



# 基于安全多方计算在机器学习算法中的技术研究 毕业答辩

数学与信息科学学院

答辩人：刘坤                  导师：唐春明教授  
答辩时间：2023年11月27日

# 目录

CONTENTS

01

隐私保护机器学习

02

论文主要工作

03

总结和展望

# 目录

CONTENTS

01

隐私保护机器学习

02

论文主要工作

03

总结和展望

# 隐私保护机器学习

- 输入隐私：确保其他方(包括模型开发人员)无法看到用户的输入数据
- 输出隐私：确保模型的输出只能由其数据被推断的客户端访问
- 模型隐私：确保敌对方无法窃取模型
- 参数隐私：确保敌对方无法从机器学习过程中产生的参数得到隐私信息

技术	隐私要求					安全性要求	优缺点
	原始数据	身份	特征	模型	输入		
差分隐私	√	√	√	√		可证明安全	损失 ML 模型的准确性
同态加密	√	√	√		√	密文计算，可证明安全	高计算量，便于数值数据
多方计算	√	√	√		√	密文计算	高通信成本、适用性强、高计算量
联邦学习	√	√	√	√		去中心化训练	高通信成本、适用性强

# 隐私保护机器学习的发展

- 定制的 PPML 协议，高效率获得特定学习任务
- 在两方环境中，具有半诚实安全性的机器学习推理似乎非常高效，但针对恶意对手的机器学习推理和训练仍然效率较低
- 机器学习应用程序，一般需要处理多个不同类型的函数
- 基于函数秘密共享(FSS)构建混合模式下 MPC 协议的新方法，这适用于高通信复杂度和多通信轮次的联合机器学习。

参与方数量	隐私保护机器学习模型	同态	加法秘密共享	重复秘密共享	不经意传输	混淆电路	半诚实	恶意	预测	训练
两方参与	SecureML [26]	✓	✓			✓	✓		✓	✓
	MiniONN [29]	✓	✓			✓	✓		✓	
	GAZELLE [30]	✓	✓			✓	✓		✓	
	Delphi [31]	✓	✓			✓	✓		✓	
	QuantizedNN [25]	✓	✓		✓		✓	中止	✓	
	EzPC [32]		✓			✓	✓		✓	
	XONN [33]		✓			✓	✓		✓	
	QUOTIENT [27]		✓			✓	✓		✓	✓
	MP2ML [34]	✓	✓			✓	✓		✓	
	CrypTFlow2 [28]	✓	✓		✓		✓		✓	
	SIRNN [35]		✓		✓		✓		✓	
	Chameleon [36]		✓			✓	✓		✓	
三方参与	ABY3 [37]		✓			✓	✓		✓	✓
	ASTRA [38]		✓	✓			✓	中止	✓	✓
	SecureNN [39]		✓				✓		✓	✓
	QuantizedNN [21]			✓			✓	中止	✓	
	BLAZE [40]		✓	✓			✓	失败	✓	✓
	CrypTFlow [41]		✓				✓		✓	
四方参与	Falcon [42]			✓			✓	中止	✓	✓
	SWIFT [43]		✓	✓			✓	保证输出	✓	✓
	CryptGPU [44]			✓			✓		✓	✓
	FLASH [45]		✓	✓			✓	保证输出	✓	✓
	Trident [46]		✓	✓		✓	✓	失败	✓	✓
	SWIFT [43]		✓	✓			✓	保证输出	✓	✓
	Tetrad [47]		✓	✓			✓	保证输出	✓	✓

# 研究内容与主要贡献



- 第 3 章提出基于安全两方计算的朴素贝叶斯分类方法，保护模型训练数据隐私。
- 第 4 章提出一种基于函数秘密共享的安全决策树分类方法。决策树通过顺序测试特征和阈值来对数据进行分类。
- 第 5 章提出基于门限加法同态的安全两方线性聚合算法，使机器学习模型能够对加密的用户输入执行线性操作。

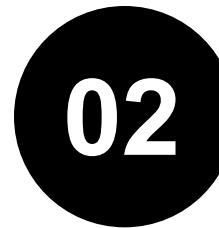
综上所述，本文主要研究了基于安全两方计算来保护朴素贝叶斯分类，决策树分类和数据聚合中的数据隐私。这为构建保护隐私的机器学习模型提供了可行高效的框架和范式。在实验中，这些方法均实现了与原始模型相当的分类准确率，同时提供了明确的安全性证明。

# 目录

CONTENTS



隐私保护机器学习



论文主要工作



总结和展望

# 目录

## CONTENTS

01

隐私保护机器学习



基于安全多方计算  
(STPC) 的隐私保护  
朴素贝叶斯分类

02

论文主要工作



基于函数秘密共享的安  
全多方决策树分类

03

总结和展望



基于门限加法同态加密  
的安全多方线性聚合机  
器学习

# 目录

## CONTENTS

01

隐私保护机器学习

02

论文主要工作

03

总结和展望

基于安全多方计算  
(STPC) 的隐私保护  
朴素贝叶斯分类



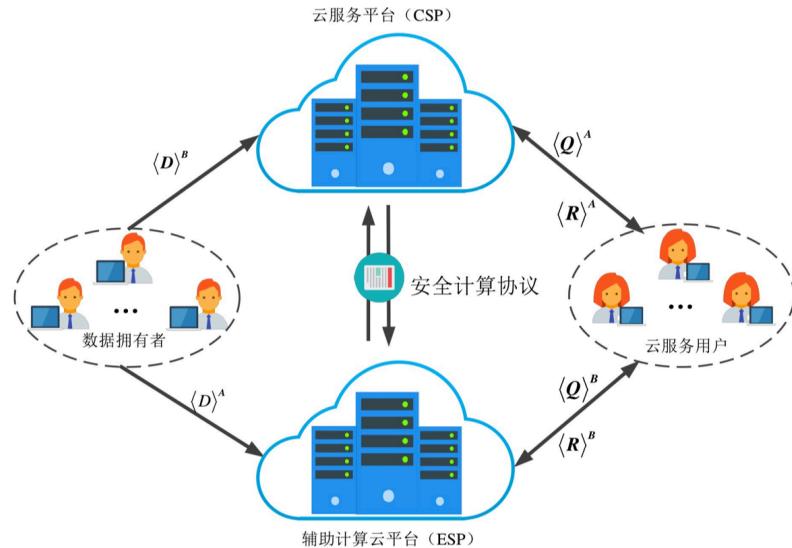
基于安全多方计算(STPC) 的  
隐私保护朴素贝叶斯分类



# 研究问题



基于安全多方计算的隐私保护朴素贝叶斯分类 (STPC) 已经存在于训练回归和神经网络模型的机器学习算法，其双服务器模型在传统的 PPML 工作中常用。由此，本文算法解决了基于多方安全的贝叶斯分类计算问题，同时保护了提供训练数据的个人的隐私，从而使公司和组织能够实现其实用目标，同时帮助个人保护其隐私。





