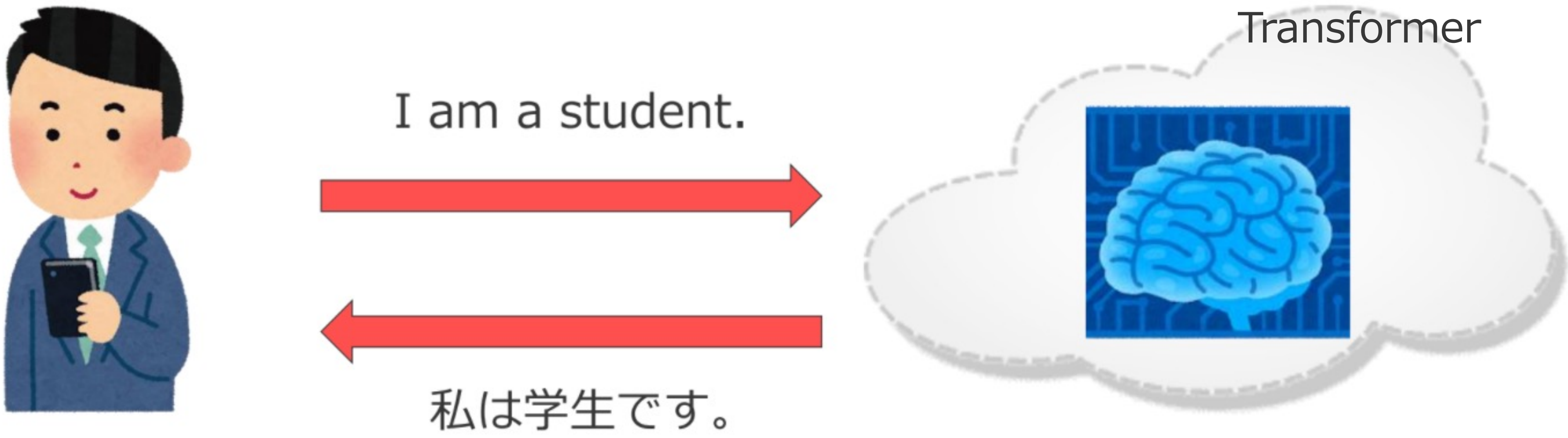
Privformer: Privacy-preserving Transformer with MPC

Yoshimasa Akimoto	U. Tsukuba
Kazuto Fukuchi	U. Tsukuba
Yohei Akimoto	U. Tsukuba
Jun Sakuma	Tokyo Tech./RIKEN AIP

