



北京郵電大學

Beijing University of Posts and Telecommunications

PCNNcec：基于云-边缘-客户端协作的高效 隐私保护卷积神经网络推理

论文研讨

Speaker：刘洋



目录

1. PCNNcec 论文摘要，研究背景、动机、贡献
2. 系统方案陈述
3. 预备知识
4. 系统实现
5. 实验、安全性证明
6. 论文讨论



目录

1. PCNNcec 论文摘要，研究背景、动机、贡献

2. 系统方案陈述

3. 预备知识

4. 系统实现

5. 实验、安全性证明

6. 论文讨论



1.1 论文概要

问题：

- 在资源受限的设备上部署卷积神经网络（CNN）推理是工业物联网（IIoT）面临的一个重大挑战。
- 尽管云计算在机器学习、培训和预测方面显示出巨大的前景，但将数据外包到远程云总是会带来隐私风险和高延迟。

贡献：

我们设计了一个新的基于云-边缘-客户端协作的高效和隐私保护的CNN推理框架（称为 $PCNN_{cec}$ ）。

- 云模型和IIoT中客户端数据被分成两个秘密共享，发送到两个非共谋边缘服务器，离线三元组委托给云生成，不引入额外可信方，避免两个边缘服务器生成三元组的交互以加快离线计算。
- 提出了一种新的基于加性秘密共享技术的高效私有比较协议，该协议可用于在半诚实对手模型中，实现无需近似的ReLU函数的安全计算。



1.2 作者信息

Authors: Jing Wang, Debiao He, Aniello Castiglione, Brij B. Gupta, Marimuthu Karuppiah, Libing Wu

Title: PCNN_{CEC}: Efficient and Privacy-Preserving Convolutional Neural Network Inference Based on Cloud-Edge-Client Collaboration

Authors: Qi Feng, Debiao He, Zhe Liu, Huaqun Wang, and Kim-Kwang Raymond Choo

Title: SecureNLP: A system for multi-party privacy-preserving natural language processing

Journal: IEEE Transactions on Information Forensics and Security



武汉大学密码学与区块链技术实验室
WUHAN UNIVERSITY CRYPTOGRAPHY AND BLOCKCHAIN TECHNOLOGY LABORATORY



1.3 研究背景

1) 应用角度 (IIOT+ML+边缘计算) :

工业物联网(IIOT)[1], [2]通过收集、分析工业数据和持续自优化模型构建，提高了决策和控制能力，这意味着机器学习对于IIOT越来越重要，如何部署仍面临问题。

尽管云在数据建模和机器学习方面具有巨大优势，但它通常无法支持制造、运输和智能建筑行业等应用所需的实时数据收集或分析。因此，有一种趋势是借助边缘计算来增强云能力[3], [4],根据加纳2018年的报告，未来四年内将有75%的企业生成数据在边缘处理。

- [1] B. Chen, J. Wan, Y. Lan, M. Imran, D. Li, and N. Guizani, "Improving cognitive ability of edge intelligent iiot through machine learning," *IEEE Network*, vol. 33, no. 5, pp. 61–67, 2019.
- [2] P. C. M. Arachchige, P. Bertok, I. Khalil, D. Liu, S. Camtepe, and M. Atiquzzaman, "A trustworthy privacy preserving framework for machine learning in industrial iot systems," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 9, pp. 6092–6102, 2020.
- [3] H. Yao, P. Gao, P. Zhang, J. Wang, C. Jiang, and L. Lu, "Hybrid intrusion detection system for edge-based iiot relying on machine-learning-aided detection," *IEEE Network*, vol. 33, no. 5, pp. 75–81, 2019.
- [4] W. Dou, X. Zhao, X. Yin, H. Wang, Y. Luo, and L. Qi, "Edge computing-enabled deep learning for real-time video optimization in iiot," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 4, pp. 2842–2851, 2020.



1.3 研究背景

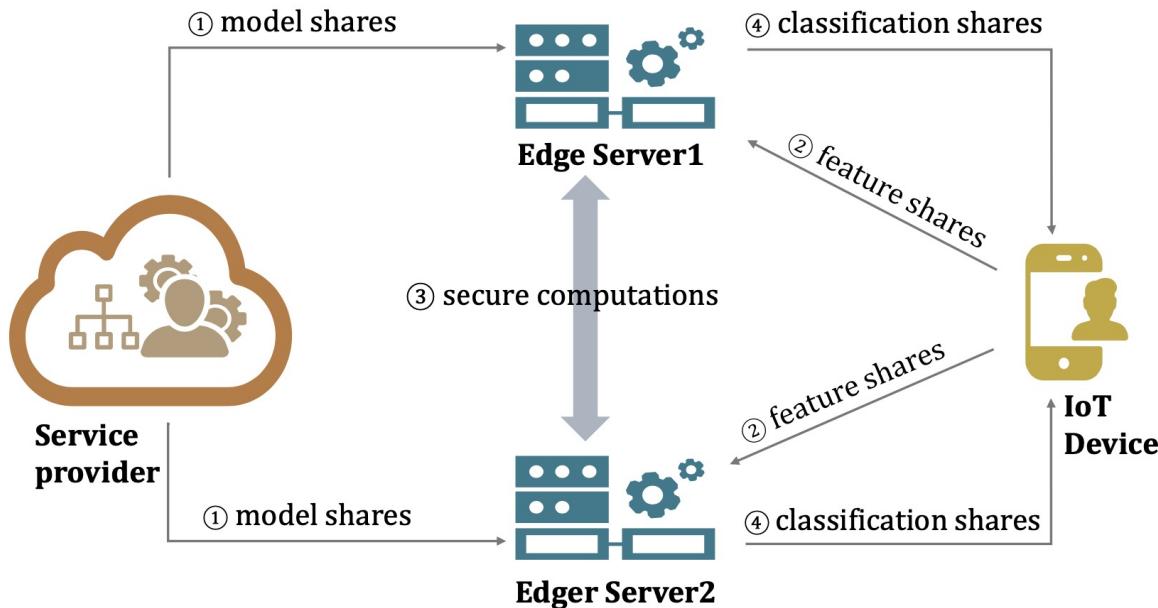
2) 方案角度 (CNN+边缘计算框架+安全协议) :

卷积神经网络（**CNN**）作为一种深度学习算法，被证明在图像和视频识别方面具有巨大优势，因此在IIoT中得到了广泛应用。然而，将模型和数据外包到**不完全可信的边缘服务器**会引起严重的隐私和安全问题。

- **同态加密**（**HE**）密码系统，该系统允许对密文进行线性计算。然而，他不支持非线性计算，因此一些基于**HE**的方案必须将非线性函数转换为近似次多项式，这可能会降低预测精度。
- **安全多方计算**（**MPC**）技术，可分为三类：
 - 基于低延迟的乱码电路（**GC**）技术，这导致恒定的循环但高的通信开销；
 - 另一种是基于秘密共享（**SS**）技术，用于高吞吐量，但比基于**GC**的类别具有更多的交互轮次。
 - 混合协议框架，其中基于**SS**计算线性函数，使用**GC**计算非线性函数（如**CNN**中的**ReLU**激活）。但是，现实中**SS**共享和**GC**共享之间的**转换会产生额外的通信成本**。一些**非线性功能的方案**需要假设存在可信第三方参与协议执行，**牺牲安全性**。

1.4 论文动机

本文设计了一种新的基于云-边缘-客户端协作的隐私保护卷积神经网络推理 ($PCNN_{cec}$)，用于IoT。在我们的方案中，云代表持有模型并提供预测服务的服务提供商，私有预测计算由两个非共谋的边缘服务器执行。





