这次介绍的是 M. Sadegh Reiazi 等人发表在 Usenix Security'19 上的XONN。

XONN聚焦在隐私保护机器学习中的安全预测问题。和之前的文章不同的是，XONN不再单独只考虑优化密码学协议，而是借助DNN的一些优化方法，并为这些方法设计高效的密码学协议，从而更好的优化系统性能。整体来说，本文一方面借助神经网络中的二值量化（Binarization）和剪枝（Pruning）在保证模型性能（例如准确率）不受很大影响的前提下，将模型参数从float32量化为$\{-1,1\}$，得到二值化神经网络（Binary Neural Network， BNN），从而减少编码使用的比特位长。进一步，对于BNN中的计算，利用混乱电路等密码学技术设计专门的乘法、激活函数、BatchNorm、和池化操作的安全计算算子。本文针对的安全模型是半诚实模型。

0. 基础知识

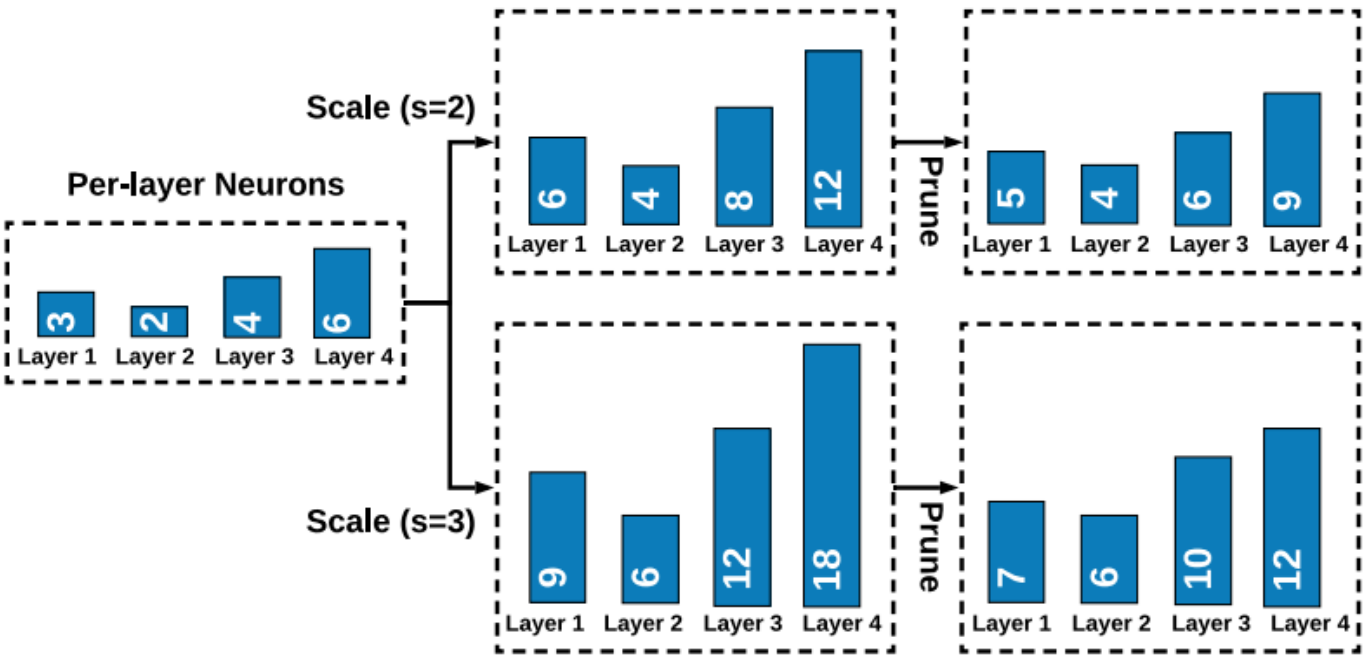
和之前介绍的方案类似，XONN要求一些对于神经网络计算的一些基础知识，例如线性层、激活层、BatchNorm、和池化；对于安全计算部分，则需要了解秘密分享（Secret Sharing）、茫然传输（Oblivious Transfer）、和混乱电路（Garbled Circuits，GC）。

1. Customized Network Binarization

BNN虽然减小了计算开销，但是其模型的性能也有一定程度的下降。为了尽可能的保证模型的性能，XONN首先提出了一种生成处理BNN的方法。大致分为两步Linear Scaling 和 Network Trimming 两步。