Transformer

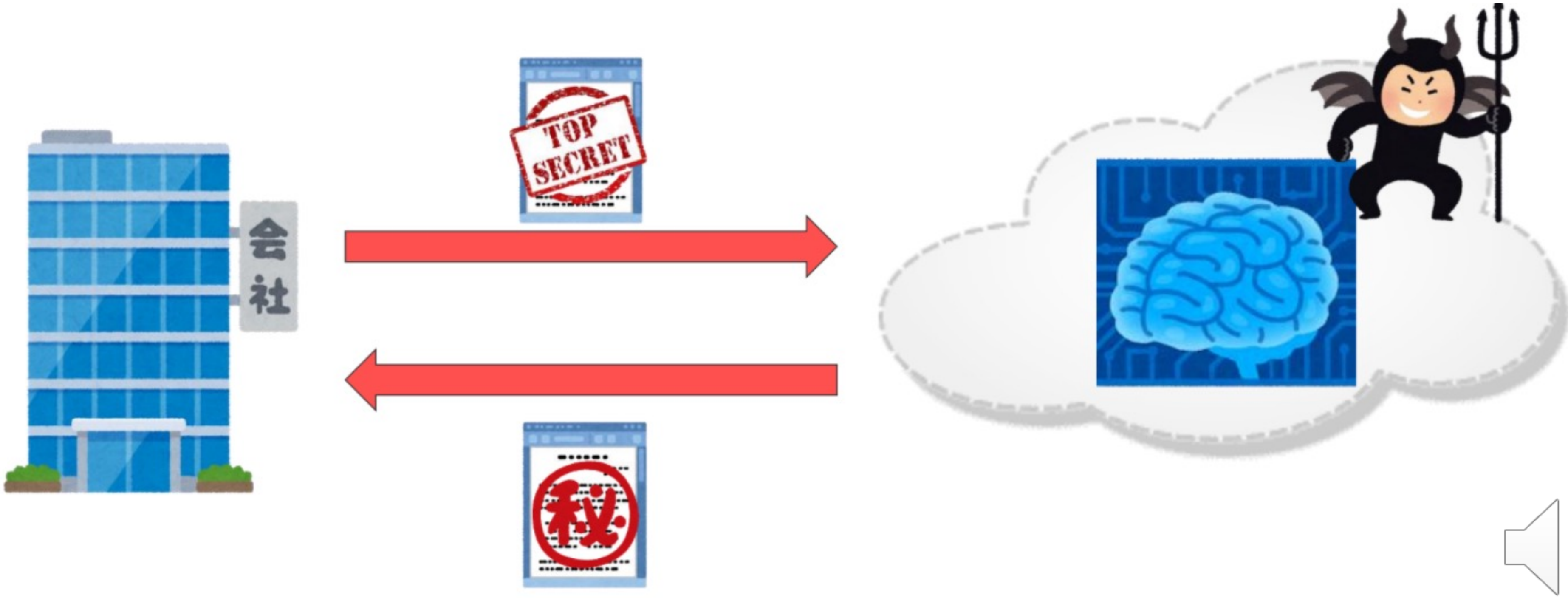
- Generic deep learning model for sequence processing [1].
- Used for a variety of applications including machine translation, question answering, and automatic summarization
- Foundational architecture used in modern language generation models such as ChatGPT, BERT, and LLaMA



[1] Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems. 2017.