1.4 论文动机

模型	安全协议	线性操作与非线性操作的实现
[30-32]	FHE	不能执行非线性操作
CrypteoNets	FHE	不能执行非线性操作
CryptoDL、E2DM	FHE	高次多项式近似线性操作，精度不足
[33-35]	HE	假设两个非共谋服务器
SecureML [36]	MPC	两个非共谋服务器，用GC+OT实现非线形操作
ABY1、2、3 [24-26]	MPC	使用SS实现线性操作，GC实现非线性操作

- [30] J. W. Bos, K. Lauter, “Private predictive analysis on encrypted medical data,”
- [31] X. Sun, P. Zhang, “Private machine learning classification based on fully homomorphic encryption,”
- [32] S. Park, J. Lee, “Security- preserving support vector machine with fully homomorphic encryption,”
- [33] L. Liu, R. Chen, “Towards practical privacy-preserving decision tree training and evaluation in the cloud,”
- [34] J. Xiong, R. Bi, “Towards lightweight, privacy-preserving cooperative object classification for connected autonomous vehicles”
- [35] J. Wang, L. Wu, “An efficientand privacy-preserving outsourced support vector machine training for internet of medical things,”
- [36] P. Mohassel and Y. Zhang, “Secureml: A system for scalable privacy- preserving machine learning,”
- [24] D. Demmler, T. Schneider, “Aby-a framework for efficient mixed-protocol secure two-party computation.,”
- [25] A. Patra, T. Schneider, “Aby2. 0: Improved mixed-protocol secure two-party computation,”
- [26] P. Mohassel and P. Rindal, “Aby3: A mixed protocol framework for machine learning,”



1.4 论文动机

- MiniONN[38]，安全的基于两方的不经意神经网络预测方案，**GC**来实现非线性运算，效率并没有得到很大提高。
- XONN[18]，它用**GC**的XNOR门操作代替矩阵乘法，以在不牺牲精度的情况下**最小化GC的运行时间**。
- EzPC[39]，第一个结合算术和布尔电路的安全两方计算协议,基于这两种电路实现隐私保护计算

然而，Li等人[21]指出，MiniONNN、XONN和EzPC 无法保护模型隐私。此外，它们大多使用近似多项式来代替神经网络中的激活函数，或者不为非线性计算提供具体的协议。

- CrypTFlow2 [20]，设计了一个新的私有ReLU协议，与基于**GC**的私有ReLU协议相比，提高了效率，**但我们的实验结果表明，该协议不够有效。**

[38] J. Liu, M. Juuti, “Oblivious neural network predictions via minionn transformations,”

[18] M. S. Riazi, M. Samraghi, , “XONN: Xnor-based oblivious deep neural network inference,”

[39] N. Chandran, D. Gupta, “Ezpc: programmable, efficient, and scalable secure two-party computation for machine learning,”

[21] M. Li, S. S. Chow, “Optimizing privacy-preserving outsourced convolutional neural network predictions,”

[20] D. Rathee, M. Rathee, “Cryptflow2: Practical 2-party secure inference,”



1.5 作者贡献

- 基于云-边缘-客户端框架设计了新的隐私推理。计算密集操作转移到两个边缘服务器，离线三元组生成任务被委托给云。与现有方案相比，不引入额外的可信方，并通过避免两个边缘服务器之间生成三元组的交互来加快离线计算。
- 提出了一种新的两方比较隐私计算协议 $\pi_{PrivComp}^{2p}$ ，用来实现安全的非线性操作 ReLU 和安全的 Maximum Pooling，使得在 CNN 中不做任何近似所提出的 $PCNN_{cec}$ 可以达到明文相同的精度。实验结果表明在批处理大小大于 100 的情况下并行运行时，特别是在广域网设置中，具有低延迟和高吞吐量。
- 给出了具体的 $PCNN_{cec}$ 安全性分析，并在 MINIST、CIFAR-10 两个数据集上部署，实验表明在两个数据集上 $PCNN_{cec}$ 比之前的工作更有效。