1.1 Linear Scaling

在BNN训练之前，先将每一层的通道/神经元扩大s倍，例如$s=2$。之后，再训练扩大的BNN。例如下图，就是分别对BNN扩大2倍和3倍。



1.2 Network Trimming

完成BNN的训练之后，则需要利用剪枝相关的技术在尽可能保证模型性能可接受的情况下删除多余的通道和神经元，从而加速下文的安全计算协议的执行。具体来说，该部分分为 *Feature Ranking* 和 *Iterative Pruning* 两步：

1.2.1 Feature Ranking

在float类型的DNN中，可以利用神经元绝对值的大小来判断重要性，从而简单的完成剪枝。但是，在BNN中模型训练得到的参数都是$\{-1,1\}$，因为无法利用神经元剪枝。为了解决这个问题，XONN提出了利用梯度绝对值的大小来判断一个神经元是否重要。直观上，如果一个神经元在训练过程中变化比较大，相应的在最终模型中的占据更高的重要性。

1.2.2 Iterative Pruning

实现Feature Ranking之后，便可以根据重要性进行剪枝。例如对于BNN第$l*$层，删除$p*$个最不重要的神经元，定义收益函数

$$reward(l,p)=\frac{c_{curr}-c_{next}}{e^{a_{curr}-a_{next}}}$$

其中c_{curr}代表现在的模型用GC表示的复杂度，c_{next}代表剪枝后的模型GC复杂度。而a_{curr}和a_{next} 分别代表当前和剪枝后的模型准确率。直观上，Iterative Pruning的主要目的就是在保证模型性能的同时尽可能的降低模型的GC复杂度。

2. Oblivious Inference

如前所述，DNN的执行主要分为线性层，激活层，BatchNorm，和池化操作。下面将一一介绍相关计算在BNN中的安全计算协议。

2.1 Binary Linear Layer

在线性层，主要是计算向量乘法，包括全连接和卷积操作，都可以看作是向量乘法的操作。除此之外，$\{-1,1\}$在安全协议中被编码为 $\{0,1\}:-1\rightarrow 0,1\rightarrow 1$。

2.1.1 Integer-VDP

在计算的第一层，由于用户上传的数据是多比特的整数（不是 $\{-1,1\}$ ），因此需要对第一层的计算额外考虑。XONN使用OT设计协议Oblivious Conditional Addition Protocol (OCA) 来实现该层的计算。注意，为了和原文保持一致，此处使用 v_2 和 v_1 分别表示服务器的模型 w 和客户端输入样本 x 。

对于服务器端的持有选择比特向量 v_2 ，客户端持有输入向量 v_1 。假设 v_1 中的每个数据都是 b 比特，客户端首先对持有的数据进行比特拉伸 $b\rightarrow b'$ ，其中 $b'=\lceil log_2(n\cdot(2^b-1))\rceil$ ， $|v_1|=n$ 。拉伸之后得到 v_1^* 。进一步，求拉伸之后的数据的二位最高有效位补码 (two's complement) 得到 $\bar{v_1^*}$

。随机生成向量 r 。对于每一个数，客户端发送 $(\overline{v_1^*}[i]-r[i], v_1^*[i]-r[i])$ 而服务器输入 $v_2[i]$ 作为选择比特：如果 $v_2[i]=0$ （真实值 $=-1$ ）则获得 $\overline{v_1^*}[i]-r[i]$ ，否则获得 $v_1^*[i]-r[i]$ 。公式表示如下：

$$v_t[i]=\begin{cases}\overline{v_1^*}[i]-r[i] \mod 2^{b'}, & v_2[i]=0\\ v_1^*[i]-r[i] \mod 2^{b'}, & v_2[i]=1\end{cases}$$

整体的协议如下图所示：

Sender:

(1) Bit-extends all elements of $\mathbf{v_1}$ and creates $\mathbf{v_1^*}$

(2) Creates two's complement of $\mathbf{v_1^*}$: $\overline{\mathbf{v_1^*}}$

(3) Creates random vector $\mathbf{r}$: same size as $\overline{\mathbf{v_1^*}}$

(4) Creates list of first messages as $\mathbf{m^2}=\overline{\mathbf{v_1^*}}-\mathbf{r}\ mod\ 2^{b'}$