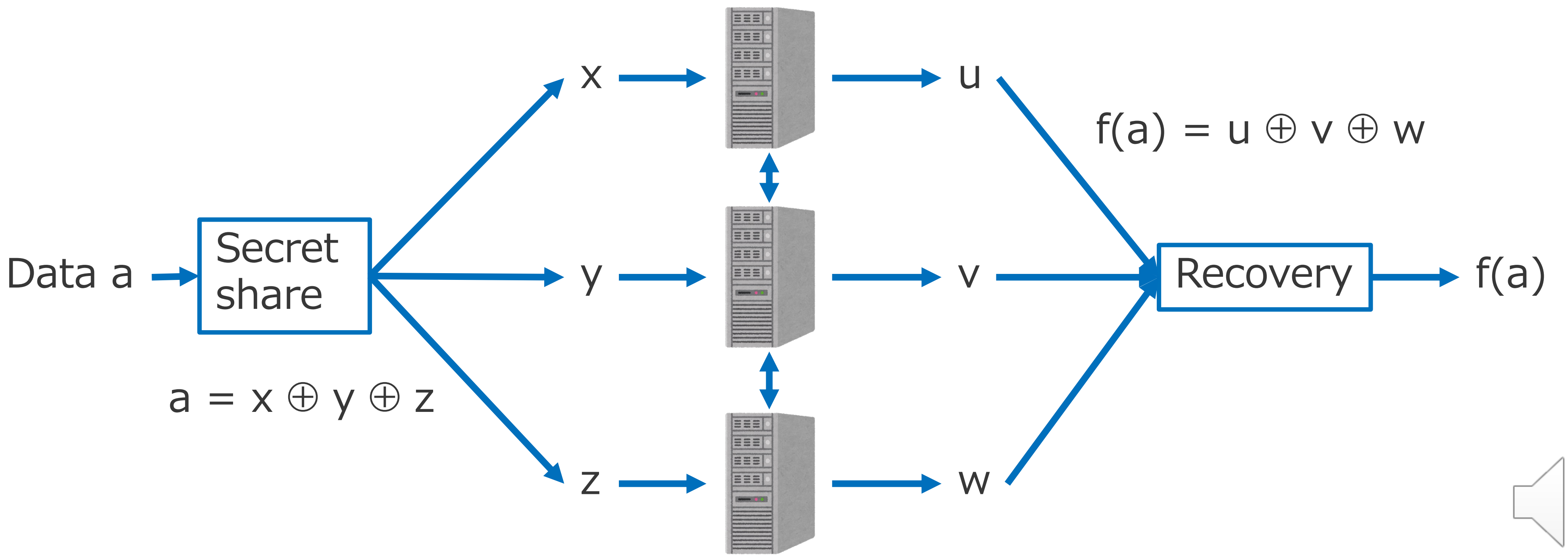
Privacy concern

- Privacy protection is an issue when using AI services via API
  - E.g., I want to use ChatGPT to solve a question related to my business.
- If prompts contain personal or confidential information, users cannot send them to the server
- Also, the model is so huge that it is not practical to run it locally.



Secure multi-party computation (MPC)

- Private information is distributed among multiple parties in the form of secret shares, and the computation is performed securely among multiple parties over the shares.



Our goal

Application of MPC to Transformer

- Evaluation of the Transformer contains numerous number of exponents and inverses, which can be a bottleneck
- Less studied compared to CNN and RNN

Goal

- To propose an MPC-friendly Transformer architecture for the Transformer model
- To implement Transformer inference on MPC

Related work

SecureNN[2], Falcon[3]

- DNN and CNN training and inference on MPC
- Supports convolution, ReLU, Maxpool, and Batch Normalization layers

SIRNN[4], SecureNLP[5]

- Enabling RNN inference on MPC
- Can compute , $\sigma(x) = \frac{1}{1+e^{-ax}}$ $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ used in RNNs

[2] Wagh, Sameer, Divya Gupta, and Nishanth Chandran. "SecureNN: 3-Party Secure Computation for Neural Network Training." *Proc. Priv. Enhancing Technol.* 2019.3 (2019): 26-49.
[3] Wagh, Sameer, et al. "Falcon: Honest-majority maliciously secure framework for private deep learning." *arXiv preprint arXiv:2004.02229* (2020).
[4] Rathee, Deevashwer, et al. "SIRNN: A Math Library for Secure RNN Inference." *arXiv preprint arXiv:2105.04236* (2021).
[5] Feng, Qi, et al. "SecureNLP: A system for multi-party privacy-preserving natural language processing." *IEEE Transactions on Informatio Forensics and Security* 15 (2020): 3709-3721.

Related work

Wang et al.[6]

- Enabling Transformer inference on MPC
- Supports the embedding process, but does not support softmax in the attention layer
- Proposed a method to speed up the Embedding process
- No improvement proposed for Attention processing

[6] Wang, Yongqin, et al. "Characterizing and Improving MPC-based Private Inference for Transformer-based Models." *NeurIPS 2021 Workshop Privacy in Machine Learning*. 2021.

