QUOTIENT: Two-Party Secure Neural Network Training and Prediction

本次介绍Agrawal等人在ACM CCS'2019上提出的两方计算下三值量化神经网络的训练和预测方案，该方案针对三值化神经网络对两方计算协议进行了优化，使用秘密分享、OT和GC等技术组合成高效方案。原文链接。

0. 背景知识

对于一般神经网络和2PC方面的知识在这里不做过多赘述，我们把主要精力放在对于三值量化神经网络络上。

0.1 量化函数

本文借鉴了方案WAGE中的量化函数 Q_w, Q_a, Q_g, Q_e 分别将权重 $\mathbf{w}$，激活函数 $\mathbf{a}$，权重梯度 $\mathbf{G}$ 和激活函数梯度 $\mathbf{e}$ 量化为小范围的、定点数有限集合。在WAGE中，所有的量化函数基于如下设计：

- 给定一个精度为p的定点数 $v_{(p)} = v/2^p$，找到距离其最近的另一个精度为q的定点数 $v_{(q)}$ 满足：$v_{(q)} = N(v_{(p)}, q) = \frac{\lfloor \frac{v}{2^p} 2^q \rceil}{2^q}$，其中 $\frac{a}{2^p} 2^q$根据$p > q$是否成立来决定是否是面向$2^{|p-q|}$的乘法或者除法。
- 进一步，WAGE方案引入了saturation function $S(\cdot)$来实现函数 $Q(\cdot)$ 的计算：$v_{(q)} = Q(v_{(p)}, q) = S(N(v_{(p)}, q), q)$，其中 $S(x, q) = min(max(x, -1+2^{-q}), 1-2^{-q})$。

如此，则有如下量化函数：

- $\mathbf{W}_{(p_w)} = Q_w(\mathbf{W}_{(p)}, p_w) = Q(\mathbf{W}_{(p)}, p_w)$;
- $\mathbf{a}_{(p_a)} = Q_a(\mathbf{a}_{(p)}, p_a) = Q(\frac{\mathbf{a}_{(p)}}{2^\alpha}, p_a)$，其中$\alpha$ 是整数，由 $\mathbf{W}$ 的维度决定；
- $\mathbf{e}_{(p_e)} = Q_e(\mathbf{e}_{(p)}, p_e) = Q(\frac{\mathbf{e}_{(p)}}{2^{\mathsf{cpow}(max\{|\mathbf{e}_{(p)}|\})}}, p_e)$，其中$\mathsf{cpow}(x) = 2^{\lfloor \log_2(x) \rceil}$ 是距离 x 最近的2的次幂；
- $\mathbf{G}_{p_g} = Q_g(\mathbf{G}^n_{(p)}, p_g) = \frac{\mathsf{sign}(\mathbf{G}^n_{(p)})}{2^{p_g - 1}}[\lfloor \mathbf{G}^n_{(p)} \rfloor + \mathsf{B}(|\mathbf{G}^n_{(p)}| - \lfloor |\mathbf{G}^n_{(p)}| \rfloor)]$，其中 $\mathbf{G}^n_{(p)} = \eta \mathbf{G}_{(p)} / 2^{\mathsf{cpow}(max\{|\mathbf{G}_{(p)}|\})}$，$\mathsf{B}(p)$ 是伯努利分布采样。

0.2 针对密码学的改造

为了加速安全计算，本文对上述算法进行了针对性的改造：

- 对于 Q_w，令$p_w = 1$ 且 $Q_w = 2Q$，那么 $\mathbf{W} \in \{-1, 0, 1\}$；
- 对于Q_e，令 cpow 替换为 npow $= 2^{\lceil \log_2 x \rceil}$，加速计算；

- 对于Q_g，本文提出了新的计算方法 $\mathbf{G}_g = Q_g(\mathbf{G}_{(p)}, p_g) = N(\frac{\mathbf{G}_{(p)}}{2^{\text{npow}(max\{|\mathbf{G}_{(p)}|\})}}, p_g)$加速计算。

和一般性的浮点数神经网络相比，量化网络在计算完每一层之后多了量化函数的计算，其他流程并无太大区别。

1. 密码学协议

1.1 Ternary-Integer Matrxi-Vector Product

对于线性层，权重$\mathbf{W} \in \{-1, 0, 1\}^{n\times m}$，数据 $\mathbf{a} \in \mathbb{Z}_q^m$。二者的矩阵-向量乘积$\mathbf{z} = \mathbf{W}\mathbf{a}$功能函数如下：