目录

1. PCNNcec 论文摘要，研究背景、动机、贡献

2. 系统方案陈述

3. 预备知识

4. 系统实现

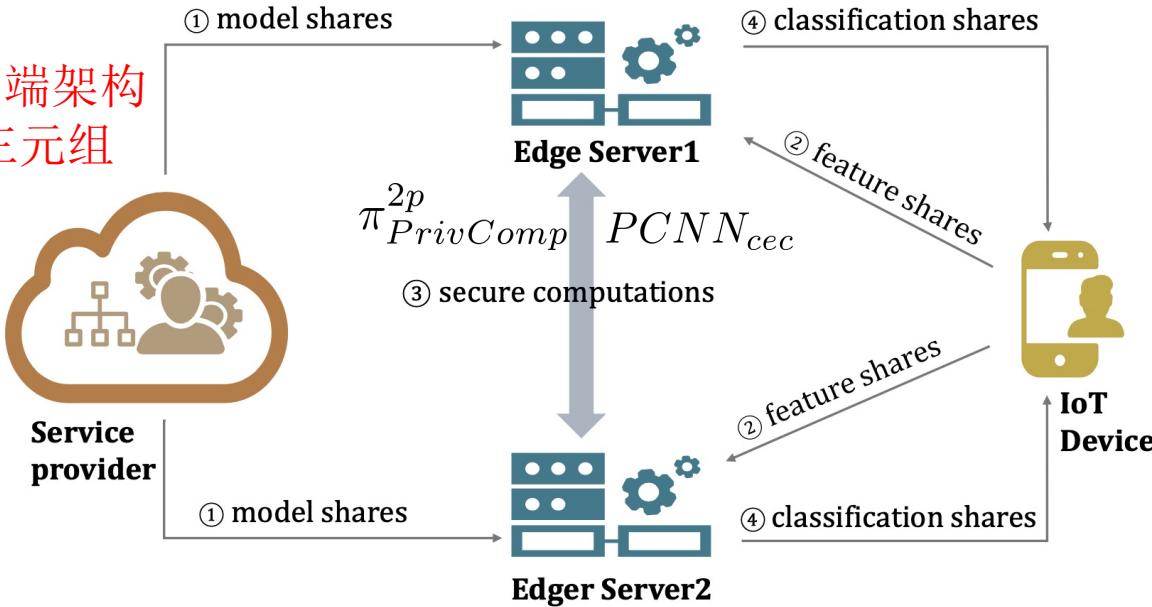
5. 实验、安全性证明

6. 论文讨论

2.1 系统整体模型、威胁模型

2. 提出新协议，完成安全推理网络

1. 云-边缘-客户端架构
云生成离线三元组



威胁模型：半诚实

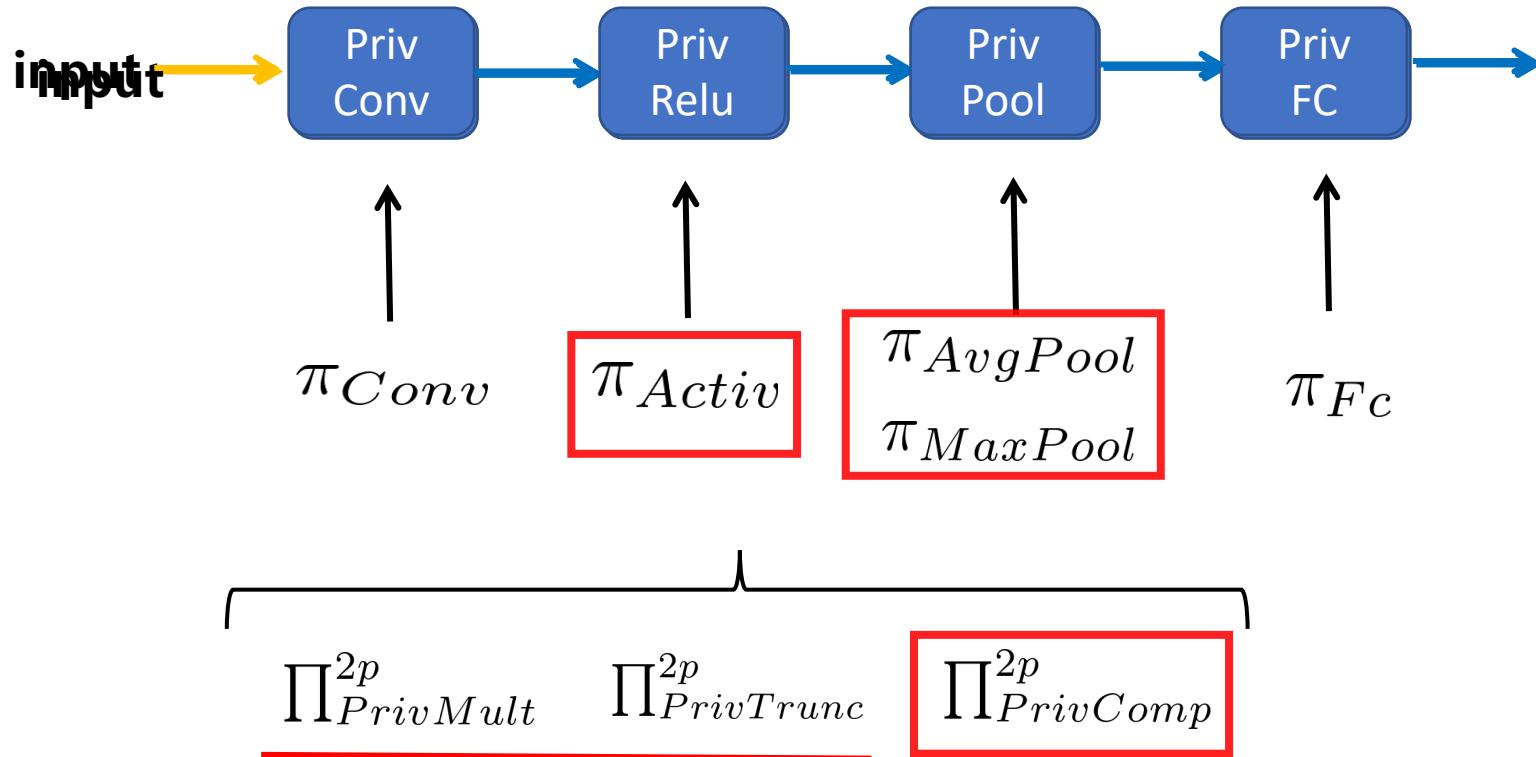
我们假设面向区域的边缘服务器已经与服务提供商（即模型所有者）和物联网设备（即数据所有者）进行了身份验证，并且他们将诚实地执行用于隐私保护CNN推断的所有安全两方计算。



2.2 设计目标

- **正确性:** 边缘服务器协作执行两方计算，保证CNN预训练模型得到正确的预测结果
- **安全性:** 防止诚实但好奇的任一方得到对手的数据
- **效率:** 减少实体间的计算、通信开销

2.3 方案的算法定义



- [22] K. Huang, X. Liu, “A lightweight privacy-preserving cnn feature extraction framework for mobile sensing,” IEEE Transactions on Dependable and Secure Computing, 2019.
- [36] P. Mohassel and Y. Zhang, “Secureml: A system for scalable privacy- preserving machine learning,” in Proc. of IEEE S&P, 2017.



目录

1. PCNNcec 论文摘要，研究背景、动机、贡献

2. 系统方案陈述

3. 预备知识

4. 系统实现

5. 实验、安全性证明

6. 论文讨论

3.1 预备知识

加法秘密共享 (Additive Secret Sharing) :

1) 加法

可信方生成随机数 r , 令 $\langle x \rangle_0 = r \pmod{Z2I}$, $\langle x \rangle_1 = x - r \pmod{Z2I}$ 。
其中 r 是从环 $Z2I$ 中随机选取的。因此它与 $Z2I$ 的任何元素都是不可区分的分布,是随即均匀的。





3.2 方案的构造基础

2) 乘法 (Beaver乘法三元组)

- 可信方生成 a,b,c 且满足 $a*b=c$ (a,b,c 对两个参与 $p1,p2$ 方保密)

具体：可信方生成 $[a]0, [a]1, [b]0, [b]1, [c]0$ 五个数字

$$a=[a]0+[a]1 \quad b=[b]0+[b]1 \quad c=a*b \quad c1=c-[c]0$$

计算 $x*y$	P1	P2
初始	$x0,y0$	$x1,y1$
	$x0,y0,[a]0,[b]0,[c]0$	$x1,y1,[a]1,[b]1,[c]1$
计算	$[\alpha]0=[x]0-[a]0$ $[\beta]1=[y]0-[b]0$	$[\alpha]1=[x]1-[a]1$ $[\beta]1=[y]1-[b]1$
通信	各自公开 $[\alpha]i, [\beta]i$, 得到 α, β	各自公开 $[\alpha]i, [\beta]i$, 得到 α, β
计算	$[z]0=[c]0+\alpha*[b]0+\beta*[a]0$	$[z]1=[c]1+\alpha*[b]1+\beta*[a]1 +\alpha*\beta$



3.2 方案的构造基础

验证：

$$=[c]1+(x-a)*[b]1+(y-b)*[a]1+(x-a)(y-b)$$

$$=[c]1+x*[b]1-a*[b]1+y*[a]1-b*[a]1+x*y-a*x-b*y+a*b$$

$$[z]0+z[1]=c+xb-ab+ya-ba+xy-ay-bx+ab$$

$$=xy$$



目录

1. PCNNcec 论文摘要，研究背景、动机、贡献

2. 系统方案陈述

3. 预备知识

4. 系统实现

5. 实验、安全性证明

6. 论文讨论

4.1 building blocks

1) 安全乘法 $\Pi_{PrivMult}^{2p}$

Protocol $\Pi_{privMult}^{2P}$

Input: \mathcal{ES}_1 holds the secret shares $(\llbracket x \rrbracket_1, \llbracket y \rrbracket_1)$ and the share of Beaver's triplet $\llbracket \text{Tri} \rrbracket_1$.

\mathcal{ES}_2 holds the secret shares $(\llbracket x \rrbracket_2, \llbracket y \rrbracket_2)$ and the share of Beaver's triplet $\llbracket \text{Tri} \rrbracket_2$.

Output: \mathcal{ES}_1 gets $\llbracket z \rrbracket_1$; \mathcal{ES}_2 gets $\llbracket z \rrbracket_2$

At the \mathcal{ES}_i 's side:

- 1) Computes $\llbracket u \rrbracket_i = \llbracket x \rrbracket_i - \llbracket \Delta a \rrbracket_i$, $\llbracket v \rrbracket_i = \llbracket y \rrbracket_i - \llbracket \Delta b \rrbracket_i$;
- 2) Sends $\{\llbracket u \rrbracket_i, \llbracket v \rrbracket_i\}$ to \mathcal{ES}_{3-i} ;
- 3) Receives $\{\llbracket u \rrbracket_{3-i}, \llbracket v \rrbracket_{3-i}\}$ from \mathcal{ES}_i , and the computes $u = \llbracket u \rrbracket_1 + \llbracket u \rrbracket_2$ and $v = \llbracket v \rrbracket_1 + \llbracket v \rrbracket_2$;
- 4) Outputs $\llbracket z \rrbracket_i$ locally by computing $\llbracket z \rrbracket_i = \llbracket x \rrbracket_i \times v + \llbracket y \rrbracket_i \times u + (i-1) \times u \times v + \llbracket \Delta c \rrbracket_i$;

Fig. 2: Private multiplication protocol $\Pi_{privMult}^{2P}$ [22]