## 基于安全多方计算(STPC) 的隐私保护朴素贝叶斯分 类

- 使用 STPC 在两个非共谋服务器之间分割训练数据，在不泄漏信息的情况下训练模型
- 使用 Beaver 的乘法三元组和 Yao 的乱码电路等加密技术秘密共享数据和实现中间计算
- 在半诚实模型下，进行安全性分析

## ■ 安全乘法重构算法

**输入:** 参与方  $S_0$  共享秘密信息  $\llbracket a \rrbracket_0$  和  $\llbracket b \rrbracket_0$ , 另一参与方  $S_1$  共享秘密信息  $\llbracket a \rrbracket_1$  和  $\llbracket b \rrbracket_1$ 。

**输出:** 两个参与方  $S_0$  和  $S_1$  都可以恢复  $\llbracket x \rrbracket$  和  $\llbracket y \rrbracket$  之间的乘积。

$S_i$  一方秘密分享  $\llbracket a \rrbracket_i, \llbracket b \rrbracket_i, i \in \{0, 1\}$  在本地的信息。双方执行乘法重构获得乘积  $z = x \cdot y$ 。

## ■ 安全分布式矩阵乘法三元组协议

**输入:** 分别是  $S_0$  和  $S_1$  的矩阵  $\llbracket X \rrbracket_q$  和  $\llbracket Y \rrbracket_q$ 。

**输出:** 生成  $\llbracket Z \rrbracket_q$  的  $\llbracket X \rrbracket_q$  和  $\llbracket Y \rrbracket_q$ .

$S_0$

$$\llbracket D \rrbracket_q \leftarrow \llbracket X_i \rrbracket - \llbracket U \rrbracket_q$$

$S_1$

$$\llbracket E \rrbracket_q \leftarrow \llbracket X_{i-1} \rrbracket - \llbracket V \rrbracket_q$$

$$\llbracket Z \rrbracket_q \leftarrow \llbracket W \rrbracket_q + E \llbracket U \rrbracket_q + D \llbracket V \rrbracket_q + DE$$

## ■ 定点算术计算

$$Q(x) = \begin{cases} 2^\lambda - \lceil 2^a \cdot |x| \rceil, & \text{if } x < 0 \\ \lceil 2^a \cdot |x| \rceil, & \text{if } x \geq 0 \end{cases}$$

- 有均匀随机的有限集合来确保加法秘密共享的安全性
- 使用一个序列表示连续的十进制数
- Alice 和 Bob 共享一个比特保密共享来实现，每个交互值都需要与其阈值进行比较。

## ■ 安全比特提取： $\mathbb{Z}_2$ 中的秘密共享安全转换为 $\mathbb{Z}_q$ 中的秘密共享

输入：Alice 输入  $\llbracket x \rrbracket_2$ ，让  $x_A \in \{0, 1\}$  和  $x_B \in \{0, 1\}$  表示 Bob 的输入共享。

输出：Alice 和 Bob 获得秘密共享  $\llbracket z \rrbracket_q$ 。

- 1: 定义  $\llbracket x_A \rrbracket_q$  为  $(x_A, 0)$  的共享， $\llbracket x_B \rrbracket_q$  为  $(0, x_B)$  的共享。
- 2: Alice 和 Bob 计算  $\llbracket y_A \rrbracket_q \leftarrow \llbracket x_A \rrbracket_q \llbracket x_B \rrbracket_q$ 。
- 3: 他们输出  $\llbracket z \rrbracket_q \leftarrow \llbracket x_A \rrbracket_q + \llbracket x_B \rrbracket_q - 2\llbracket y_A \rrbracket_q$ 。

## ■ 安全最大值计算

**输入:** 为了将  $\ell$  与  $k$  值的位长度进行比较,  $P_i$  方执行分布式乘法三元组协议和比较协议。

$P_i$  作为输入按位共享  $\llbracket x_{j,i} \rrbracket_q$  对于所有  $j \in \{1, 2, \dots, k\}$  和  $i \in \{1, 2, \dots, \ell\}$ .

**输出:** 如果  $w_j = 1$ , 则  $P_i$  将  $j$  附加到最后要输出的值。

- 假设各方  $P_1, P_2, \dots, P_n$  按位共享一组值  $(x_1, x_2, \dots, x_k)$  并希望其中一个 (假设  $P_1$ ) 学习所有参数  $c \in \{1, 2, \dots, k\}$  使得  $x_m \geq x_i$  对于所有  $i \in \{1, 2, \dots, k\}$ , 各方只想  $P_1$  学习

$$\mathbf{c} = \arg \max_{i \in \{1, 2, \dots, k\}} x_i.$$

## ■ 安全比特分解 : 将 $x$ 的共享从较大域 $\mathbb{Z}_q$ 转换为较小域 $\mathbb{Z}_2$ 的问题

**输入:**  $S_0$  和  $S_1$  有  $\llbracket m_i \rrbracket_q$  作为输入,  $q = 2^\ell$ 。

**输出:**  $S_0$  和  $S_1$  获取秘密共享  $\llbracket m \rrbracket_2$ .

## ■ 朴素贝叶斯分类

● N维向量 :  $\{x_1, x_2, \dots, x_n\}$  样本空间 :  $\{c_1, c_2, \dots, c_m\}$

● 先验概率 :  $\Pr(x)$  后验概率 :  $\Pr(c|x)$

$$\Pr(c|x) = \frac{\Pr(x|c) \Pr(c)}{\Pr(x)}.$$

● 累乘导致结果变得更小，小浮点数的乘法会产生小数点下溢，取对数

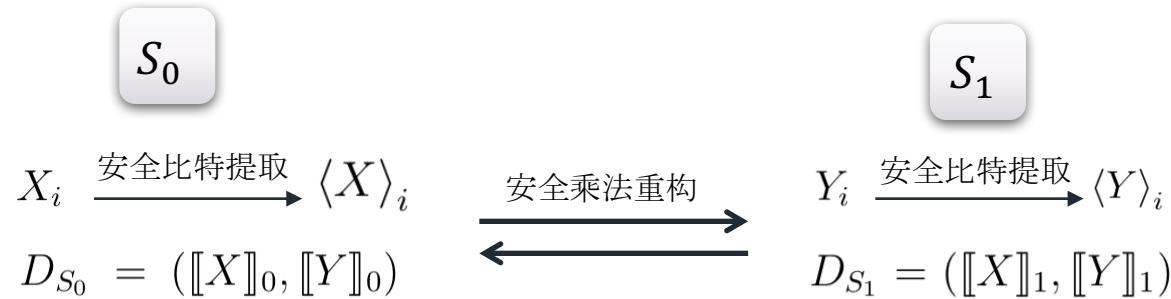
$$\log(\Pr(c|x)) = \log \left( \Pr(c) \prod_{i=1}^d \Pr(x_i|c) \right) = \log(\Pr(c)) + \sum_{i=1}^d \log(\Pr(x_i|c))$$