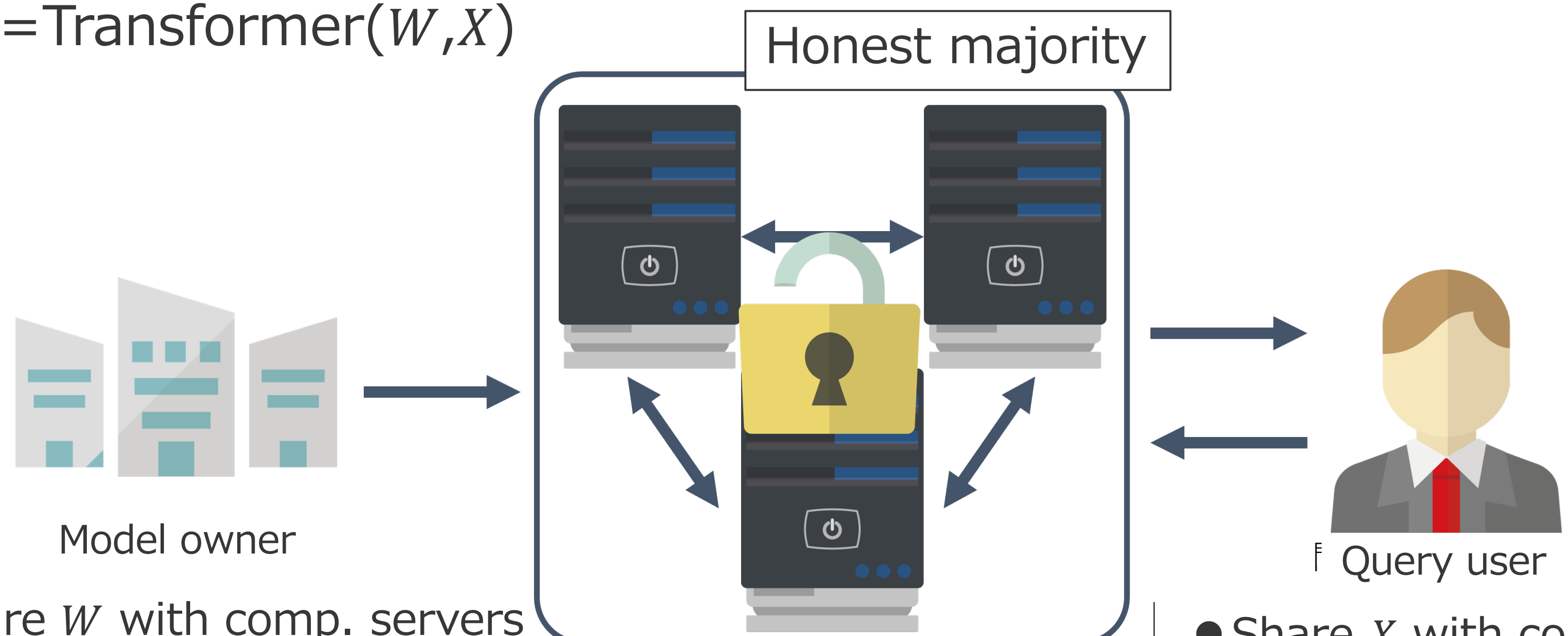
Contribution of this study

- Introducing Attention using ReLU, instead of Softmax to reduce the computation and communication cost of the Attention layer on MPC.
- Implemented Transformer inference on FALCOM framework and measured execution time and communication cost by experiments

Problem setting

Parties

- Model owner: hold pmodel parameters W for the Transformer
- Query user: hold input X used for inference
- Computing servers: three servers that perform MPC to compute Y=Transformer(W,X)



Computing servers

Model owner:
- Share W with comp. servers
- Never learns X, Y

Query user:
- Share X with comp servers
- Never learns W and learn$Y$

Computing servers:
- Obtain shares of W, X and compute Y
- Never learns W, X, Y

MPC on FALCON

Falcon framework was used for MPC implementation

- Three party MPC
- Honest-majority
- Support efficient computation of functions frequently used for deep learning

Share representation

- Fixed-point rep. :$x \in \mathbb{R} \rightarrow \lfloor x \cdot 2^{FP} \rfloor (\mathrm{mod}\ L) = \hat{x} \in \mathbb{Z}_L$, FP: Precision param.
- Share rep. :share of $\hat{x}$ $\langle \hat{x} \rangle = (\hat{x}_1, \hat{x}_2, \hat{x}_3)\ where\ \hat{x} \equiv \hat{x}_1 + \hat{x}_2 + \hat{x}_3 \mod L$

Arithmetic operations

- addiction : $\langle x \rangle + \langle y \rangle = \langle x + y \rangle$
- constant magnification : $c\langle x \rangle = \langle cx \rangle$
- Multiplication : $\Pi_{\mathrm{Mult}}(\langle x \rangle, \langle y \rangle) \rightarrow \langle xy \rangle$

Matrix multiplication : $\Pi_{\mathrm{MatMul}}(\langle X \rangle, \langle Y \rangle) \rightarrow \langle XY \rangle$

- $X \in \mathbb{R}^{l \times m}, Y \in \mathbb{R}^{m \times n}$ Time $O(lmn)$, Communication $O(ln)$