4.1 building blocks

2) 安全截断 $\Pi_{PrivTrunc}^{2p}$

CNN推理的计算实际上是在浮点数上进行的，而秘密共享技术通常是在属于环（或域）的整数上进行的。因此，有必要将所有浮点数转换为 Z_2^l 的“整数”，要对每个浮点数进行缩放。

转化为定点数后， l_D 位小数变为定点数后 l 位数字。同样的，
 $\Pi_{PrivMul}^{2p}$ 操作也在定点数上进行，但结果的小数部分就会扩大 $2x l_D$ 位，这时必须将后 l_D 位截断(Truncate)，以保证结果为 l 位小数

通过右移 l 位的方式进行截断，其错误的概率是可以忽略不计的，只会导致 2^{-l_D} 大小的误差。



4.1 building blocks

3) 安全比较 $\Pi_{PrivComp}^{2p}$

$$f(u, v) = 1 , \quad u > v$$

$$f(u, v) = 0 , \quad u \leq v.$$

问题:

- 一种方法是使用GC计算，但将SS的share转化为GC的形式将会花费高额的带宽
- 第二种方案基于安全位提取协议(Huang等人[22]提出)，但该协议需要(2l-1)二进制字段上的安全乘法，时间开销很大。
- 第三种方案是安全的确定中间结果的符号 $z=r * (u-v)$, 但当 $u=v$ 时，中间结果z的符号只需要至少一方就可以拿到

为了解决上述问题，提出了新的安全比较协议 $\Pi_{PrivComp}^{2p}$



4.1 building blocks

3) 安全比较 $\prod_{PrivComp}^{2p}$

4.2 PrivConv

安全的卷积运算:

Algorithm 1 Private Convolution Computation π_{Conv}

Input: Each edge server $\mathcal{E}\mathcal{S}_i$ inputs the volume shares $\{\llbracket \mathbf{X}^j \rrbracket_i\}_{j=0}^{d-1}$, the kernel shares $\{\llbracket \mathbf{w}^j \rrbracket_i\}_{j=0}^{d-1}$ and the triplet shares $\llbracket \text{TRI} \rrbracket_i$.

Output: $\mathcal{E}\mathcal{S}_i$ gets its own result share $\{\llbracket \mathbf{Z}^j \rrbracket_i\}_{j=0}^{d-1}$.

- 1: **for** each volume $j \in \{0, \dots, d-1\}$ **do**
- 2: **for** each vector $k \in \{0, \dots, m-1\}$ **do**
- 3: $\mathcal{E}\mathcal{S}_i$ computes the vector $\llbracket \mathbf{u}_k^j \rrbracket_i = \llbracket \mathbf{x}_k^j \rrbracket_i - \llbracket \Delta \mathbf{a}_k^j \rrbracket_i$
 and the vector $\llbracket \mathbf{v}_k^j \rrbracket_i = \llbracket \mathbf{w}_k^j \rrbracket_i - \llbracket \Delta \mathbf{b}_k^j \rrbracket_i$;
- 4: **end for**
- 5: **end for**
- 6: $\mathcal{E}\mathcal{S}_i$ sends a sequence of $\{\llbracket \mathbf{u}_k^j \rrbracket_i, \llbracket \mathbf{v}_k^j \rrbracket_i\}_{j=0, k=0}^{d-1, m-1}$ to the other edge server $\mathcal{E}\mathcal{S}_{3-i}$ simultaneously. In return, it receives a sequence of $\{\llbracket \mathbf{u}_k^j \rrbracket_{3-i}, \llbracket \mathbf{v}_k^j \rrbracket_{3-i}\}_{j=0, k=0}^{d-1, m-1}$;
- 7: **for** each volume $j \in \{0, \dots, d-1\}$ **do**
- 8: **for** each vector $k \in \{0, \dots, m-1\}$ **do**
- 9: $\mathcal{E}\mathcal{S}_i$ computes $\mathbf{u}_k^j = \llbracket \mathbf{u}_k^j \rrbracket_1 + \llbracket \mathbf{u}_k^j \rrbracket_2$, $\mathbf{v}_k^j = \llbracket \mathbf{v}_k^j \rrbracket_1 + \llbracket \mathbf{v}_k^j \rrbracket_2$ and $\llbracket \tilde{z}_k^j \rrbracket_i = \llbracket \mathbf{x}_k^j \rrbracket_i \cdot \mathbf{u}_k^j + \llbracket \mathbf{w}_k^j \rrbracket_i \cdot \mathbf{v}_k^j + \llbracket \Delta \mathbf{c}_k^j \rrbracket_i \cdot \mathbf{1}_k^j + (2-i) \mathbf{u}_k^j \cdot \mathbf{v}_k^j$;
- 10: $\mathcal{E}\mathcal{S}_i$ executes the truncation operation for each $\llbracket \tilde{z}_k^j \rrbracket_i$ and gets $\llbracket z_k^j \rrbracket_i$ following the sub-protocol $\Pi_{\text{privTrunc}}^{2P}$
- 11: **end for**
- 12: **end for**
- 13: $\mathcal{E}\mathcal{S}_i$ re-arranges the vector $\{\llbracket z_0^j \rrbracket_i, \dots, \llbracket z_{m-1}^j \rrbracket_i\}$ (for each j in range d) be a new matrix as the next layer's input volume $\llbracket \mathbf{Z}^j \rrbracket_i$;
- 14: **return** the result share $\{\llbracket \mathbf{Z}^j \rrbracket_i\}_{j=0}^{d-1}$.



4.3 Priv_x0008_Activ

安全的激活函数：

Relu函数公式： $y = \text{relu}(x) = \max\{x, 0\} = (x > 0 ? 1 : 0)^* x$

$$\prod_{PrivComp}^{2p} \prod_{PrivMult}^{2p}$$



4.3 PrivActiv

1) 安全的max函数

$$\max\{x, y\} = \frac{x + y}{2} + \frac{|x - y|}{2} = \begin{cases} \frac{x + y + (x - y)}{2} = x, & x > y \\ \frac{x + y + (y - x)}{2} = y, & x \leq y \end{cases}$$

Algorithm 2 Private Maximum Computation π_{Max}

Input: Each edge server inputs its share $\llbracket x \rrbracket_i, \llbracket y \rrbracket_i$ and the triplet shares $\llbracket \text{TRI} \rrbracket_i$.

Output: Each edge server gets its result share $\llbracket z \rrbracket_i$.

- 1: The two edge servers \mathcal{ES}_1 and \mathcal{ES}_2 jointly runs the sub-protocol $\Pi_{\text{privComp}}^{2P}$, where the inputs are $\{\llbracket x \rrbracket_1, \llbracket y \rrbracket_1\}, \{\llbracket x \rrbracket_2, \llbracket y \rrbracket_2\}$ and the outputs are $[f]_1$ and $[f]_2$ respectively And let $\llbracket u \rrbracket_1 := [f]_1, \llbracket u \rrbracket_2 := [f]_2$ respectively;
 - 2: \mathcal{ES}_1 and \mathcal{ES}_2 jointly run the sub-protocol $\Pi_{\text{privMult}}^{2P}$, where the inputs are $\{(-1)^{\llbracket u \rrbracket_1} \llbracket x - y \rrbracket_1, (-1)^{\llbracket u \rrbracket_1}\}, \{(-1)^{\llbracket u \rrbracket_2} \llbracket x - y \rrbracket_2, (-1)^{\llbracket u \rrbracket_2}\}$ and the outputs are $\llbracket v \rrbracket_1, \llbracket v \rrbracket_2$ respectively;
 - 3: \mathcal{ES}_i locally computes $\llbracket z \rrbracket_i = \llbracket x \rrbracket_i - \frac{1}{2} \llbracket v \rrbracket_i$.
-

$$= x - \frac{1}{2}v = x - \frac{1}{2}[x + (-1)^u x]$$

如果 $y-x>0$: $x - \frac{1}{2}(2x) = 0$;

else : $x - \frac{1}{2} \times 0 = x$.