● 求最大值，执行分类任务

$$\hat{c} = \arg \max_c \left[ \log(\Pr(c)) + \sum_{k=1}^d \log(\Pr(x_k|c)) \right]$$

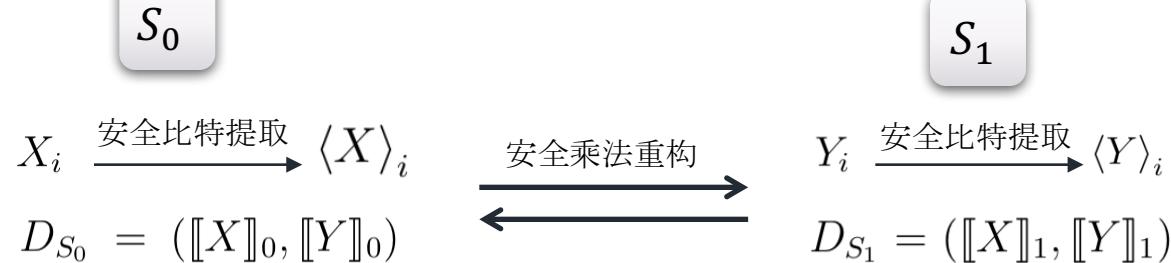


## 安全两方计算的朴素贝叶斯分类模型





# 安全两方计算的朴素贝叶斯分类模型



各自执行贝叶斯分类算法，计算秘密共享块

$\{\log(\Pr(c_j)), \log(\Pr(y_1|c_j)), \log(\Pr(y_2|c_j)), \dots, \log(\Pr(y_m|c_j)), \log(1 - \Pr(y_1|c_j)), \dots, \log(1 - \Pr(y_m|c_j))\}$



# 安全两方计算的朴素贝叶斯分类模型



各自执行贝叶斯分类算法，计算秘密共享块

$$\{\log(\Pr(c_j)), \log(\Pr(y_1|c_j)), \log(\Pr(y_2|c_j)), \dots, \log(\Pr(y_m|c_j)), \log(1 - \Pr(y_1|c_j)), \dots, \log(1 - \Pr(y_m|c_j))\}$$

$$\llbracket w \rrbracket_q \leftarrow \llbracket y_i \rrbracket_q \llbracket \log(\Pr(x_i|c_j)) \rrbracket_q + (1 - \llbracket y_i \rrbracket_q) \llbracket \log(1 - \Pr(x_i|c_j)) \rrbracket_q$$

$$\llbracket u_i \rrbracket_q \leftarrow \llbracket \log(\Pr(c_j)) \rrbracket_q + \sum_{i=1}^n \llbracket w_i \rrbracket_q$$



# 安全两方计算的朴素贝叶斯分类模型



各自执行贝叶斯分类算法，计算秘密共享块

$$\{\log(\Pr(c_j)), \log(\Pr(y_1|c_j)), \log(\Pr(y_2|c_j)), \dots, \log(\Pr(y_m|c_j)), \log(1 - \Pr(y_1|c_j)), \dots, \log(1 - \Pr(y_m|c_j))\}$$

$$[\![w]\!]_q \leftarrow [\![y_i]\!]_q [\![\log(\Pr(x_i|c_j))]\!]_q + (1 - [\![y_i]\!]_q) [\![\log(1 - \Pr(x_i|c_j))]\!]_q$$

$$[\![u_i]\!]_q \leftarrow [\![\log(\Pr(c_j))]\!]_q + \sum_{i=1}^n [\![w_i]\!]_q$$

$$[\![c]\!]_2$$

各自执行安全最大值算法计算c

$$[\![c]\!]_2$$



# 安全两方计算的朴素贝叶斯分类模型



- 两方服务器不泄漏各自均能得到模型
- 不涉及双方数据的分割方式
- 加密部分使用到加同态加密，因此协议是可证明完全
- 特征提取需要  $O(mn)$  操作来进行分布式预处理，计算类别条件概率的  $k$  不经意矩阵乘法的主要成本为  $O(kmn)$ ，离线阶段的总计算量为  $O(kmn)$ ，每次预测的总计算量为  $O(km)$



# 安全两方计算的朴素贝叶斯分类模型安全性分析

- 引理 1. 通过证明两方隐私保护朴素贝叶斯分类协议  $\pi$  的安全性，证明它在半诚实对手存在的情况下  $UC$  实现了理想的功能  $F_{NB}$ 。
  
- 引理 2. 通过证明两方隐私保护朴素贝叶斯分类协议  $\pi$  的安全性，证明它实现了  $\mathcal{F}_{NB}$  中的理想功能  $(\mathcal{F}_{OT}, \mathcal{F}_{BEAVER})$ - 混合模型，具有针对半诚实对手的安全性。