Algorithm 5: Ternary-Integer Matrix-Vector Product

Input: Matrix $\mathbf{W} \in \{-1, 0, 1\}^{n\times m}$ and vector $\mathbf{a} \in \mathbb{Z}_q^m$

$\mathbf{z} = (0)_{i\in[n]}$

for $i \in [n], j \in [m]$ **do**

  if $\mathbf{W}_{\mathbf{i},\mathbf{j}} > 0$ **then** $z_i \mathrel{+}= \mathbf{a}_j$

  if $\mathbf{W}_{\mathbf{i},\mathbf{j}} < 0$ **then** $z_i \mathrel{-}= \mathbf{a}_j$

return $\mathbf{z}$

为了实现其2PC下的高效计算，本文首先将$\mathbf{W}$ 分解为 $\mathbf{W} = \mathbf{W}^+ - \mathbf{W}^-$，如此$\mathbf{W}^+, \mathbf{W}^- \in \{0, 1\}^{n\times m}$。从而进一步基于OT实现$\mathbf{W}^+\mathbf{a}$，最终结果$\mathbf{z} = \mathbf{W}^+\mathbf{a} - \mathbf{W}^-\mathbf{a}$。

对于布尔-整数乘积，可以直接用OT构造，构造如下：

Protocol 6: Boolean-Integer Inner Product

Parties: P_1 and P_2

Input: Arithmetic shares of integer vector $\mathbf{a} \in \mathbb{Z}_q^m$ and
Boolean shares of binary vector $\mathbf{w} \in \{0,1\}^m$

Output: Arithmetic shares of $z = \mathbf{w}^\top \mathbf{a}$

1: Each P_i generates random values $(z_{i,j})_{j\in[m]}$.
2: **for** $j \in [m]$ **do**
3: $\quad$ P_i sets
$$m_{i,0} := \langle \mathbf{w}_j \rangle_i \cdot [\![\mathbf{a}_j]\!]_i - z_{i,j}$$
$$m_{i,1} := \neg\langle \mathbf{w}_j \rangle_i \cdot [\![\mathbf{a}_j]\!]_i - z_{i,j}$$
4: $\quad$ P_1 and P_2 run $\mathrm{OT}(m_{1,0}, m_{1,1}, \langle \mathbf{w}_j \rangle_2)$, with P_1 as Sender and P_2 as Chooser, for P_2 to obtain $z'_{1,j}$
5: $\quad$ P_1 and P_2 run $\mathrm{OT}(m_{2,0}, m_{2,1}, \langle \mathbf{w}_j \rangle_1)$, with P_2 as Sender and P_1 as Chooser, for P_1 to obtain $z'_{2,j}$
6: Each P_i sets $[\![z]\!]_i = \sum_{j\in[m]}(z_{i,j} + z'_{i,j})$.

进一步，可以利用COT对其进行优化：

Protocol 7: Boolean-Integer Inner Product via $m\times\mathrm{COT}_\ell$

Parties: P_1 and P_2

Input: ℓ-bit arithmetic shares of integer vector $\mathbf{a} \in \mathbb{Z}_q^m$ and Boolean shares of binary vector $\mathbf{w} \in \{0,1\}^m$

Output: Arithmetic shares of $z = \mathbf{w}^\top \mathbf{a}$

1: Each party P_i constructs a vector of correlation functions $\mathbf{f}^{\,\mathbf{i}} = (f_{i,j}(x))_{j\in[m]}$,
$$f_{i,j}(x) = x - [\![\mathbf{w}_j]\!]_i \cdot [\![\mathbf{a}_j]\!]_i + \neg[\![\mathbf{w}_j]\!]_i \cdot [\![\mathbf{a}_j]\!]_i$$
2: The parties run $m \times \mathrm{COT}_\ell(\mathbf{f}^{\,\mathbf{1}}, [\![\mathbf{w}_j]\!]_2)$ with P_1 acting as the Sender, and P_1 obtains $\mathbf{x}$ while P_2 obtains $\mathbf{y}$.
3: The parties run $m \times \mathrm{COT}_\ell(\mathbf{f}^{\,\mathbf{2}}, [\![\mathbf{w}_j]\!]_1)$ with P_2 acting as the Sender, and P_2 obtains $\mathbf{x}'$ while P_1 obtains $\mathbf{y}'$.
4: P_1 sets $[\![z]\!]_1 = \sum_{j\in[m]}([\![\mathbf{w}_j]\!]_1 \cdot [\![\mathbf{a}_j]\!]_1 - \mathbf{x}_j + \mathbf{y}'_j)$
5: P_2 sets $[\![z]\!]_2 = \sum_{j\in[m]}([\![\mathbf{w}_j]\!]_2 \cdot [\![\mathbf{a}_j]\!]_2 - \mathbf{x}'_j + \mathbf{y}_j)$