(5) Creates list of second messages as $\mathbf{m^1}=\mathbf{v_1^*}-\mathbf{r}\ mod\ 2^{b'}$

Sender & Receiver:

(6) Parties engage in Oblivious Transfer (OT)

Sender puts $\mathbf{m^1}$ and $\mathbf{m^2}$ as message vectors

Receiver puts $\mathbf{v_2}$ vector as selection bits

Receiver:

(7) Gets vector $\mathbf{v_t}$ where:

$$\mathbf{v_t}[i]=\begin{cases}\overline{\mathbf{v_1^*}}[i]-\mathbf{r}[i]\ mod\ 2^{b'} & (\text{if } \mathbf{v_2}[i]=0)\\ \mathbf{v_1^*}[i]-\mathbf{r}[i]\ mod\ 2^{b'} & (\text{if } \mathbf{v_2}[i]=1)\end{cases}$$

Sender:

(8) Computes her additive share of VDP result as:

$$y_1=\sum_{i=1}^{n}\mathbf{r}[i]\ mod\ 2^{b'}$$

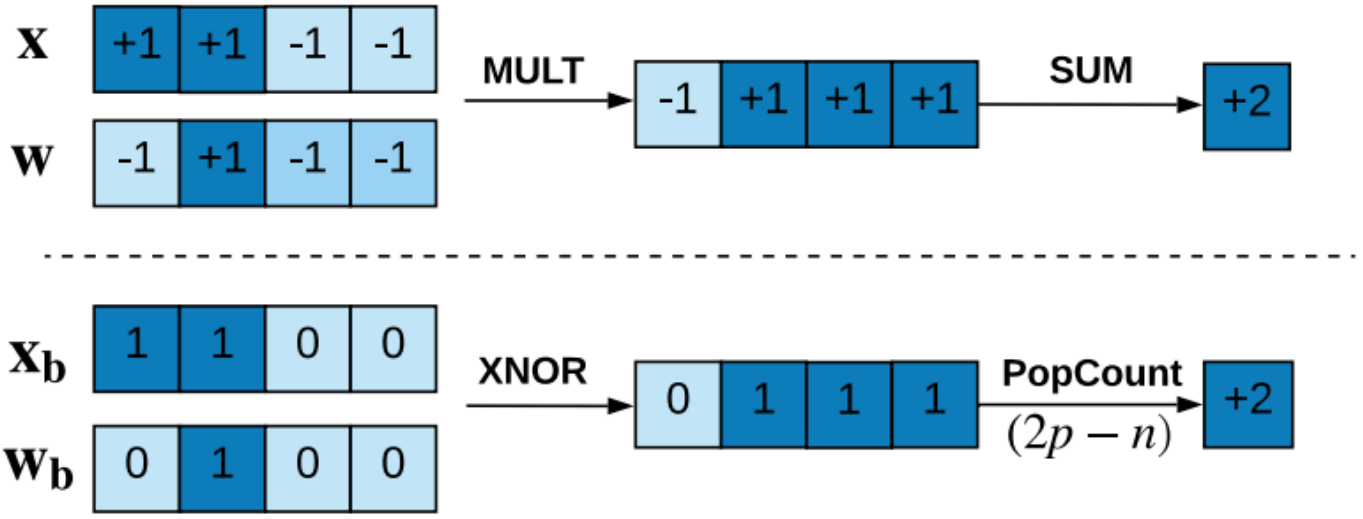
Receiver:

(9) Computes his additive share of VDP result as:

$$y_2=\sum_{i=1}^{n}\mathbf{v_t}[i]\ mod\ 2^{b'}$$

2.1.2 Binary-VDP

除了第一层的计算，后续线性层的计算则是在二值上进行的。因为所有的值都是从$\{-1,1\}$编码到了$\{0,1\}$, XONN有如下的观察：



即向量乘法可以利用XNOR（同或)和PopCount来替代。正确性在于假设真实值中1的个数是p，那么-1的个数则为$n-p$。则向量乘积的结果则为$p-(n-p)=2p-n$，和XNOR， PopCount之后的结果一样。