## 实验分析

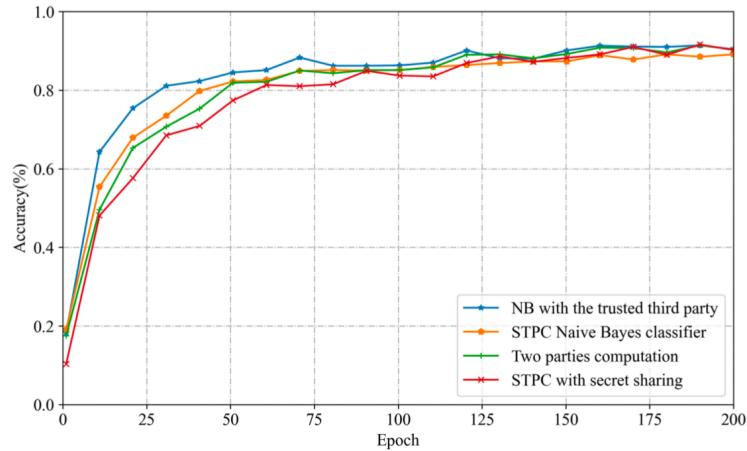


图 3-2 每一轮次模型的准确率

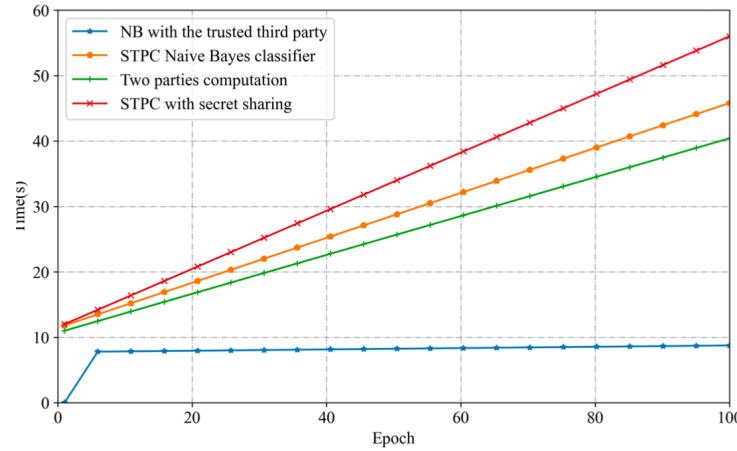


图 3-3 模型在不同时期消耗时间成本

- 本研究的方法达到了同等的精度，并且只需要很少的额外计算和通信开销。
- 具有 60,000 个训练图像的 MNIST 数据集上，获得了 97.8% 的测试准确率，与纯文本朴素贝叶斯训练几乎相同。
- 现实世界中服务器不串通的假设下，本节设计的协议可以安全地抵御半诚实的对手。

# 目录

## CONTENTS

01

隐私保护机器学习



基于安全多方计算  
(STPC) 的隐私保护  
朴素贝叶斯分类

02

论文主要工作



基于函数秘密共享的安  
全多方决策树分类

03

总结和展望





# 基于函数秘密共享的安全两 方决策树分类



# 研究问题



	我们的结果	De Cock 等人 [79]	Lu 等人 [91]
准确度损失	0.05%	2-5%	-
每一次比较的通讯	2 个环元素	n 个环元素	>100 个环元素
安全计算方法	同态, 秘密共享	秘密共享	同态
安全模型	半诚实	恶意	半诚实

表 4-1 与之前工作比较

- 基于 2PC(两方计算)的协议，用于更快更精确的决策树训练和评估
- 在不依赖可信第三方的情况下，两个不同的服务器共享其数据源，并结合其公共数据对密文进行安全决策树训练。
- 解决对来自不同数据源的机密数据进行私有决策树训练的问题。两台服务器的数据特征是公开的，服务器需要结合对方的数据对私有属性向量进行分类。



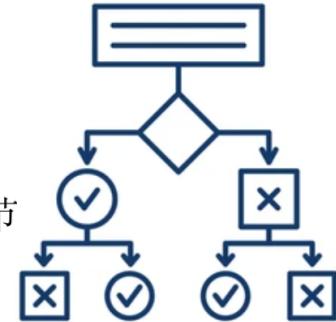
## 基于函数秘密共享的安全两方决策树分类

- 在不依赖可信第三方的情况下，两个不同的服务器共享其数据源，并结合其公共数据对密文进行安全决策树训练
- 利用函数秘密共享(FSS)，提出比较协议选择最佳分割方式
- 实现在线运行时间大约减少 $2\times$ ，同时导致树训练的准确性略有损失。对于通信，本协议仅需要一轮具有2环元素的通信
- 系统在半诚实模型中单方面可模拟和安全



## ■ 决策树分类器

- 输入  $x = (x_1, \dots, x_{n-1})$  的决策树评估由  $\ell^{label} = \mathcal{T}(x)$  和  $\mathcal{T} : \mathbb{Z}^k \leftarrow \{\ell_{label}^1, \dots, \ell_{label}^m\}$ , 其中  $m$  是叶节点的数量。
- 设  $t_j$  为决策节点  $j$  的阈值。如果  $x_{f(j)} \geq t_j$  对于节点  $j$  成立, 则选择右子节点作为下一个决策节点, 否则选择左子节点。
- 函数输出最终叶节点的分类标签  $\ell_{label}$ 。
- 函数  $thr : [0, m - 1] \rightarrow \mathbb{Z}$  为每个决策节点分配阈值。
- 通过函数  $att : [0, m - 1] \rightarrow [0, n - 1]$  将属性索引分配给每个决策节点
- 每个叶节点的标签通过标签函数  $lab : [m, M - 1] \rightarrow \{c_0, \dots, c_{k-1}\}$ 。
- 在分配的阈值和属性值之间进行“大于”比较, 即节点  $v$  处的决策为  $[x_{att(v)} \geq thr(v)]$





## ■ 同态加密

- 基于格的同态加密对明文进行多个加法和乘法

**定义 11.** 考虑定义为环  $\mathbb{Z}_q[X]/(X^N + 1)$  的明文空间，其中  $q$  是素数， $N$  可以表示为  $2$  的幂。正在考虑的同态加密 (HE) 方案包括以下算法：

- $(\text{pk}, \text{sk}, \text{ek}) \leftarrow \text{KGen}(\lambda)$ ：生成私钥  $\text{sk}$ , 公钥  $\text{pk}$ , 评估密钥  $\text{ek}$  是通过  $\text{KGen}(\lambda)$  表示的概率算法实现的。该算法采用安全参数  $\lambda$  来保证生成密钥的随机性和安全性。
- $c \leftarrow \text{Enc}(\text{pk}, m)$ ：采用概率算法的加密算法产生密文  $c$  给定消息  $m$  和公钥  $\text{pk}$ 。将生成的加密表示为  $\llbracket m \rrbracket$ 。
- $c \leftarrow \text{Eval}(\text{ek}, f, c_1, \dots, c_n)$ ：概率算法是用于通过使用评估密钥  $\text{ek}$ 、 $n$  元函数  $f$  和表示为  $c_1$  的  $n$  密文来生成密文  $c_1 \dots c_n$ 。
- $m' \leftarrow \text{Dec}(\text{sk}, c)$ ：消息  $m'$  可以从给定的生成使用确定性算法的密文  $c$  和私钥  $\text{sk}$ 。



## ■ 同态加密

- 基于格的同态加密对明文进行多个加法和乘法
- 噪声水平在乘法时呈指数增长，但添加密文会导致线性增加，如果噪声水平变得太高，则无法解密密文。
- 采用刷新算法，或者可以将函数  $f$  的电路深度保持足够低。



## ■ 函数秘密共享

- 函数秘密共享 (FSS) 的工作原理是将函数  $f$  拆分为两个简洁的函数部分，这样每个部分都不会透露任何有关函数  $f$  的信息，但是当在某个点  $x$  组合评估时，结果为  $f(x)$ 。
- 定义

两方函数秘密共享 (FSS) 方案是一对算法  $(\text{Gen}, \text{Eval})$ ，使得：

- $\text{Gen}(1^\kappa, \hat{f})$  是一个概率多项式时间 (PPT) 密钥生成算法，给定安全参数  $1^\kappa$  和函数  $\text{hat } f \in \{0, 1\}^*$  输出一对键  $(k_0, \dots, k_\sigma)$ 。假设  $\hat{f}$  明确包含输入和输出组  $\mathbb{G}_{\text{in}}, \mathbb{G}_{\text{out}}$  的描述。
- $\text{Eval}(\sigma, k_\sigma, x)$  是一个多项式时间评估算法，给定  $\sigma \in \{0, \dots, m\}$  (部分索引)， $k_\sigma$  被定义为函数  $f_\sigma$  的密钥： $\mathbb{G}_{\text{in}} \leftarrow \mathbb{G}$ 。设  $x \in \mathbb{G}_{\text{in}}$  为函数  $f_\sigma$  的输入，输出群元素  $y_\sigma \in \mathbb{G}_{\text{out}}$ .