上述两个协议的正确性和安全性根据OT、COT的基本性质很容易推导。因此，最终计算$\mathbf{z}$的协议如下：

Protocol 8: Ternary-Integer Matrix-Vector Product

Parties: P_1 and P_2

Input: Arithmetic shares of integer vector $\mathbf{a} \in \mathbb{Z}_q^m$ and Boolean shares of binary matrices $\mathbf{W}^+, \mathbf{W}^- \in \{0,1\}^{n,m}$

Output: Arithmetic shares of $\mathbf{z} = \mathbf{W}^+\mathbf{a} - \mathbf{W}^-\mathbf{a}$

1: P_1 and P_2 compute $[\![\mathbf{W}^+\mathbf{a}]\!]$ using n executions of Protocol 7.

2: P_1 and P_2 compute $[\![\mathbf{W}^-\mathbf{a}]\!]$ using n executions Protocol 7.
3: P_i sets $[\![\mathbf{z}]\!]_i := [\![\mathbf{W}^+\mathbf{a}]\!]_i - [\![\mathbf{W}^+\mathbf{a}]\!]_i$.

1.2 Secure Forward Pass

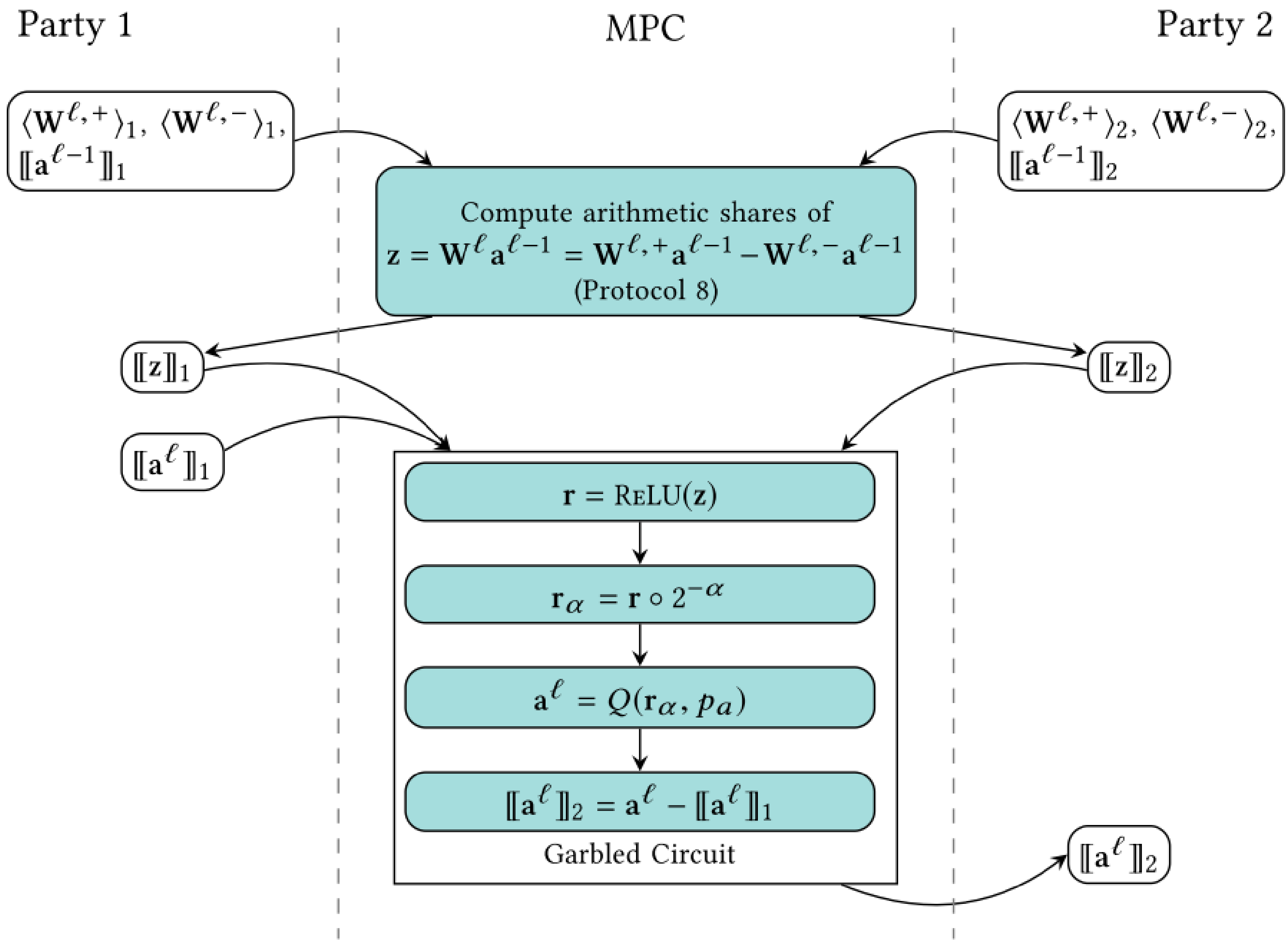

Figure 3: Our MPC protocol for private prediction (forward pass). We compose three protocols to evaluate one layer of the form $f(\mathbf{W}\mathbf{a})$, with ternary $\mathbf{W}$, and where f = ReLU.

如下图所示，进行完线性层计算之后，需要进行激活函数计算：$r = RELU(z) = (1 \oplus msb(z)) \cdot z$。然后进行量化操作，该部分计算利用GC实现。

