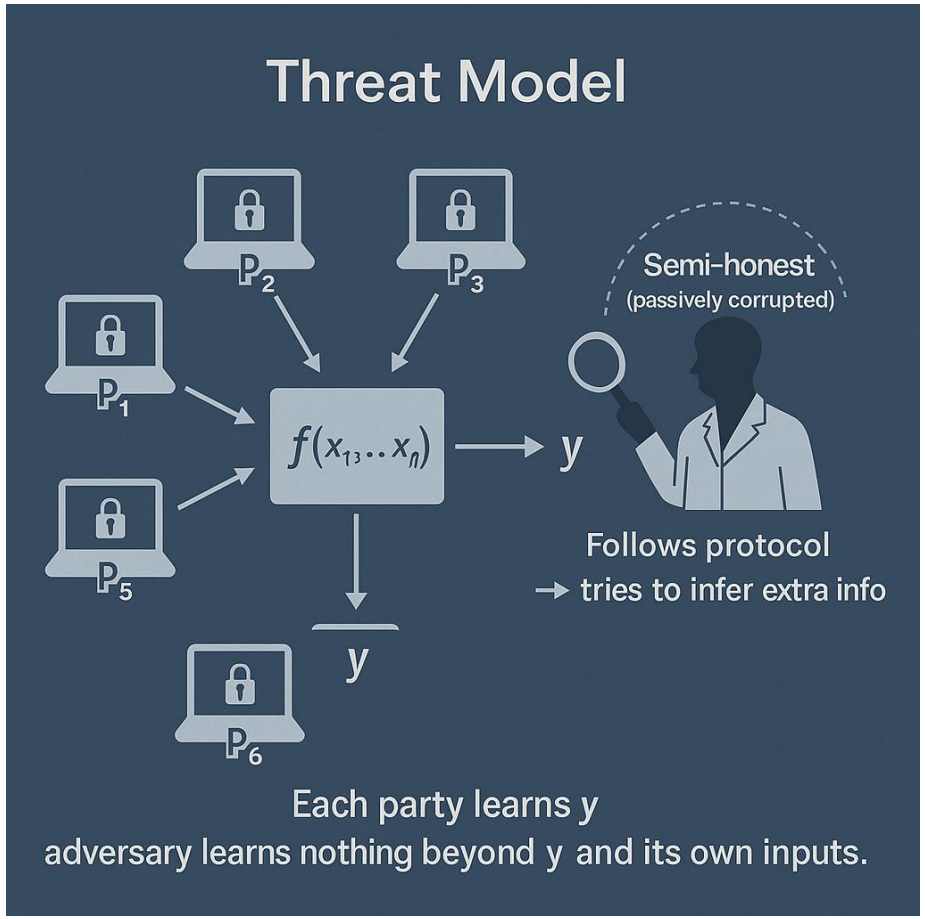
1) All parties call $\mathcal{F}_{\mathsf{A2B}}$ on input $\langle x \rangle^A$ to obtain $\langle x \rangle^B$.
2) All parties compute $\langle t \rangle^B \leftarrow \mathbf{1}\{\langle x \rangle^B > \frac{p-1}{2}\}$ to get the true sign bit of each sharing.
3) All parties compute $\langle x \rangle^B \leftarrow \langle x \rangle^B - \langle t \rangle^B \cdot p$ to convert the value representing in $\mathbb{F}_p$ to the true value.
4) All parties shift the most significant $\ell - s$ bits of $\langle x \rangle^B$ to the right, and fill the most significant s bits with its true sign bit $\langle t \rangle^B$, obtaining $\langle y \rangle^B \leftarrow \langle t \rangle^B || \ldots || \langle t \rangle^B || \langle x_{\ell-1} \rangle^B || \ldots || \langle x_s \rangle^B$.
5) All parties compute $\langle y \rangle^B \leftarrow \langle y \rangle^B \cdot \neg \langle t \rangle^B$ to set the negative value with the sharing of 0.
6) All parties call $\mathcal{F}_{\mathsf{B2A}}$ on input $\langle y \rangle^B$ to obtain $\langle y \rangle^A$.