2.2 Binary Activation Function

除了线性层，激活层在BNN中则是求输入x的$y=Sign(x)$。具体来说如果$x\geq 0$，$y=1$；否则为$y=-1$。该功能可以用混乱电路实现。

2.3 Binary Batch Normalization

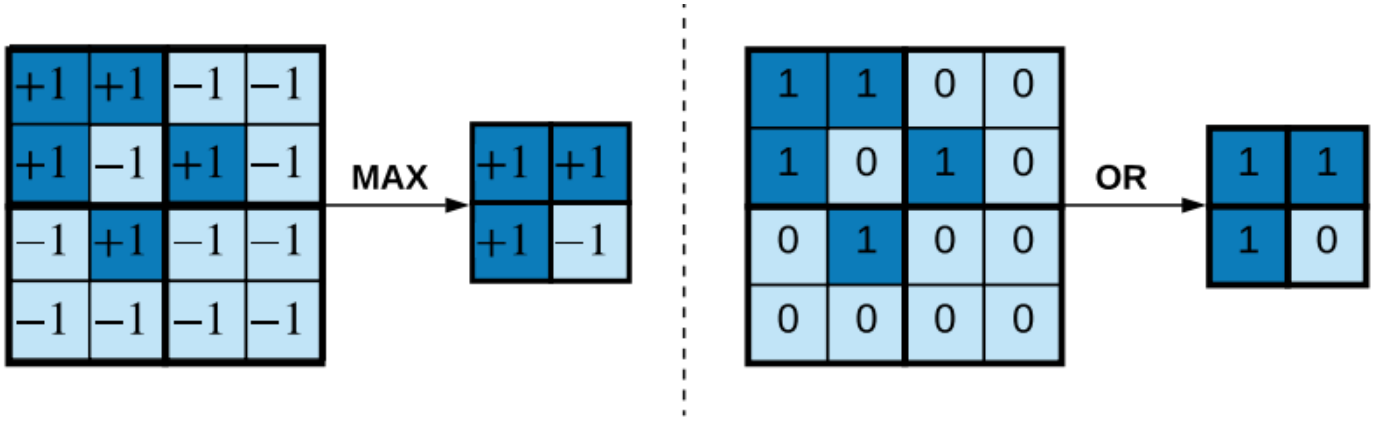
在BNN的inference中，Batch Normalization一般紧随一个 Activation Function。两者合并等价于：

$$y=Sign(\gamma\cdot x+\beta)=Sign(x+\frac{\beta}{\gamma})$$

当神经网络训练完成之后，γ 和 β 都是固定参数，其中$\gamma>0$。因此，$y=Sign(x+\frac{\beta}{\gamma})$可以利用GC通过比较$x$和$-\frac{\beta}{\gamma}$ 实现。除此之外，第一层的Integer-VDP之后也通常跟着BN+BA，因此可以自动的将中间结果转化到$\{0,1\}$上。