4.3 PrivActiv

2) 安全的Relu函数

Algorithm 3 Private Activation Computation with ReLU function π_{Acitv}

Input: Each edger server inputs its share $\llbracket x \rrbracket_i$ and the triplet shares $\llbracket \text{TRI} \rrbracket_i$.

Output: Each edger server gets its result share $\llbracket y \rrbracket_i$.

- 1: The two edge servers \mathcal{ES}_1 and \mathcal{ES}_2 jointly generate a pair of random values $(\llbracket r \rrbracket_1, \llbracket r \rrbracket_2)$ satisfying that $r = 0$.
 - 2: The two edge servers \mathcal{ES}_1 and \mathcal{ES}_2 jointly runs the private maximization algorithm π_{Max} as the Algorithm 2 defined, where the inputs are $\{\llbracket x \rrbracket_1, \llbracket r \rrbracket_1\}$, $\{\llbracket x \rrbracket_2, \llbracket r \rrbracket_2\}$ and the outputs are $\llbracket z \rrbracket_1$ and $\llbracket z \rrbracket_2$ respectively;
 - 3: \mathcal{ES}_i locally learns its output $\llbracket y \rrbracket_i := \llbracket z \rrbracket_i$.
-



4.3 PrivPool

2) 安全的池化

- 平均池化：两方只需要在本地用share做平均就好了
- 最大池化：

Algorithm 4 Private Maximum Pooling Computation

π_{MaxPool}

Input: Each edger server inputs its shares $\llbracket \mathbf{x} \rrbracket_i = \{\llbracket x \rrbracket_i\}_{j=0}^{n^2-1}$ and the triplet shares $\llbracket \text{TRI} \rrbracket_i$.

Output: Each edger server gets its result share $\llbracket y \rrbracket_i$.

- 1: Each edge server \mathcal{ES}_i first lets $\llbracket y \rrbracket_i := \llbracket x_0 \rrbracket_i$, then it jointly runs the following steps with \mathcal{ES}_{3-i} ;
 - 2: **for** each $j \in \{1, 2, \dots, n^2 - 1\}$ **do**,
 - 3: \mathcal{ES}_1 and \mathcal{ES}_2 jointly run the algorithm π_{Max} , where the inputs are $\{\llbracket y \rrbracket_1, \llbracket x_j \rrbracket_1\}$, $\{\llbracket y \rrbracket_2, \llbracket x_j \rrbracket_2\}$ and the intermediate outputs are $\llbracket z \rrbracket_1$ and $\llbracket z \rrbracket_2$ respectively;
 - 4: \mathcal{ES}_i lets $\llbracket y \rrbracket_i := \llbracket z \rrbracket_i$;
 - 5: **end for**
 - 6: \mathcal{ES}_i locally learns its final output $\llbracket y \rrbracket_i$.
-

4.4 完整的系统

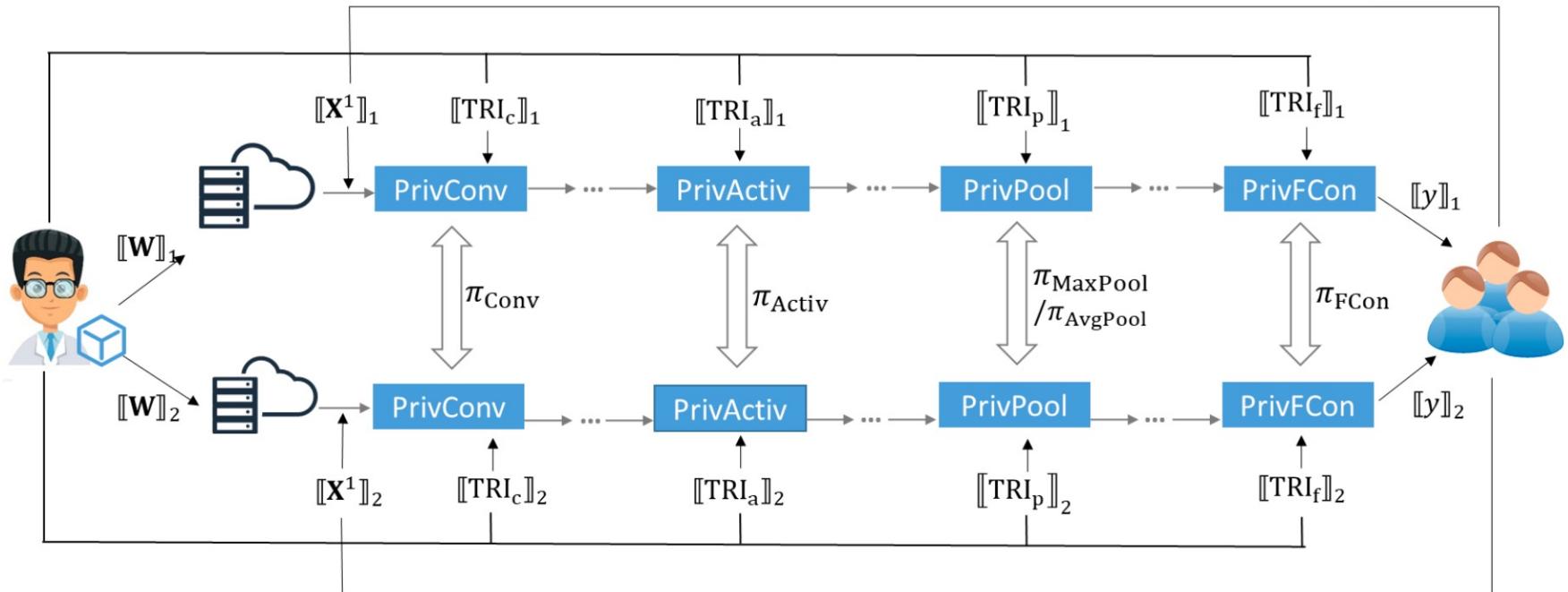


Fig. 4: Framework of the proposed scheme



目录

1. PCNNcec 论文摘要，研究背景、动机、贡献

2. 系统方案陈述

3. 预备知识

4. 系统实现

5. 实验、安全性证明

6. 论文讨论



5.1 安全性证明

- 根据我们提出的模型，移动用户的数据和服务提供商的模型参数都被随机共享，并独立分布到两个非共谋边缘服务器，**模型和用户数据的隐私**得到了保护。
- 在预测过程中，两个边缘服务器总是在Beaver的三元组的帮助下处理它们及其在秘密共享上的中间值。虽然**三元组是由服务提供商生成的**，但仍然不能侵犯用户的隐私，因为服务提供商不涉及以下预测过程，边缘服务器只能知道自己的三元组。

Theorem 1. *The protocol $\Pi_{privComp}^{2P}$ is secure in the presence of semi-honest adversaries.*

Theorem 2. *The private convolution computation algorithm π_{Conv} is secure in the presence of semi-honest adversaries.*

Theorem 3. *The private activation computation algorithm π_{Activ} is secure in the presence of semi-honest adversaries.*

Theorem 4. *The private pooling computation algorithm $\pi_{AvgPool}$ / $\pi_{MaxPool}$ is secure in the presence of semi-honest adversaries.*

Theorem 5. *The private fully connected computation algorithm π_{FCon} is secure in the presence of semi-honest adversaries.*



5.2 实验

为了证明PCNCEC的性能，我们将我们的方案与几个类似的工作进行了比较，包括 MiniONN[38]、Huang等人的方案[22]、Li等人的方案[21]和CryptoFlow2[20]。

1) 隐私保护比较函数

TABLE 2: Comparison of communication with prior works
for private comparison protocol $\Pi_{\text{privComp}}^{2P}$