Our Contributions

- Propose ***an MPC framework*** for non-linear functions in model inference with the table lookup technique.
- Achieve ***3.71×*** and ***3.42×*** improvement in communication and computation cost.

Threat Model

- ***Semi-honest*** probabilistic polynomial time(PPT) adversary.
- Adversaries follow protocol ***but*** try to infer extra information beyond the output and their own inputs.



Optimized ReLU Protocol

1. A2B conversion ($\mathcal{F}_{A2B}$).
2. Compute sign bit ($\langle t \rangle^B$).
3. Set negative values to zero ***early***.
4. Direct truncation for scale recovery (***since numbers are non-negative***).
5. B2A conversion ($\mathcal{F}_{B2A}$).

Protocol Π_{ReLU}

Inputs: All parties hold an arithmetic sharing $\langle x \rangle^A$, and a module number p.

Processing:

1) All parties call $\mathcal{F}_{\text{A2B}}$ on input $\langle x \rangle^A$ to obtain $\langle x \rangle^B$.
2) All parties compute $\langle t \rangle^B \leftarrow \mathbf{1}\{\langle x \rangle^B > \frac{p-1}{2}\}$ to get the true sign bit of each sharing.
3) All parties compute $\langle x \rangle^B \leftarrow \langle x \rangle^B \cdot \neg \langle t \rangle^B$.
4) All parties set $\langle y \rangle^B \leftarrow \langle x_{\ell-1} \rangle^B || \ldots || \langle x_s \rangle^B$.
5) All parties call $\mathcal{F}_{\text{B2A}}$ on input $\langle y \rangle^B$ to obtain $\langle y \rangle^A$.

Optimized Maxpool Protocol

1. Extract and flatten windows from input matrix into a new matrix.
2. Initialize $\langle T_{max} \rangle^A$ with the first column.
3. Iterate through columns:
   a. Compute difference between current max and next column($\langle T_{sub} \rangle^A$).
   b. Apply $\mathcal{F}_{ReLU}$ (optimized ReLU) to set negative values to zero.
   c. Update $\langle T_{max} \rangle^A$ (this effectively finds the maximum of two values).
4. $\langle T_{max} \rangle^A$ contains row-wise maximums.
5. Reshape for final output.

Protocol Π_{Maxpool}

Inputs: All parties hold an arithmetic sharing $\langle \boldsymbol{X} \rangle^A$ of size $n \times n$, window size $m \times m$, stride size s, and a modulus number p.

Processing:

1) Extract each window of size $m \times m$ from $\langle \boldsymbol{X} \rangle^A$ with stride s and flatten it into a row vector. Concatenate all row vectors to form a new matrix $\langle \boldsymbol{Y} \rangle^A$ of size $t^2 \times m^2$, where $t = \lfloor \frac{n-m}{s} \rfloor + 1$.
2) Initialize $\langle T_{\mathsf{max}} \rangle^A \leftarrow \langle \boldsymbol{Y}[\cdot, 0] \rangle^A$.
3) For each column j from 1 to $m^2 - 1$:
   a) Compute: $\langle T_{\mathsf{sub}} \rangle^A \leftarrow \langle T_{\mathsf{max}} \rangle^A - \langle \boldsymbol{Y}[\cdot, j] \rangle^A$.
   b) ReLU: $\langle T_{\mathsf{relu}} \rangle^A \leftarrow \mathcal{F}_{\mathsf{ReLU}}(\langle T_{\mathsf{sub}} \rangle^A, p)$.
   c) Update: $\langle T_{\mathsf{max}} \rangle^A \leftarrow \langle T_{\mathsf{relu}} \rangle^A + \langle \boldsymbol{Y}[\cdot, j] \rangle^A$.
4) After processing all columns, $\langle T_{\mathsf{max}} \rangle^A$ contains the maximum value for each row of $\langle \boldsymbol{Y} \rangle^A$.
5) Reshape $\langle T_{\mathsf{max}} \rangle^A$ into a matrix of size $t \times t$ for the final output.