## ■ 函数秘密共享

- 函数秘密共享 (FSS) 的工作原理是将函数  $f$  拆分为两个简洁的函数部分，这样每个部分都不会透露任何有关函数  $f$  的信息，但是当在某个点  $x$  组合评估时，结果为  $f(x)$ 。
- 定义
- 正确性和安全性

设  $\mathcal{F} = \{f\}$  为函数族，Leak 为指定允许泄漏的函数关于  $\hat{f}$ 。当省略 Leak 时，理解为仅输出  $\mathbb{G}_{in}$   $\mathbb{G}_{out}$ 。定义 12 中的  $(Gen, Eval)$  是  $\mathcal{F}$  的 FSS 方案（相对于泄漏 Leak）如果满足

- 正确性：对于所有  $\hat{f} \mathbb{G}_{in} \leftarrow \mathbb{G}_{out}$ ，并且每个  $x \in \mathbb{G}_{in}$ ，如果  $(k_0, k_1) \leftarrow Gen(1^\lambda, \hat{f})$  然后
$$\Pr[Eval(0, k_0, x) + Eval(1, k_1, x) = f(x)] = 1$$
- 安全性：对于每个  $\sigma \in \{0, 1\}$  都有一个 PPT 算法  $Sim_\sigma$ （模拟器），例如对于每个序列  $(\hat{f}_\lambda)_{\lambda \in \mathbb{N}}$  来自  $\mathcal{F}$  的多项式大小的函数描述和多项式大小的输入序列  $x_\lambda$  对于  $f_\lambda$ ，输出 Real 和 Ideal 在计算上是无法区分的：
  - $Real_\lambda: (k_0, k_1) \leftarrow Gen(1^\lambda, \hat{f}_\lambda);$  输出  $k_\sigma$ 。
  - $Ideal_\lambda:$  输出  $Sim_\sigma(1^\lambda, Leak(\hat{f}_\lambda))$ 。



## ■ 决策树模型数据结构

● 对于决策树模型  $\mathcal{M} = (\mathcal{T}, \text{thr}, \text{att})$ ，表示每个节点  $v$  的树  $\mathcal{T}$  由组成：

- $v.\text{threshold}$ : 节点  $v$  的阈值，用  $\text{thr}(v)$  表示，存储在变量  $v.\text{threshold}$ 。
- $v.\text{aIndex}$ : 关联的索引，用  $\text{att}(v)$  表示，存储在变量  $v.\text{aIndex}$  中。
- $v.\text{parent}$ : 指向父节点的指针存储在变量  $v.\text{parent}$  中。对于根节点，该指针为空。
- $v.\text{left}$ : 指向左子节点的指针存储在变量  $v.\text{left}$  中。对于叶节点，这些指针为空。
- $v.\text{right}$ : 指向右子节点的指针存储在变量  $v.\text{right}$  中。对于叶节点，这些指针为空。
- $v.\text{cmp}$ : 在树评估过程中，比较位  $b \leftarrow [x_{\text{att}(v.\text{parent})} \geq x_{\text{thr}(v.\text{parent})}]$  被计算并存储在变量  $v.\text{cmp}$  中。如果  $v$  是右节点，则存储  $b$ ，否则存储  $1 - b$ 。
- $v.\text{cLabel}$ : 如果  $v$  是叶节点，则分类标签存储在变量  $v.\text{cLabel}$  中；否则，它存储一个空字符串。



## ■ 决策树模型数据结构

●(分类函数). 设属性向量为  $x = (x_0, \dots, x_{n-1})$ , 决策树模型为  $\mathcal{M} = (\mathcal{D}, \mathcal{L})$ 。将分类函数定义为

$$f_c(x, \mathcal{M}) = \text{tr}(x, \text{root})$$

其中  $\text{root}$  是根节点,  $\text{tr}$  是遍历函数, 定义为:

$$\text{tr}(x, v) = \begin{cases} \text{tr}(x, v.\text{left}) & \text{if } v \in \mathcal{D} \text{ and } x_{v.\text{Index}} < v.\text{threshold} \\ \text{tr}(x, v.\text{right}) & \text{if } v \in \mathcal{D} \text{ and } x_{v.\text{Index}} \geq v.\text{threshold} \\ v & \text{if } v \in \mathcal{L} \end{cases}$$



## ■ 决策树模型模块构建

- 初始化决策位：每个节点  $v \in \mathcal{D}$  比较位  $b \leftarrow [x_{\text{att}(v)} \geq \text{thr}(v)]$  并将  $b$  存储在右子节点 ( $v.\text{right}.\text{cmp} = 1 - b$ )

---

### Algorithm 6 决策位计算

---

- 1: 函数 EVALDNODE( $\mathcal{D}, [x]$ )
  - 2: 对每个  $v \in \mathcal{D}$  遍历
  - 3:    $[b] \leftarrow [[x_{v.\text{Index}} \geq v.\text{threshold}]]$
  - 4:    $[v.\text{right}.\text{cmp}] \leftarrow [b]$
  - 5:    $[v.\text{left}.\text{cmp}] \leftarrow [1 - b]$
-



## ■ 决策树模型模块构建

- 初始化决策位：每个节点  $v \in \mathcal{D}$  比较位  $b \leftarrow [x_{att(v)} \geq thr(v)]$  并将 b 存储在右子节点 ( $v.right cmp = 1 - b$ )
- 聚合决策位：对于每个叶节点 v，服务器沿着从根到 v 的路径聚合比较位。使用队列并在 BFS 中遍历树来实现它，

---

### Algorithm 7 聚合决策位

---

```
1: 函数 EVALPATHS( $\mathcal{D}, \mathcal{L}$ )
2:   令  $Q$  是一个队列
3:    $Q.enqueue(root)$ 
4:   判断  $Q.empty = false$  遍历
5:      $v \leftarrow Q.dequeue()$ 
6:      $\llbracket v.left.cmp \rrbracket \leftarrow \llbracket v.left.cmp \rrbracket \odot \llbracket v.cmp \rrbracket$ 
7:      $\llbracket v.right.cmp \rrbracket \leftarrow \llbracket v.right.cmp \rrbracket \odot \llbracket v.cmp \rrbracket$ 
8:     如果  $v.left \in \mathcal{D}$  那么
9:        $Q.enqueue(v.left)$ 
10:    如果  $v.right \in \mathcal{D}$  那么
11:       $Q.enqueue(v.right)$ 
```