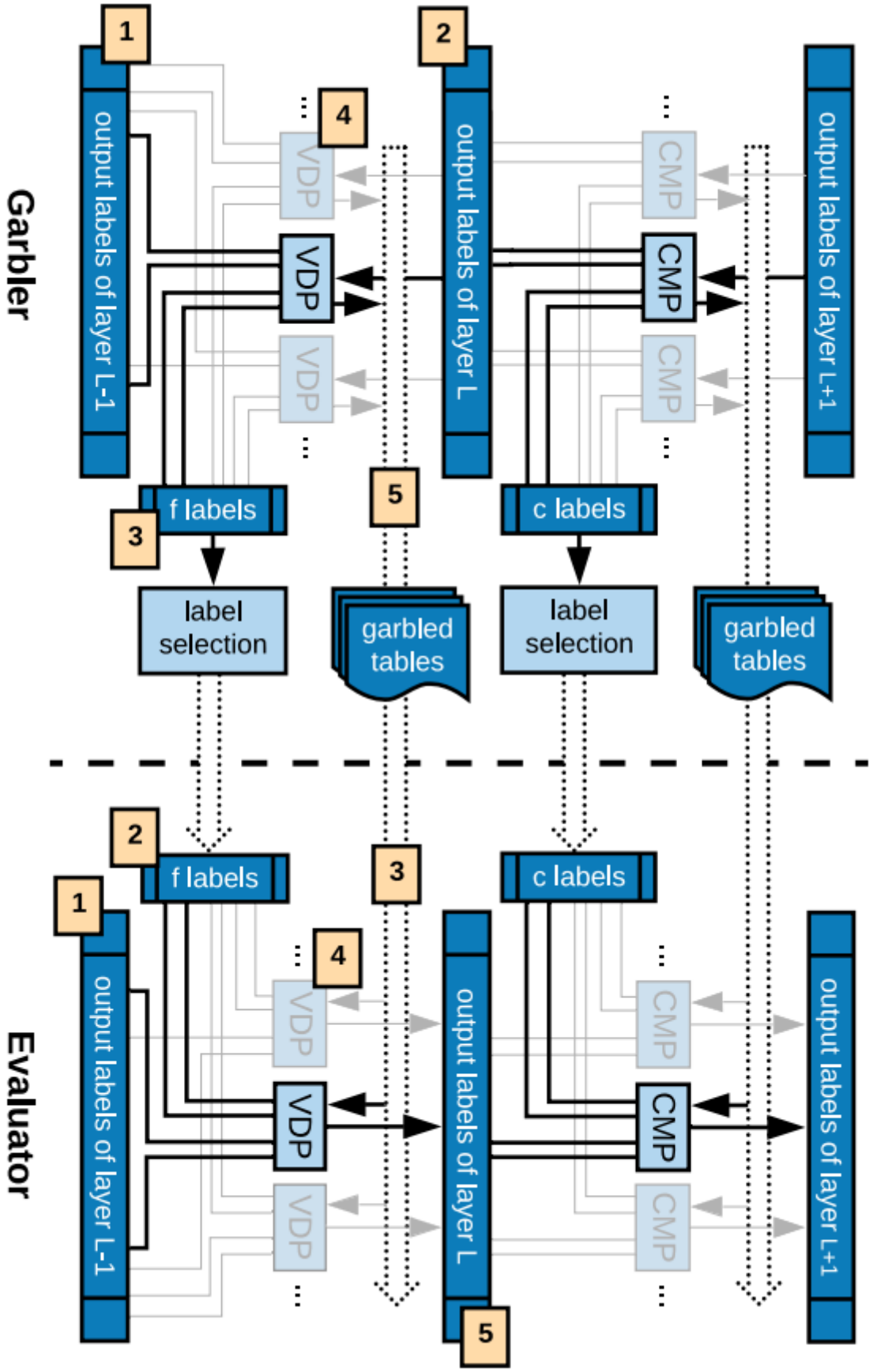
2.4 Binary Max-Pooling

XONN利用OR操作实现了Max-Pooling操作。如下图所示，$\{-1,1\}$上的MAX的Max和$\{0,1\}$上的OR是等价的。



3. 实验

XONN使用OT、GC、和Secret Sharing实现全部的模块，并使用流水线技术加速系统性能。整体的系统架构如下图所示：



3.1 High-Level Comparison

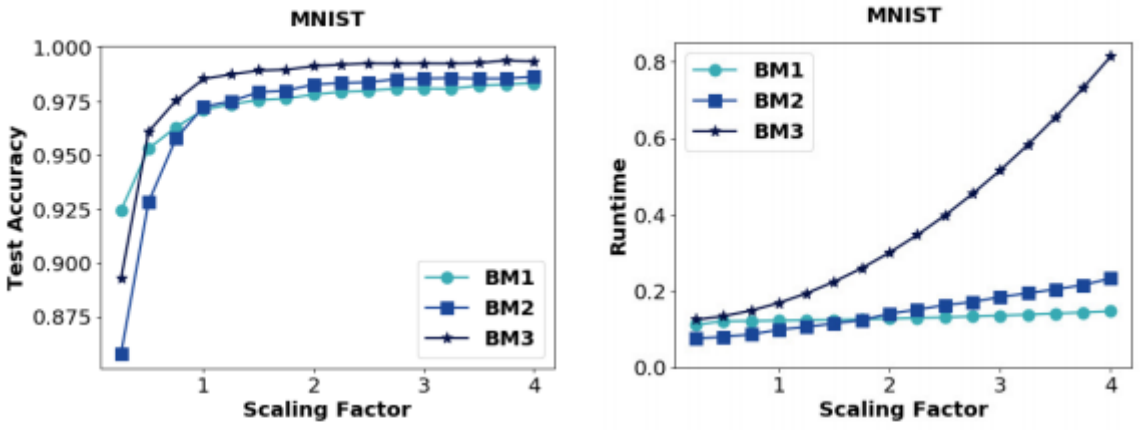
首先，作者从宏观了比较了XONN和现存的几种方案，结果如下：

Table 2: High-Level Comparison of oblivious inference frameworks. "**C**"onstant round complexity. "**D**"eep learning/secure computation co-design. "**I**"ndependence of secondary server. "**U**"pgradeable to malicious security using standard solutions. "**S**"upporting any non-linear layer.