Experiment Setup

- **Implementation:** C++ based on GYKW[1] (for A2B/B2A) and MOTION[2] (for non-linear operations).
- **Focus:** Online phase (offline part precomputed).
- **Network Settings:** Evaluated in both LAN and WAN.
  - **LAN:** 1 Gbps bandwidth, 0.1 ms latency.
  - **WAN:** 200 Mbps bandwidth, 100 ms latency.
- **Parameters:** 128-bit security, plaintext prime $2^{32} - 2^{30}+1$, ciphertext prime > 530 bits, N = 65536 slots.
- **Runs:** Average results across 10 runs.

[1] Scalable mixed-mode mpc, *IEEE S&P*, 2024.

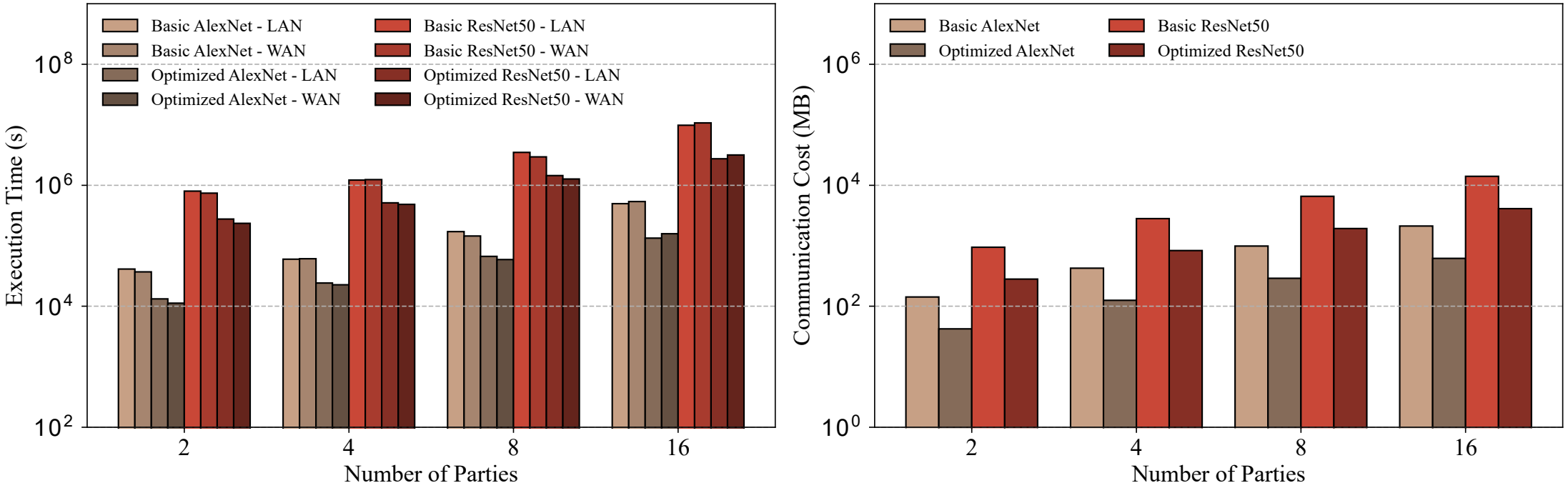
[2] Motion – a framework for mixed-protocol multi-party computation, *ACM TOPS*, 2022

Microbenchmarks - ReLU & Maxpool

Protocol	Setting	Number of Parties			
		2	4	8	16
ReLU (basic)	Comm.	94	281	654	1400
	LAN	45.60	69.43	200.13	558.56
	WAN	42.33	70.95	168.37	611.61
ReLU (optimized)	Comm.	28	83	192	409
	LAN	15.79	29.33	82.99	156.74
	WAN	13.45	27.72	72.62	180.32
Maxpool (basic)	Comm.	169	506	1177	2520
	LAN	94.15	121.38	336.39	1088.27
	WAN	77.85	121.66	289.17	1153.57
Maxpool (optimized)	Comm.	50	149	345	737
	LAN	24.47	42.20	105.86	261.14
	WAN	20.48	38.19	97.45	335.49

Optimized protocols provide significant performance gains in both ***computation*** and ***communication cost***.

End-to-End Inference Evaluation



Substantial reduction in ***inference time*** and ***communication costs*** for both models with optimized protocols.

Thank You

Contact us:

Haotian Deng, *h.deng@std.uestc.edu.cn*