Works	Comm. (bits)	Rounds
MiniONN [38]	$4\lambda l$	2
Huang et al. [22]	$8l - 4$	l
Li et al. [21]	—	—
CryptFlow2 [20]	$\lambda l + 14l$	$\log l$
This work (Complete, i.e., DReLU)	$16l$	5
This work (Simplified)	$12l$	3

The parameter l denotes the length of an integer in the ring \mathbb{Z}_{2^l} , while λ is the security parameter and typically 128

GC
比特分解

- 我们的私有比较协议的通信成本略高于 Huang等人的方案，但低于其他协议。

5.2 实验

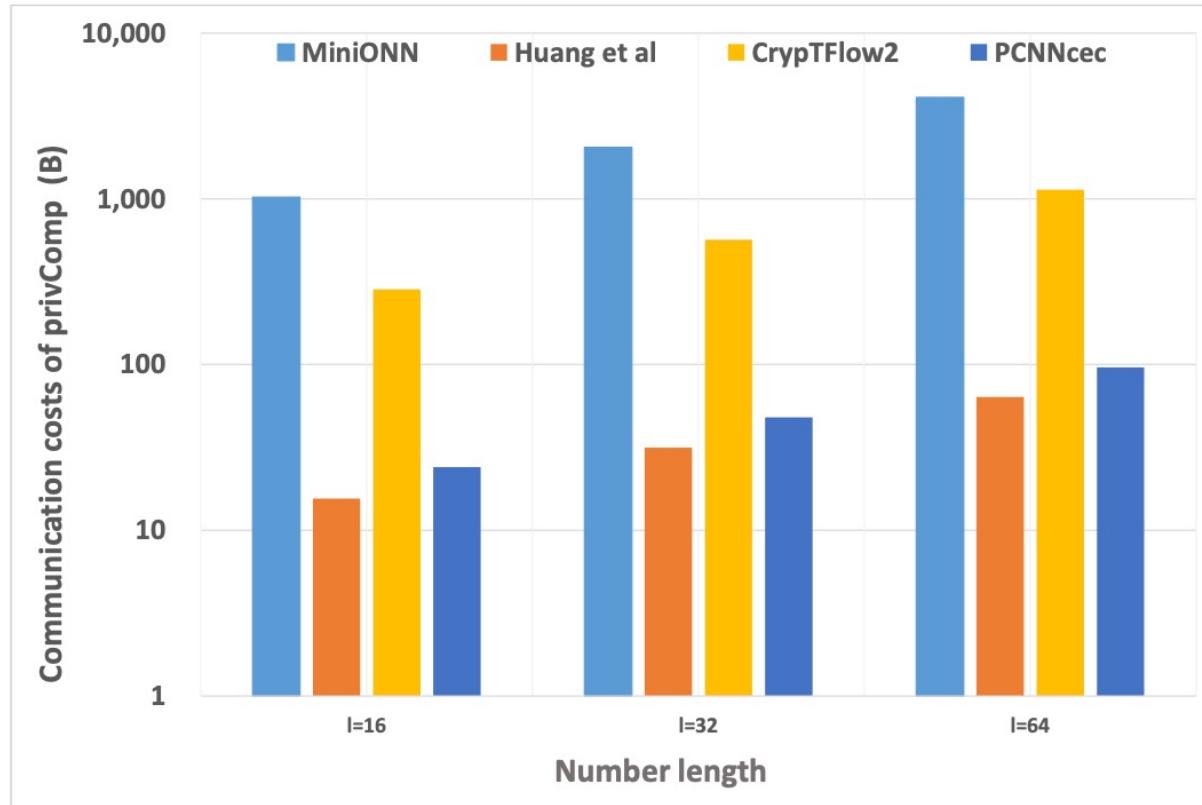


Fig. 6: Comparison of bandwidths for privacy-preserving comparison protocols

- 我们可以观察到，我们的私有比较协议的通信成本略高于Huang等人的协议，但低于其他协议。

5.2 实验

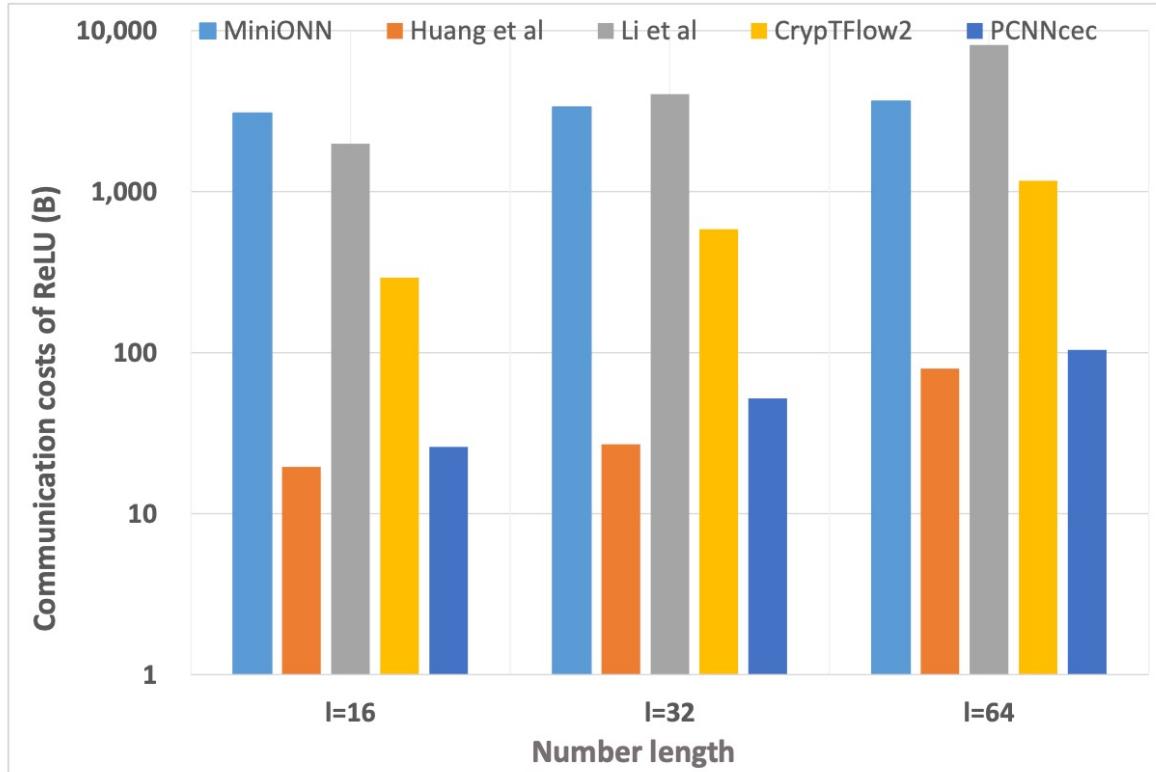


Fig. 8: Comparison of communication costs with different l

- 我们可以观察到，我们的私有Relu协议的通信成本略高于Huang等人的协议，但低于其他协议。