1.3 Secure Backward Pass

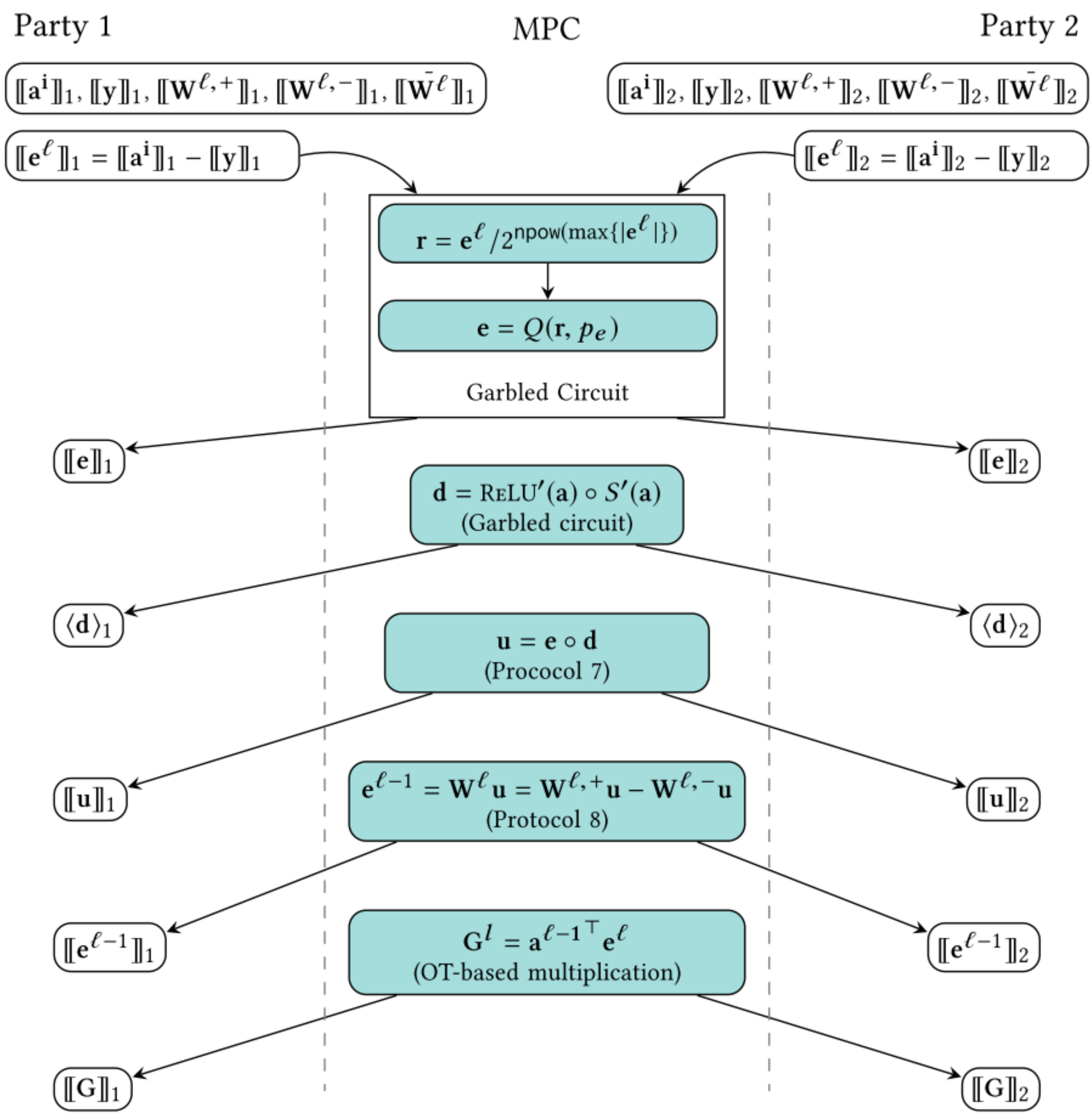

Figure 4: Our protocol for the backward pass, corresponding to Algorithms 2 from Section 3.

在反向传播中，首先利用GC计算量化输出层梯度，然后根据反向传播算法，计算激活层、线性层等层的梯度。和传统的NN相比，需要额外处理梯度量化函数。

2. Evaluation

本文主要和SecureML方案对比时间。

矩阵向量乘积对比如下:

Network	k	QUOTIENT (s)	GC (s)	SecureML (OT) (s)	SecureML (LHE) (s)
LAN	10^3	0.08	0.025	0.028	5.3
	10^4	0.08	0.14	0.16	53
	10^5	0.13	1.41	1.4	512
	10^6	0.60	13.12	14*	5000*
	10^7	6.0	139.80	140*	50000*
WAN	10^3	1.7	1.9	1.4	6.2
	10^4	1.7	3.7	12.5	62
	10^5	2.6	20	140	641
	10^6	7.3	148	1400*	6400*
	10^7	44	1527	14000*	64000*