---



## ■ 决策树模型模块构建

● 初始化决策位：每个节点  $v \in \mathcal{D}$  比较位  $b \leftarrow [x_{att(v)} \geq thr(v)]$  并将  $b$  存储在右子节点 ( $v.right cmp = 1 - b$ )

● 聚合决策位：对于每个叶节点  $v$ ，服务器沿着从根到  $v$  的路径聚合比较位。使用队列并在 BFS 中遍历树来实现它，

● 计算决策位：比较位  $b \leftarrow [x_{att(v)} \geq thr(v)]$  并将  $b$  存储在右子节点 ( $v.right cmp = b$ ) 并将  $1 - b$  存储左子节点 ( $v.left cmp = 1 - b$ )。

---

### Algorithm 6 决策位计算

---

- 1: 函数 EVALDNODE( $\mathcal{D}, [x]$ )
  - 2: 对每个  $v \in \mathcal{D}$  遍历
  - 3:  $[b] \leftarrow [[x_{v.Index} \geq v.threshold]]$
  - 4:  $[[v.right cmp]] \leftarrow [b]$
  - 5:  $[[v.left cmp]] \leftarrow [1 - b]$
-



## ■ 决策树模型模块构建

- 初始化决策位
- 聚合决策位
- 计算决策位
- 分类：沿着到叶子节点的路径聚合决策位后，每个叶子节点  $v$  存储  $v.cmp = 0$  或  $v.cmp = 1$ 。然后，服务器通过计算每个叶子  $v$  的值  $[v.cmp] \oplus [v.cLabel]$  并对所有结果求和。

---

### Algorithm 8 分类

---

- 1: 函数 FINALIZE( $\mathcal{L}$ )
- 2:  $[\![\text{result}]\!] \leftarrow [\![0]\!]$
- 3: 对每个  $v \in \mathcal{L}$  遍历
- 4:  $[\![\text{result}]\!] \leftarrow [\![\text{result}]\!] \oplus ([\![v.cmp]\!] \odot [\![v.cLabel]\!])$
- 5: 返回  $[\![\text{result}]\!]$

---



## ■ 决策树模型模块构建

- 初始化决策位
- 聚合决策位
- 计算决策位
- 分类
- 比较：输入 秘密共享，而不是输入的公共值，满足基于 FSS 的特定比较协议  $\text{Compare}([\mathbf{x}], [\mathbf{y}])$ ，输出  $\mathbf{z} = \mathbf{1}\{\mathbf{y} > \mathbf{x}\}$  的共享份额。

---

### Algorithm 9 安全比较

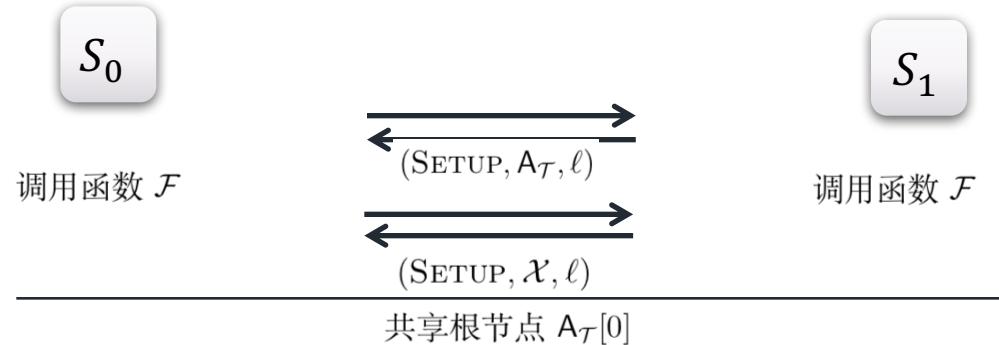
---

- 1: 函数  $\text{COMPARE}([\mathbf{x}], [\mathbf{y}])$
  - 2:  $S_i$  用 PRFs 做种子生成  $[\mathbf{r}]_0$  。
  - 3:  $S_i$  随机取样  $r \in \mathbb{Z}_{z^n}$  并发送  $[\mathbf{r}]_1 = r - [\mathbf{r}]_0$  给  $S_{1-i}$
  - 4:  $S_i$  估计  $(k_0, k_1) \leftarrow \text{Gen}_{r,1}$  并发送  $k_i$  给  $S_i$
  - 5:  $S_i$  发送  $[\mathbf{y}]_i - [\mathbf{x}]_i + [\mathbf{r}]_i$  to  $S_{1-i}$ , 并重构  $y - x + r$ .
  - 6:  $S_i$  评估  $[\mathbf{z}]_i \leftarrow \text{Eval}_{r,1}(i, k_i, y - x + r)$
  - 7: 返回  $[\mathbf{z}]$
-



## 安全两方决策树分类

参数：计算安全函数  $\mathcal{F}$ 。 $S_i$  提供一棵树  $\mathcal{T}$ ，其中  $i \in \{0, 1\}$ 。 $S_i$  提供特征向量  $\mathcal{X}$ ，以及最长深度  $d$ 。





## 安全两方决策树分类

**参数:** 计算安全函数  $\mathcal{F}$ 。 $S_i$  提供一棵树  $T$ , 其中  $i \in \{0, 1\}$ 。 $S_i$  提供特征向量  $\mathcal{X}$ , 以及最长深度  $d$ 。

$S_0$

共享根节点  $A_T[0]$

$S_1$

1. 循环  $j \in [1, d]$

(a)  $S_i$  发送  $(\text{EVAL}, \llbracket v \rrbracket_i)$  且  $S_{1-i}$  发送  $(\text{EVAL}, \llbracket v \rrbracket_{1-i})$  到 FSS 函数  $\mathcal{F}$ 。

最终双方共享  $\langle \mathcal{X}[v] \rangle$ 。

(b)  $S_i$  执行安全比较算法 9 来计算  $\llbracket b \rrbracket \leftarrow \llbracket \mathcal{X}[v] \rrbracket > \llbracket t \rrbracket$ 。

(c)  $S_i$  计算下一个树节点的索引  $\llbracket \text{idx} \rrbracket \leftarrow \llbracket v.\text{left} \rrbracket \oplus \llbracket b \rrbracket \cdot (\llbracket v.\text{left} \rrbracket \oplus \llbracket v.\text{right} \rrbracket)$ 。