MPC on FALCON

Nonlinear operations

- Computed by combination of addition, constant multiplication, and multiplication
- Relatively costly since containing iterative communication
- Results are approximation when iterative methods are used

演算	Time[ms]	Comm[B]	Rounds	Approx.?
$\Pi_{\mathrm{Mult}}(\langle x\rangle, \langle y\rangle) \to \langle xy\rangle$	1.17	0.02	4	
$\Pi_{\exp}(\langle x\rangle) \to \langle e^x\rangle$	9.78	0.13	32	✓
$\Pi_{\mathrm{Inv}}(\langle x\rangle) \to \langle 1/x\rangle$	43.84	0.85	215	✓
$\Pi_{\mathrm{InvSqrt}}(\langle x\rangle) \to \langle 1/\sqrt{x}\rangle$	57.47	1.03	260	✓
$\Pi_{\mathrm{ReLU}}(\langle x\rangle) \to \langle \mathrm{ReLU}(x)\rangle$	6.21	0.13	30	

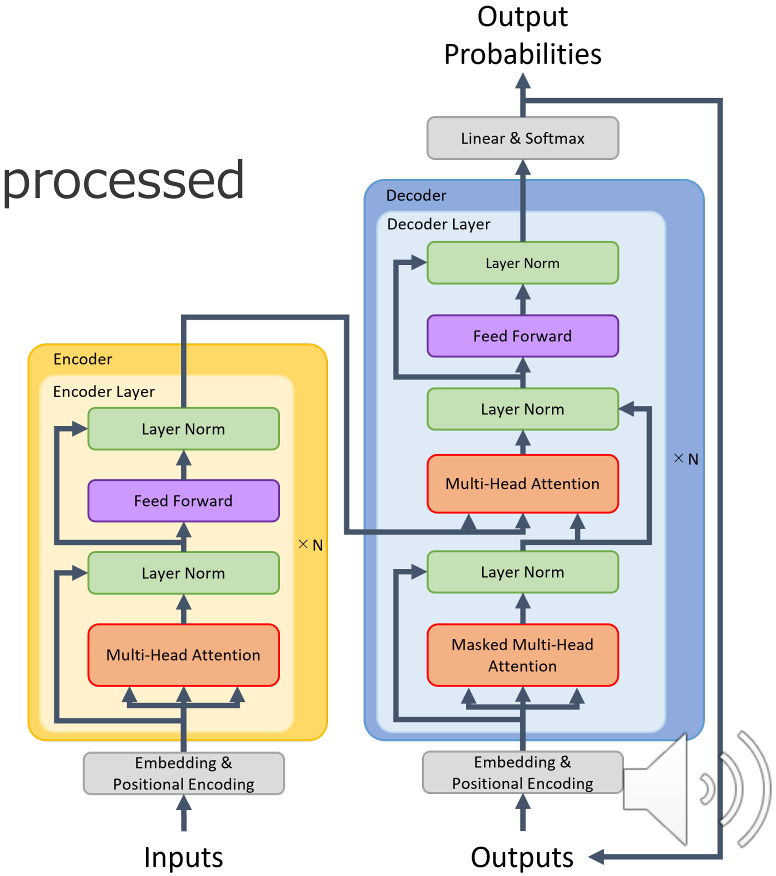
How can we realize Transformer on MPC using them?

How to realize MPC for Transformer

- Encoder transforms input token sequence X into latent representation
- Decoder transforms Z to X
- Embedding, Positional Encoding
- Linear & Softmax layer
- Multi-Head Attention
- Masked Multi-Head Attention
- Feed Forward Network
- Layer Normalization

Pre/post processing
Local computation works

Need to be processed with MPC



Complexity analysis of MPC for Multi-Head Attention

When employing existing primitives naively, which part can be the bottleneck?