Table 1: Comparison of our COT-based component-wise multiplication of k-dimensional vectors with ternary fixed-point multiplication using garbled circuits (GC) and SecureML [41] (OT, LHE). One of the vectors hold only ternary values.

Network	n	QUOTIENT (s)	SecureML (OT Vec) (s)	SecureML (LHE Vec) (s)
LAN	100	0.08	0.05	1.6
	500	0.1	0.28	5.5
	1000	0.14	0.46	10
WAN	100	1.7	3.7	2
	500	2	19	6.2
	1000	2.7	34	11

Table 2: Performance comparison of our matrix-vector multiplication approach with the vectorized approaches of SecureML [41] (OT, LHE). Here we multiply a $128 \times n$ ternary matrix with an n-dimensional vector.

不同层的开销和不同函数的开销如下：

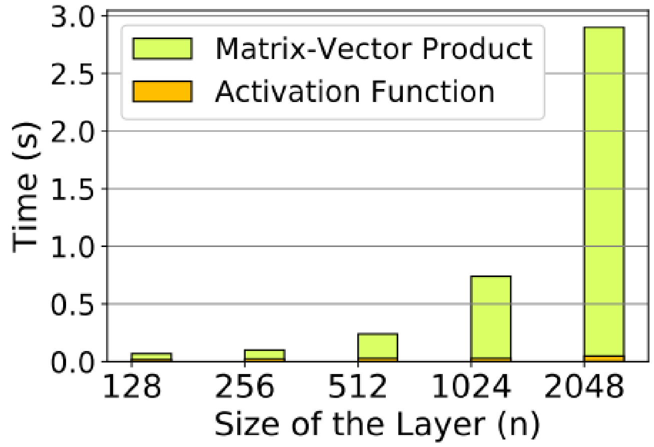

Figure 5: Forward pass time for single prediction over an $n \times n$ fully connected layer.