Framework	Crypto. Protocol	C	D	I	U	S
CryptoNets [14]	HE	✓	✗	✓	✗	✗
DeepSecure [13]	GC	✓	✓	✓	✓	✓
SecureML [8]	HE, GC, SS	✗	✗	✗	✗	✗
MiniONN [9]	HE, GC, SS	✗	✗	✓	✗	✓
Chameleon [7]	GC, GMW, SS	✗	✗	✗	✗	✓
EzPC [25]	GC, SS	✗	✗	✓	✗	✓
Gazelle [10]	HE, GC, SS	✗	✗	✓	✗	✓
XONN (This work)	GC, SS	✓	✓	✓	✓	✓

3.2 Evaluation Results

作者对几种不同的模型分别测试了经过训练、剪枝后的BNN的准确率、OCA在不同scaling-factor下的通信、已经系统整体的性能。此处只展示MNIST数据集上的效果，更多的实验结果请参考原文。



(a) (b)

Figure 9: Effect of scaling factor on (a) accuracy and (b) inference runtime of MNIST networks. No pruning was applied in this evaluation.

准确率如上图。

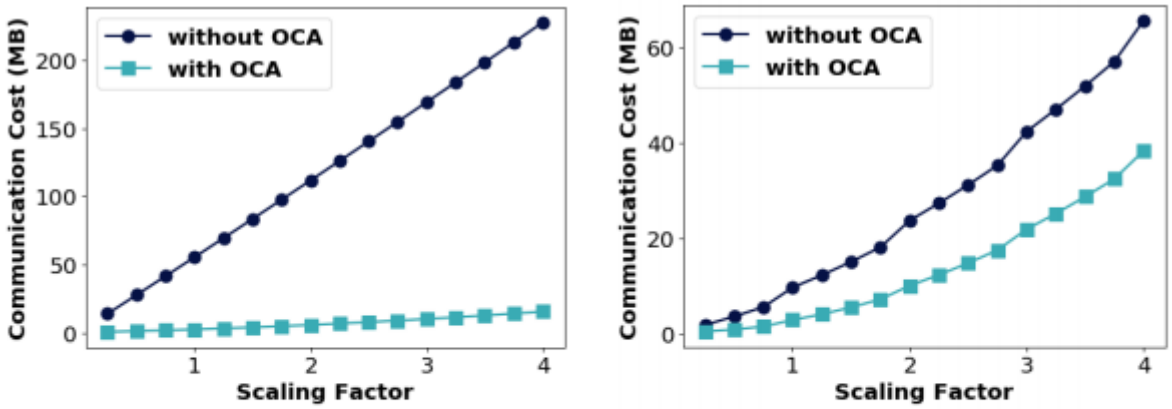


Figure 10: Effect of OCA on the communication of the BM1 (left) and BM2 (right) networks for different scaling factors. No pruning was applied in this evaluation.

OCA的通信与scaling factor的关系如上图。

Table 4: Comparison of XONN with the state-of-the-art for the MNIST network architectures.

Arch.	Framework	Runtime (s)	Comm. (MB)	Acc. (%)	s
BM1	SecureML	4.88	-	93.1	-
	MiniONN	1.04	15.8	97.6	-
	EzPC	0.7	76	97.6	-
	Gazelle	0.09	0.5	97.6	-
	XONN	0.13	4.29	97.6	1.75
BM2	CryptoNets	297.5	372.2	98.95	-
	DeepSecure	9.67	791	98.95	-
	MiniONN	1.28	47.6	98.95	-
	Chameleon	2.24	10.5	99.0	-
	EzPC	0.6	70	99.0	-
	Gazelle	0.29	8.0	99.0	-
	XONN	0.16	38.28	98.64	4.00
BM3	MiniONN	9.32	657.5	99.0	-
	EzPC	5.1	501	99.0	-
	Gazelle	1.16	70	99.0	-
	XONN	0.15	32.13	99.0	2.00

系统总体的开销和其他方案的比较如上表。

4.总结

本文是比较早的将神经网络量化和安全计算结合起来实现隐私保护预测的文章，为开发高效的系统提供了新思路。除此之外，由于本文主要使用两方计算中的OT和GC，本文方案也可以较快的改造用于恶意模型。