(d)  $S_i$  发送  $\text{EVAL}[\text{idx}]_i$ ,  $S_{1-i}$  发送  $\text{EVAL}[v]_{1-i}$  到  $F$ 。最终双方共享  $\llbracket A_T[\text{idx}] \rrbracket$

(e) 更新  $(\llbracket v.\text{threshold} \rrbracket, \llbracket v.\text{left} \rrbracket, \llbracket v.\text{right} \rrbracket, \llbracket v \rrbracket, \llbracket v.\text{cLabel} \rrbracket) \leftarrow \llbracket A_T[\text{idx}] \rrbracket$ 。

(f) 设置  $\llbracket \text{rst} \rrbracket \leftarrow \llbracket v.\text{cLabel} \rrbracket$ 。

2.  $S_i$  和  $S_{1-i}$  将  $\text{rst}$  显示为  $S_i$  作为输出。



## 安全两方决策树分类安全性分析

- 正确性 假设底层  $FSS$  方案的评估正确性，以及的  $\text{Eval}$  算法，那么上面的构建是一个双服务器私有决策树分类协议，输出正确的分类标签。

满足  $result_\ell = r_1^\ell \times 0 + \ell_{label} = \ell_{label}$ 。

- 安全性 算法  $\text{Compare}(\llbracket x \rrbracket, \llbracket y \rrbracket)$  安全地实现了功能  $\mathcal{F}_{\text{Compare}}$ ，假设存在  $FSS$  过程的  $FSS$  过程的安全协议。



## 实验结果

	批次	的方案	明文
准确度	5	74.81%	75.13%
	10	75.19%	75.56%
	15	75.37%	75.94%
时间 (小时)	5	0.24	0.09
	10	0.47	0.17
	15	0.73	0.29

表 4-2 安全两方决策树分类的性能

- 安全训练准确率的变化趋势与明文训练准确率的变化趋势相似，没有明显的波动准确率之差约为 $\pm 0.05\%$

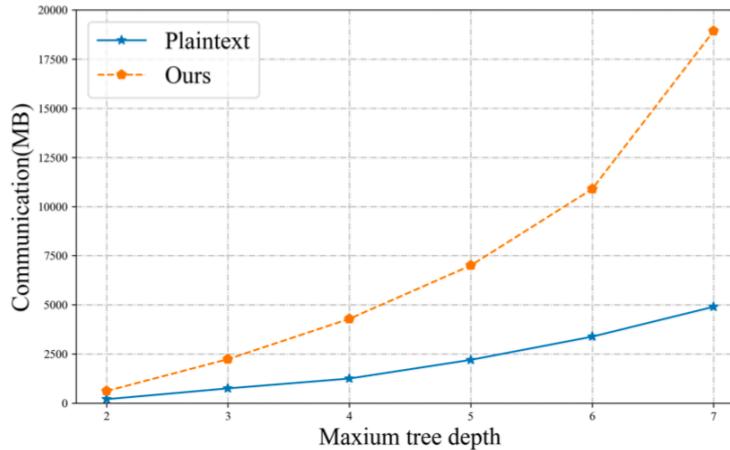
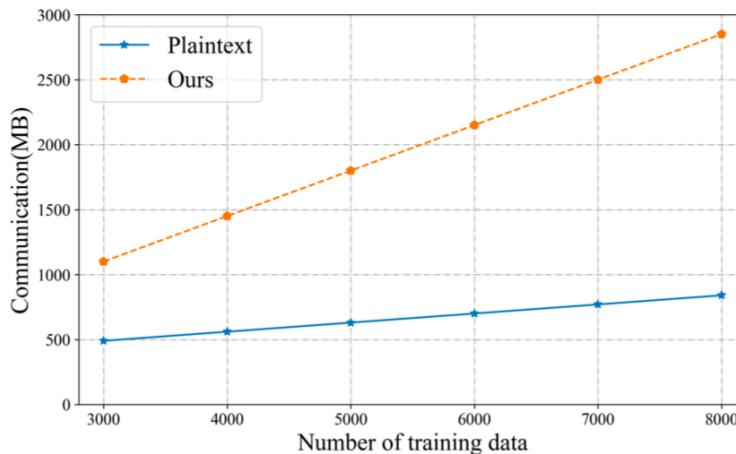


## 实验结果

	批次	的方案	明文
准确度	5	74.81%	75.13%
	10	75.19%	75.56%
	15	75.37%	75.94%
时间 (小时)	5	0.24	0.09
	10	0.47	0.17
	15	0.73	0.29

表 4-2 安全两方决策树分类的性能

- 安全训练准确率的变化趋势与明文训练准确率的变化趋势相似，没有明显的波动准确率之差约为 $\pm 0.05\%$
- 训练样本数量的增加，数据和通信的成本大致呈线性增长
- 通信开销受到训练样本数量和最大树深度等关键因素的影响



# 目录

## CONTENTS

01

隐私保护机器学习



基于安全多方计算  
(STPC) 的隐私保护  
朴素贝叶斯分类

02

论文主要工作



基于函数秘密共享的安  
全多方决策树分类

03

总结和展望



基于门限加法同态加密  
的安全多方线性聚合机  
器学习



基于门限加法同态加密的安  
全两方线性聚合机器学习



## 基于门限加法同态加密的安全两方线性聚合机器学习

- 提出了使用去中心化阈值加性同态加密(DTAHE) 的安全线性聚合算法，使机器学习模型能够对加密的用户输入执行线性操作。
- 允许服务器将用户输入乘以系数并对它们求和以构建神经网络层，同时保护 数据隐私。
- 为 DTAHE 和安全两方线性聚合机器学习算法提供安全定义和证明。



## 问题描述

- 对多项式环中的元素的系数使用大模数构造去中心化阈值加性同态加密(噪声与多项式次数成正比)
- 将许多秘密共享分发给用户，依赖于具有通用访问结构的秘密共享方案构建去中心化同态加密(通信开销太高)
- 使用 ElGamal 加密作为基本构建(难以分布式方式生成共享密钥对)
- 提供两个独立服务器安全的线性聚合算法来丰富参数服务器