- For input $Q, K, V \in \mathbb{R}^{S \times d}$ where S is input sequence length and d is the dimension of embedding vector:
- Computed with addition, constant mult., Π_{MatMul}, Π_{Exp}, Π_{Div}

Π_{MatMul} Π_{Exp}

$$\mathrm{Attention}(Q, K, V) = \mathrm{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right) V \ where\ \mathrm{Softmax}(\boldsymbol{x}_i) = \frac{e^{\boldsymbol{x}_i}}{\sum_{k=1}^{d_m} e^{\boldsymbol{x}_k}}$$

- Π_{Inv}: time/comm. complexity $O(S)$
- Π_{Exp}: time/comm. complexity $O(S^2)$
- Π_{MatMul} for QK^T: time/comm. complexity $O(S^2)$

bottleneck

MPC-friendly attention: ReLU Attention

ReLU Attention[7]

- Relacing Softmax with projection with random matrix $\Omega \in \mathbb{R}^{d\times r}$ and ReLU
- By computing $K'^{T}V$ first, avoid dealing with S×S matrix, achieving $O(rS)$ complexity
- With sufficiently large r, the predictive performance is comparable to Softmax Attention

$$\mathrm{Attention}(Q,K,V) = \mathrm{Softmax}\left(\frac{\underline{QK}^T}{\sqrt{d}}\right)V$$

$O(S^2)$ is required for QK^T

$$\mathrm{ReLU\ Attention}(Q,K,V,\Omega) = cQ'\underline{(K'^{T}V)}$$

where $Q' = \underline{\mathrm{ReLU}(Q\Omega)}$ and $K' = \underline{\mathrm{ReLU}(K\Omega)}$

$K'^{T}V$ ca n be computed in $O(rS)$

$Q,K,V \in \mathbb{R}^{S\times d}$, $\Omega \in \mathbb{R}^{d\times r}$, $Q',K' \in \mathbb{R}^{S\times r}$, r: dimension after projection

[7] Choromanski, Krzysztof, et al. "Rethinking attention with performers." *arXiv preprint arXiv:2009.14794* (2020).

MPC implementation of ReLU attention

Secure ReLU Attention

- ReLU Attention

$$\text{ReLU Attention}(Q,K,V,\Omega) = cQ'(K'^{T}V)$$

$$\text{where } Q' = \text{ReLU}(Q\Omega) \text{ and } K' = \text{ReLU}(K\Omega)$$

- MPC ReLU Attention

$$\langle \text{ReLUAttention}(Q,K,V,\Omega)\rangle \leftarrow \text{c}\Pi_{\text{MatMul}}(\langle Q'\rangle, \Pi_{\text{MatMul}}(\langle (K')^{T}\rangle, \langle V\rangle))$$

$$\text{where } \langle Q'\rangle = \Pi_{\text{ReLU}}\left(\Pi_{\text{MatMul}}(\langle Q\rangle, \langle \Omega\rangle)\right),$$

$$\langle K'\rangle = \Pi_{\text{ReLU}}\left(\Pi_{\text{MatMul}}(\langle K\rangle, \langle \Omega\rangle)\right)$$

Time/space complexity of Π_{MatMul}, Π_{ReLU} is $O(rS)$

Round complexity is $O(1)$

Summary of complexity analysis and outline of experiments

Summary of complexity analysis

	Computation				Rounds
	Π_{MatMul}	Π_{Exp}	Π_{Inv}	Π_{ReLU}	Rounds
Softmax Attention	$O(S^2)$	$O(S^2)$	$O(S)$	-	$O(1)$
ReLU Attention	$O(rS)$	-	-	$O(rS)$	$O(1)$
Masked Softmax Attention	$O(S^2)$	$O(S^2)$	$O(S)$	-	$O(1)$
Masked ReLU Attention (Naive)	$O(rS)$	-	-	$O(rS)$	$O(S)$
Masked ReLU Attention (QK first)	$O(S^2)$	-	-	$O(S^2)$	$O(1)$

Experiments

- 1) Comparison of Softmax Attention and ReLU Attention in time and communication
- 2) End-to-end evaluation of time and communication for Transformer inference

Experimental environment

Computing servers (Amazon AWS)