5.2 实验

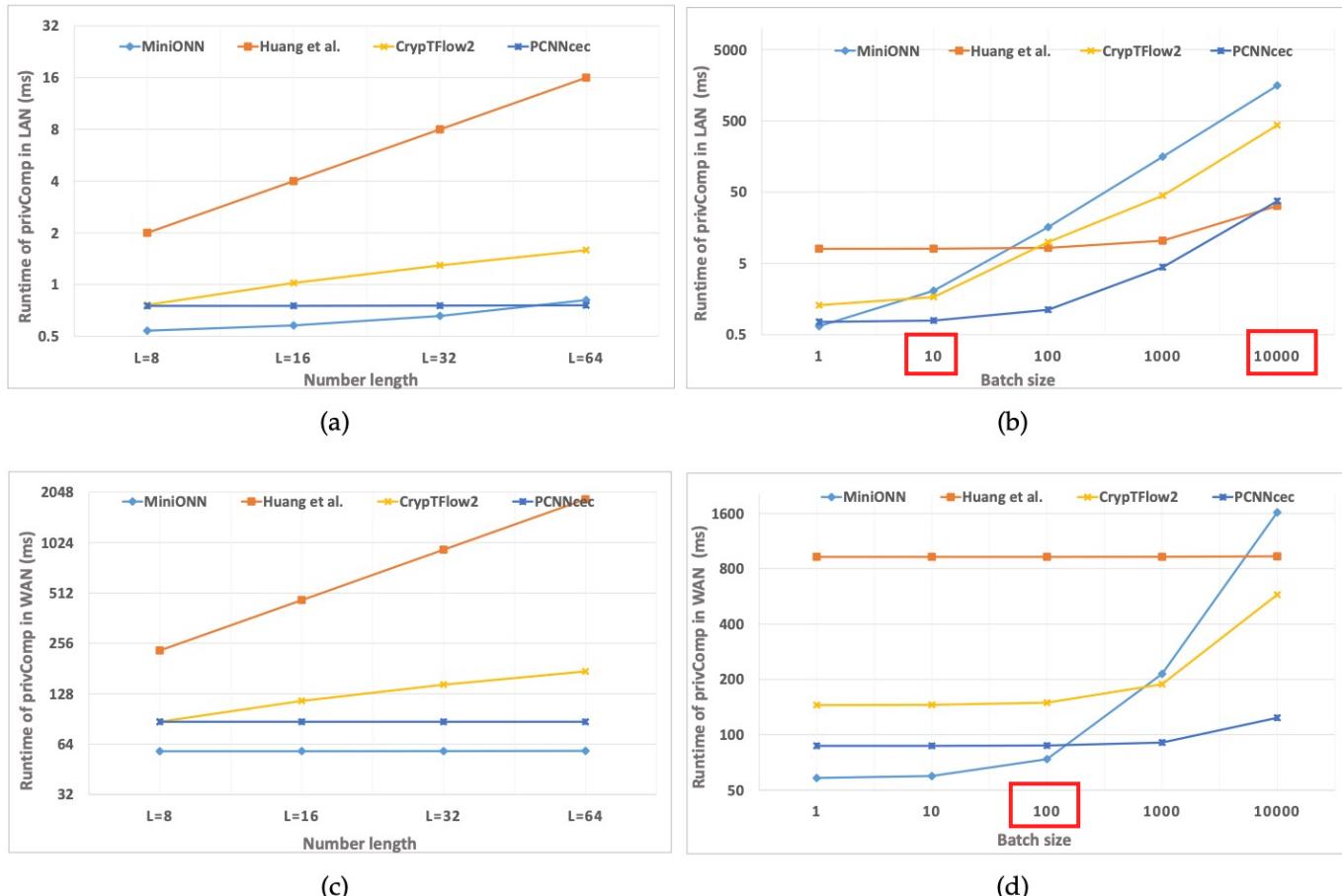


Fig. 7: Runtime of privacy-preserving comparison protocols: (a)Runtime for different bit-width l in LAN; (b)Runtime for different parallel batch size bs in the LAN setting with $l = 32$; (c)Runtime for different bit-width l in WAN; (d)Runtime for different parallel batch size bs in the WAN setting with $l = 32$

5.2 实验

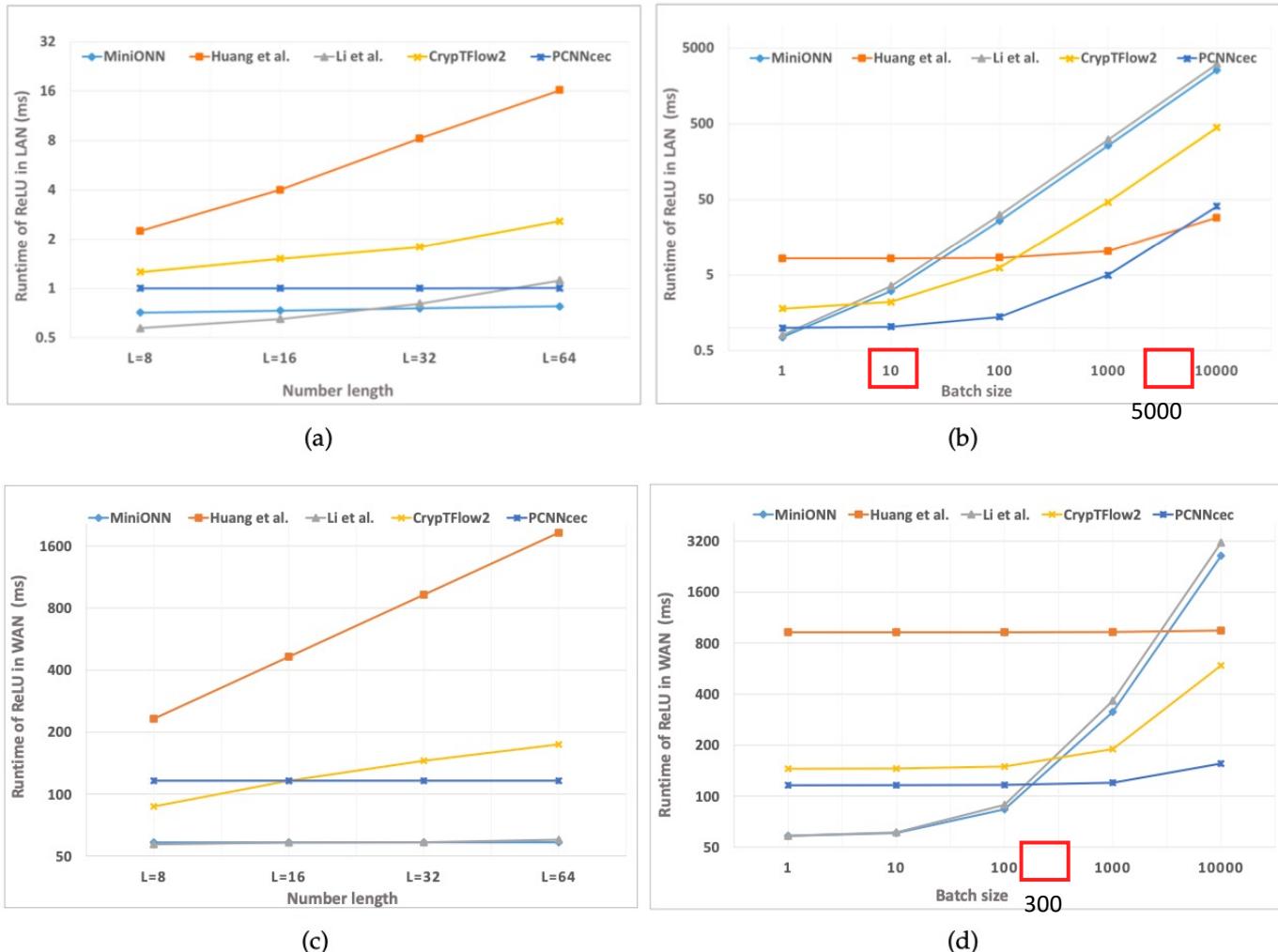


Fig. 9: Performance of private ReLU: (a)Runtime for different bit-width l in LAN; (b)Runtime for different parallel batch size bs in the LAN setting with $l = 32$; (c)Runtime for different bit-width l in WAN; (d)Runtime for different parallel batch size bs in the WAN setting with $l = 32$