# 去中心化阈值加同态加密 ( DTAHE )

DTAHE 方案是概率多项式时间 算 法的元组

$$DTAHE = (Init, KeyGen, ShareSec, AggKey, Enc, Eval, ParDec, FinDec)$$

1.  $Init(1^\lambda) \rightarrow para$ : 初始化函数输入安全参数  $\lambda$ , 输出系统参数  $para$ 。
2.  $KeyGen(para) \rightarrow (sk_u, pk_u)$ : 密钥生成函数输入系统参数  $para$ , 输出用户  $u$  生成公钥  $pk_u$  和私钥  $sk_u$ 。
3.  $ShareSec(para, \{pk_v\}_{v \in U \setminus \{u\}}, t, sk_u) \rightarrow \{e_{v,u}\}_{v \in U \setminus \{u\}}$ : 输入系统参数  $para$ 、集合  $U$  中除用户  $u$  之外的用户  $v$  之外的用户的公钥、阈值  $t$  和用户  $u$  的私钥  $sk_u$  , 为每个用户  $v \in U \setminus \{u\}$  生成加密共享  $e_{v,u}$ 。
4.  $AggKey(para, \{pk_u\}_{u \in U}) \rightarrow pk$ : 密钥聚合将系统参数  $para$  和  $U$  中一组用户的公钥作为输入, 并生成加密密钥  $pk$ 。
5.  $Enc(para, pk, m_u) \rightarrow c_u$ : 加密函数输入系统参数  $para$ 、公钥  $pk$  和来自用户  $u$  的消息  $m_u$ , 生成密文  $c_u$ 。
6.  $Eval(para, \{c_u\}_{u \in U}, \{\alpha_u\}_{u \in U}) \rightarrow \hat{c}$ : 评估函数输入系统参数  $para$ , 密文  $\{c_u\}_{u \in U}$  和参数  $\{\alpha_u\}_{u \in U}$ , 并生成评估密文  $\hat{c} = \sum_{u \in U} \alpha_u \cdot c_u$ 。
7.  $ParDec(para, \hat{c}, \{e_{u,v}\}_{v \in U}) \rightarrow \hat{m}_u$ : 参数解密函数输入系统参数  $para$ 、 $\hat{c}$  以及一组加密共享  $\{e_{u,v}\}_{v \in U \setminus \{u\}}$  给 用户  $u$  , 并产生部分解密的值  $\hat{m}_u$ 。
8.  $FinDec(para, t, \hat{c}, \{\hat{m}_u\}_{u \in V}) \rightarrow m$ : 输入系统参数  $para$ 、阈值  $t$ 、参数密文  $\hat{c}$  和来自用户集合  $V$  和  $|V| \geq t$  部分解密密文  $\{\hat{m}_u\}_{u \in V}$  , 最后输出明文  $m$ 。



## DTAHE 正确性

如果对于所有  $\lambda$  和  $t$ , 以下内容成立, 则一组用户 DTAHE 方案满足评估正确性: 对于评估密文  $\hat{c} \leftarrow Eval(para, \{c_u\}_{u \in U}, \{\alpha_u\}_{u \in U})$  的概率  $\Pr[FinDec(para, t, \hat{c}, ParDec(para, \hat{c}, \{e_{u,v}\}_{v \in U})\}_{u \in V}) = \sum \alpha_u \cdot m_u]$  是 压倒性的, 从这里可以推导出

- 生成密文  $c_u \leftarrow Enc(para, pk, m_u)$
- 输出秘密共享  $(\{e_{v,u}\}_{v \in U}) \leftarrow ShareSec(para, \{pk_v\}_{v \in U \setminus \{u\}}, t, sk_u)$
- 用户  $u$  的私钥和公钥生成  $(sk_u, pk_u) \leftarrow KeyGen(para)$
- 输出加密密钥  $pk \leftarrow AggKey(para, \{pk_u\}_{u \in U})$
- 系统参数  $para \leftarrow Init(1^\lambda)$ 。



# 安全两方线性聚合算法

---

**Algorithm 12 安全两方线性聚合**

---

- 1: **密钥生成:**  $S_0$  生成密钥对  $(pk_A, sk_A) \leftarrow KeyGen(1^k)$ ,  $S_1$  生成密钥对  $(pk_B, sk_B) \leftarrow KeyGen(1^k)$ 。 $S_0$  和  $S_1$  交换公开密钥  $pk_A, pk_B$ 。
  - 2:  **$S_0$  生成密文:**  $S_0$  用它的私钥  $x_A$ ,  $S_0$  计算  $[x_A] \leftarrow Enc_{pk_A}(x_A)$
  - 3:  **$S_1$  生成密文:**  $S_1$  用它的私钥  $x_B$ ,  $S_1$  计算  $[x_B] \leftarrow Enc_{pk_B}(x_B)$
  - 4: **交换密文:**  $S_0$  发送  $[x_A]$  给  $S_1$  并从  $S_1$  接收  $[x_B]$ ,  $S_1$  发送  $[x_B]$  给  $S_0$  并从  $S_0$  接收  $[x_A]$ 。
  - 5: **本地聚合:**  $S_0$  计算  $[x_A + x_B] \leftarrow [x_A] \cdot [x_B]$ ,  $S_1$  计算  $[x_A + x_B] \leftarrow [x_A] \cdot [x_B]$ 。
  - 6: **门限解密:**  $S_0$  和  $S_1$  参与阈值解密算法来解密  $[x_A + x_B]$  并揭示  $x_A + x_B$
-



## 性能分析

- 在第 1 轮(AdvertiseAccounts)中，每个用户向服务器发送一条包含其帐户、时间戳、签名和证书的消息。服务器响应一条消息，其中包含所有用户的帐户和会话 ID。所以这里的通信量是  $O(n)$ ，其中  $n$  是用户数量。
- 在第 2 轮(ShareKeys)中，每个用户在区块链上查找公钥并将加密的共享发送到服务器。服务器使用其他用户的加密共享来响应每个用户。由于每个用户接收来自所有其他用户的共享，因此通信时间复杂度为  $O(n^2)$ 。
- 在第 3 轮(CipherCollection)中，每个用户在交易中将其加密输入发送到区块链。服务器聚合并发回一个经过评估的密文交易。这是  $O(n)$ 。
- 在第 4 轮(解密)中，每个用户将其部分解密的值发送到服务器。服务器聚合并解密得到最终结果。这是  $O(n)$  的通信。

# 目录

## CONTENTS

01

隐私保护机器学习

02

论文主要工作

03

总结和展望

## 总结和展望

- 总结：提出了在隐私保护机器学习应用中常用的双服务器设置中进行隐私保护朴素贝叶斯分类和决策树评估的有效协议。通过将 MPC 专门用于训练和分类所需的计算，我们的工作实现了适合提出了基于算术共享和高效 Beaver 三重乘法的机密朴素贝叶斯学习的双服务器协议。
- 未来工作：协议优化、高维数据、恶意安全、硬件加速、差分隐私

## 在校期间发的论文

1. Kun Liu , Chunming Tang. Secure Two-party Decision Tree Classification based on Function Secret Sharing[J]. Complexity, 2023. (SCI , 已发表)
  
2. Kun Liu , Chunming Tang. Privacy-Preserving Naive Bayes Classification Based on Secure Two-party Computation[J]. AIMS Mathematics, 2023, 8(10), 24825- 24847. (SCI , 已发表)



感谢各位专家批评指正  
THANK YOU FOR WATCHING