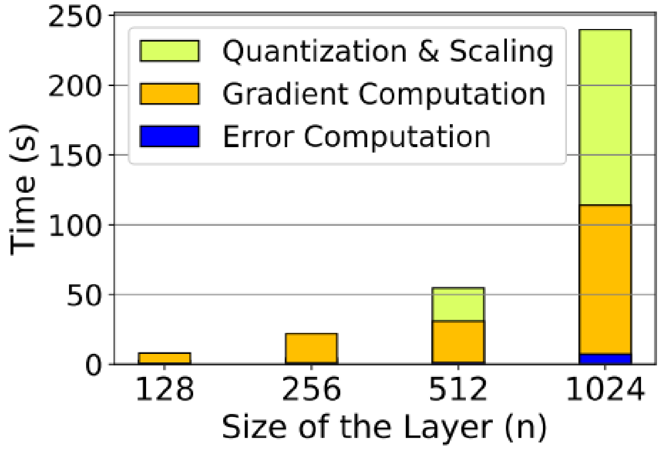

Figure 6: Forward and Backward pass time over an $n \times n$ fully-connected layer for 1 batch. Here batch size = 128.

对于真实数据集，量化下的模型性能如下：

Dataset	QUOTIENT (%)	Floating point (%)
MNIST	99.38	99.48
MotionSense	93.48	95.65
Thyroid	97.03	98.30
Breast cancer	79.21	80.00
German credit	79.50	80.50

Table 3: Accuracy comparison of training over state of the art floating point neural networks their fixed point equivalents using QUOTIENT.

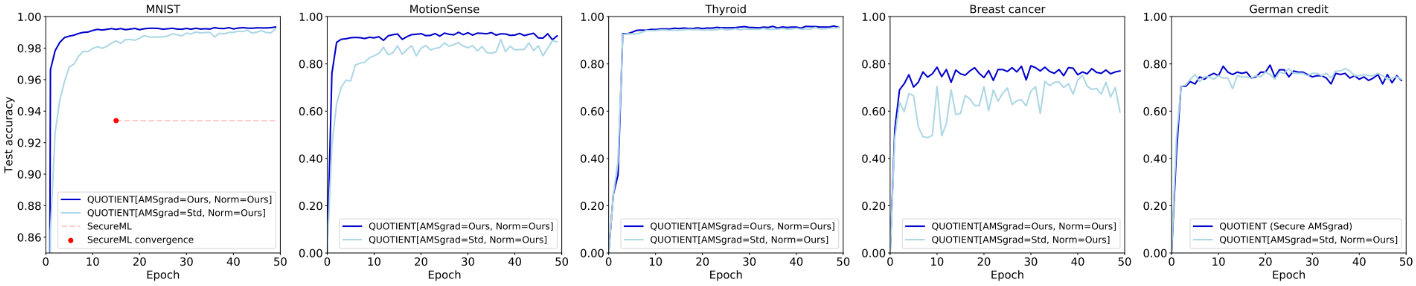

Figure 7: Performance comparison of secure AMSgrad and secure SGD for QUOTIENT. The plots compare training curves over MNIST (using CNN), MotionSense, Thyroid, Breast cancer and German credit datasets.