5.2 实验

Works	Comm. (bits)	Rounds
MiniONN [38]	$(k - 1)(18\lambda\eta - 6\lambda)$	2
Huang et al. [22]	$(k - 1) \cdot (10l - 4)$	$(l + 1) \cdot \lceil \log k \rceil$
Li et al. [21]	—	—
CRYPTFLOW2 [20]	$(k - 1) \cdot (\lambda l + 14l)$	$\log l \cdot \lceil \log k \rceil$
This work	$12(k - 1)l$	$4 \lceil \log k \rceil$

The parameter l denotes the length of an integer in the ring \mathbb{Z}_{2^l} , k denote the size of a pool filter, while λ is the security parameter and typically 128

- 我们的私有比较协议的通信成本略高于 Huang 等人的方案，但低于其他协议。



5.2 实验

TABLE 6: Performance of each layer on the MNIST dataset

Network I			Network II		
Layers	Latency (s)		Layers	Latency (s)	
	LAN	WAN		LAN	WAN
1.Convolution	1.603×10^{-2}	0.042	1.Convolution	0.082	0.17
2.Square	7.815×10^{-4}	0.03	2.ReLU	0.038	0.189
3.AvgPool	8.449×10^{-8}	9.194×10^{-8}	3.MaxPool	0.027	0.383
4.Square	3.11×10^{-4}	0.029	4.Convolution	0.008	0.045
5.Full-connected	8.604×10^{-4}	0.032	5.ReLU	0.005	0.124
-	-	-	6.MaxPool	0.005	0.238
-	-	-	7.Full-connected	0.008	0.512
-	-	-	8.ReLU	0.001	0.117
-			9.Full-connected	0.0007	0.03
Total	0.018	0.133	Total	0.175	1.808

- 从结果中我们可以观察到，在局域网设置中，每个层的通信成本主导着运行时间。在广域网环境下，通信成本和通信次数都会影响效率



5.2 实验

TABLE 7: Performance of each layer on the CIFAR-10 dataset in Network III

Layers	Latency (s)		Layers	Latency (s)	
	LAN	WAN		LAN	WAN
1.Convolution	0.54	1.109	10.AvgPool	0.00001	0.00001
2.ReLU	0.261	0.636	11.Convolution	0.72	1.469
3.Convolution	11.52	46.109	12.ReLU	0.017	0.149
4.ReLU	0.287	0.812	13.Convolution	0.08	0.189
5.AvgPool	0.00001	0.00001	14.ReLU	0.035	0.123
6.Convolution	2.88	5.789	15.Convolution	0.02	0.113
7.ReLU	0.067	0.346	16.ReLU	0.007	0.215
8.Convolution	2.912	6.011	17.Full-connected	0.004	0.076
9.ReLU	0.089	0.346	Total	19.439	63.492

- 为了进一步证明PCNCEC的效率，我们将其在网络II和网络III中运行时与其他方案进行了比较，因为这两种网络更具代表性。



5.2 实验

TABLE 3: Comparison of security levels for related works

Properties	MiniONN [38]	Huang et al. [22]	Li et al. [21]	CRYPTFLOW2 [20]	PCNN _{CEC}
Data privacy	✓	✓	✓	✓	✓
Model privacy	✗	✗	✓	✓	✓

✓ denotes that the security property can be achieved; ✗ denotes that the scheme does not meet the property.

- 从表3中，我们可以看出Huang等人的方案无法实现模型隐私，因此我们在与真实数据集运行时的效率比较中不考虑它。CryptoFlow2采用不经意传输和同态加密来执行线性矩阵乘法运算，这可能比基于加法算术秘密共享技术的运算成本和通信开销更高。因此，我们在这里主要比较PCNcec与MiniONN和Li等人的方案。

5.2 实验

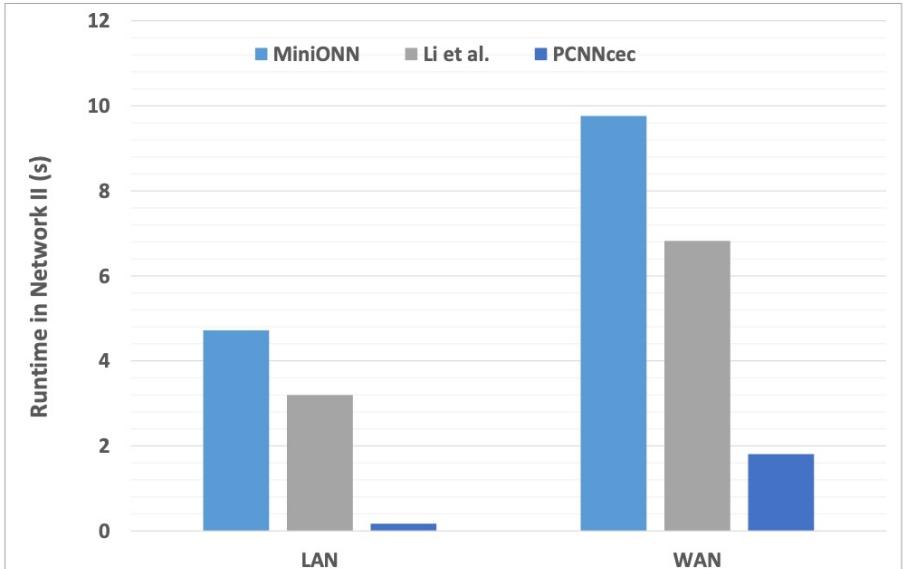


Fig. 10: Comparison of runtime for Network II

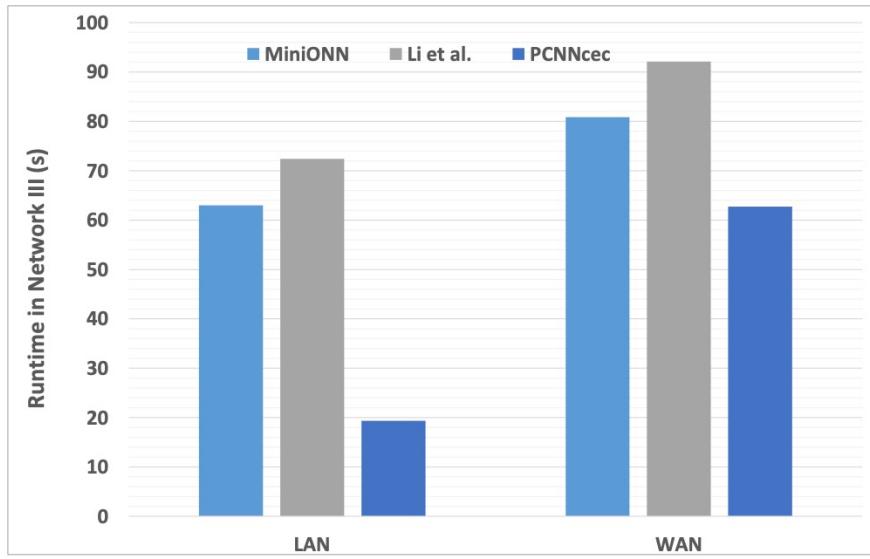


Fig. 11: Comparison of runtime for Network III

- 在具有网络II的MNIST数据集上，与MiniONN、Li等人的方案相比，PCNCEC在局域网设置下的运行时间分别提高了26.38倍和17.59倍，在广域网设置下的性能分别提高了4.39倍和2.77倍。在具有网络III的CIFAR-10数据集上，与MiniONN和Li等人的方案相比，PCNCEC在局域网设置下的运行时间分别提高了2.26倍和2.74倍，在广域网设置下的性能分别提高了1.29倍和1.47倍。因此，PCNCEC在LAN和WAN设置中都可以很好地执行。



目录

1. PCNNcec 论文摘要，研究背景、动机、贡献

2. 系统方案陈述

3. 预备知识

4. 系统实现

5. 实验、安全性证明

6. 论文讨论



6.论文讨论



Thanks for watching