- OS:Ubuntu 18.04 LTS, CPU:2.9 GHz Intel Xeon E5-2666 v3 processor, RAM:64GB

Network environment

- LAN : All comp servers were located in the same region, Average bandwidth during the experiment was 4.93 Gbits/s, average ping response time was 1.17 ms
- WAN : All comp servers were located in Ohio, Tokyo, and London. Average bandwidth during the experiment was 97.4 Mbits/s, average ping response time was 141.67 ms

Experimental settings

Parameter settings

- Parameter follows FALNCON
- Ring on Z_L where $L = 2^{32}$.
- Fixed-point representation with 13-bit precision
- FALNCON supports honest majority, but for experimental evaluation, we assumed semi-honest adversaries
- Transformer parameters: $d_m = 512, d_{ff} = 2048, h = 8, d = 64, r = 266$, Encoder/Decoder layer $N = 6$, batch size :1

Other settings

- Results are average of ten trials
- Exponentiation was approximated with 4th order Chebyshev polynomial
- For Newton method to compute inverse, the iteration number was set as four

1) Comparison of Softmax Attention and ReLU Attention

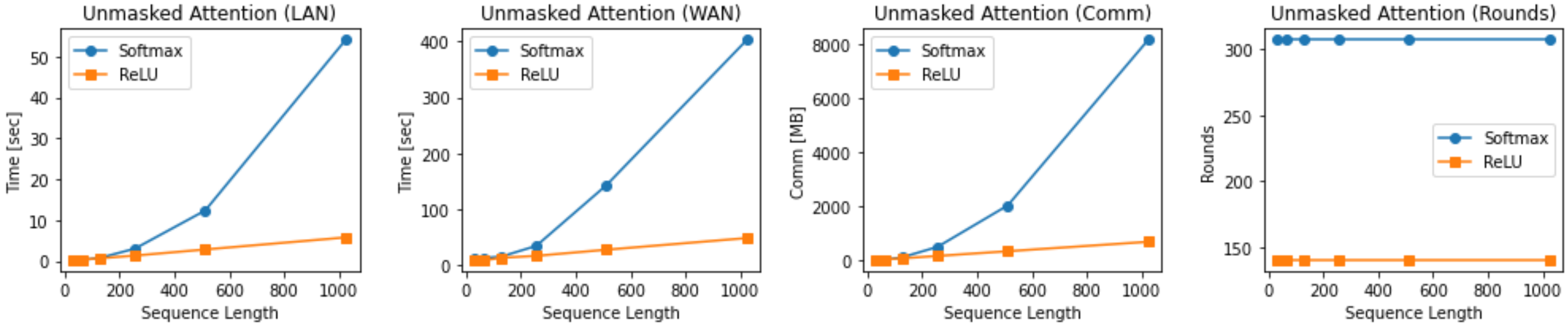


Fig. 1. : S=32, ⋯, 1024. Comparison of Softmax Attention and ReLU Attention in comp time [sec] , communication [MB] , and rounds. r=266

- Softmax Attention: time and communication increase quadratically
- ReLU Attention: time and communication increase linearly
- For a sequence length of 1024,
- 9.41x faster in LAN, 8.25x faster in WAN
- communication is reduced to 1/12

2) End-to-end evaluation of Transformer inference

Table2 . Time [sec] and comm. [MB] of each layer, $S = 64$

	Time [sec]		Comm	Rounds
	LAN	WAN	[MB]	
Multi-Head Attention	0.611	10.944	48.361	160
Masked Multi-Head Attention	0.613	12.424	47.231	192
Feed Forward Network	1.671	6.669	19.136	38
Layer Normalization	0.077	10.753	1.64	278
Encoder	14.141	253.164	424.668	4524
Decoder	17.768	402.85	717.894	7344

- Encoder is executed once and Decoder is executed S times
- When s=64, 19 minutes in LAN, 7.23 hours in WAN

Conclusion

- Introduced ReLU Attention as MPC-friendly attention
- Proposed an algorithm for Masked ReLU Attention for MPC
- Experimental comparison of Softmax Attention and ReLU Attention shows reduction of computation and communication cost
- Showed that a 64-word sentence can be inferred in about 20 minutes in a LAN environment