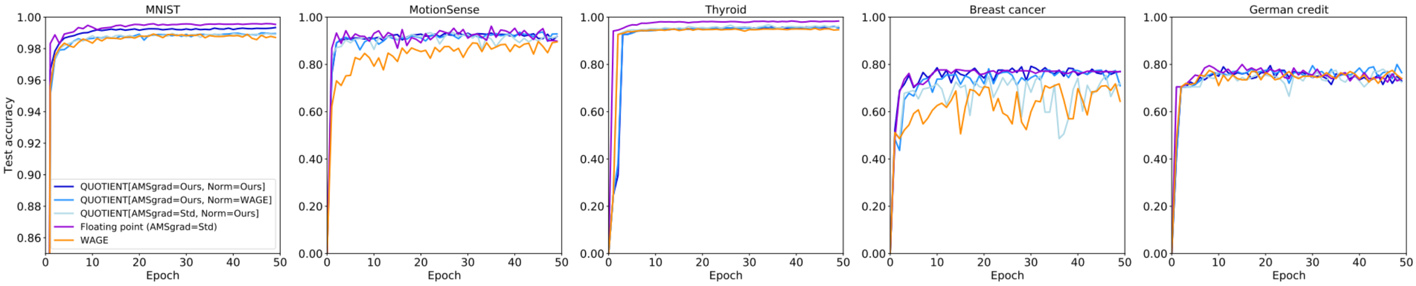

Figure 8: Performance comparison of three different variants of QUOTIENT training with floating point and WAGE [57] training on MNIST, MotionSense, Thyroid, Breast cancer and German credit datasets. `QUOTIENT[AMSgrad=Ours, Norm=WAGE]` and `QUOTIENT[AMSgrad=Std, Norm=Ours]`) differ from (`QUOTIENT[AMSgrad=Ours, Norm=Ours]` in using next power of 2 vs the Closet Power of 2 and using standard AMSgrad by changing lines 9 and 11 of secure AMSgrad in Algorithm 4, respectively.

训练和预测的时间如下：

	LAN															
	MNIST								MotionSense		Thyroid		Breast cancer		German credit	
	2 × (128FC)		3 × (128FC)		2 × (512FC)		3 × (512FC)		2 × (512FC)		2 × (100FC)		3 × (512FC)		2 × (124FC)	
Epoch	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc
1	8.72h	0.8706	10.05h	0.9023	27.13h	0.9341	38.76h	0.9424	10.07h	0.8048	0.08h	0.2480	14.51h	0.4940	0.03h	0.4100
5	43.60h	0.9438	50.25h	0.9536	135.65h	0.9715	193.80h	0.9745	50.35h	0.8847	0.40h	0.9341	72.55h	0.7360	0.15h	0.725
10	87.20h	0.9504	100.50h	0.9604	271.30h	0.9772	387.60h	0.9812	100.70h	0.8855	0.80h	0.9453	145.10h	0.7600	0.30h	0.7900
	WAN															
	MNIST								MotionSense		Thyroid		Breast cancer		German credit	
	2 × (128FC)		3 × (128FC)		2 × (512FC)		3 × (512FC)		2 × (512FC)		2 × (100FC)		3 × (512FC)		2 × (124FC)	
Epoch	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc
1	50.74h	0.8706	57.90h	0.9023	139.71h	0.9341	190.10h	0.9424	44.43h	0.8048	0.52h	0.2480	74.10h	0.4940	0.15h	0.4100
5	253.7h	0.9438	289.5h	0.9536	698.55h	0.9715	950.5h	0.9745	222.15h	0.8847	2.6h	0.9341	370.5h	0.7360	0.75h	0.725
10	507.4h	0.9504	579h	0.9604	1397.1h	0.9772	1901h	0.9812	444.3h	0.8855	5.2h	0.9453	741h	0.7600	1.5h	0.7900

Table 4: Training time and accuracy values for various datasets and architectures after 1, 5 & 10 training epochs using QUOTIENT over LAN and WAN.

	MNIST					MotionSense		Thyroid	Breast cancer		German credit
	2 × (128FC)	3 × (128FC)	2 × (512FC)	3 × (512FC)	Conv	2 × (512FC)	Conv	2 × (100FC)	3 × (512FC)	Conv	2 × (124FC)
Single Prediction(s)	0.356	0.462	0.690	0.939	192	0.439	134	0.282	3.58	62	0.272
Batched Prediction (s)	2.24	2.88	4.79	6.50	2226	2.91	1455	1.83	44.02	447	1.77

Table 5: Prediction time for various datasets and architectures using QUOTIENT over LAN. Here batch size = 128.

	MNIST				MotionSense	Thyroid	Breast cancer	German credit
	2 × (128FC)	3 × (128FC)	2 × (512FC)	3 × (512FC)	2 × (512FC)	2 × (100FC)	3 × (512FC)	2 × (124FC)
Single Prediction(s)	6.8	8.8	14.4	19.9	9.46	5.99	33.3	5.1
Batched Prediction (s)	8.3	10.9	22.6	29.9	12.08	6.89	69.1	7.3

Table 6: Prediction time for various datasets and architectures using QUOTIENT over WAN. Here batch size = 128.

3. Conclusion

本文基于三值量化网络做了一定的改进，并利用2PC实现了安全训练和预测，但是其通信还是很高。本文没有列出通信量数据，而且对于大的模型时间开销也是巨大。