

面向深度神经网络的安全计算协议设计方法

毕仁万¹, 陈前昕¹, 熊金波¹, 刘西蒙²

(1. 福建师范大学数学与信息学院, 福建 福州 350117;

2. 福州大学数学与计算机科学学院, 福建 福州 350108)

摘要: 针对深度学习神经网络模型计算过程中存在的信息泄露问题, 结合加性秘密共享方案, 在两台非共谋的边缘服务器间设计安全高效的交互计算协议。考虑到非线性函数不能直接拆分, 首先提出一组基本转换协议, 实现加性副本和乘性副本的安全转换, 经过少量调用, 可以安全计算幂函数、比较、指数、对数、除法、除法等底层函数。由于数据传递和计算特点, 协议可以扩展至数组计算。理论分析证明了协议的正确性、高效性和安全性, 实验结果表明, 协议具有较小的误差, 其计算和通信开销均优于现有设计方案。

关键词: 深度学习神经网络; 加性秘密共享; 安全计算协议; 加法-乘法转换; 数组单元

中图分类号: TP393

文献标识码: A

doi: 10.11959/j.issn.2096-109x.2020050

Design method of secure computing protocol for deep neural network

BI Renwan¹, CHEN Qianxin¹, XIONG Jinbo¹, LIU Ximeng²

1. College of Mathematics and Informatics, Fujian Normal University, Fuzhou 350117, China

2. College of Mathematics and computer science, Fuzhou University, Fuzhou 350108, China

Abstract: Aiming at the information leakage problem in the process of deep neural network model calculation, a series of secure and efficient interactive computing protocols were designed between two non-collusive edge servers in combination with the additive secret sharing scheme. Since the nonlinear function cannot be split directly, a set of basic conversion protocols were proposed to realize the secure conversion of additive and multiplicative shares. After a few invokes, the power, comparison, exponential, logarithm, division and other low-level functions can be calculated securely. Due to the characteristics of data transfer and computation, the proposed protocols can be extended

收稿日期: 2020-04-17; 修回日期: 2020-07-03

通信作者: 熊金波, jinbo810@163.com

基金项目: 国家自然科学基金(61872088, U1804263, 61702105, 61872090); 福建省自然科学基金资助项目(2019J01276); 贵州省公共大数据重点实验室开放课题(2019BDKFJJ004)

Foundation Items: The National Natural Science Foundation of China (61872088, U1804263, 61702105, 61872090), The Natural Science Foundation of Fujian Province, China (2019J01276), The Guizhou Provincial Key Laboratory of Public Big Data Research Fund (2019BDKFJJ004)

论文引用格式: 毕仁万, 陈前昕, 熊金波, 等. 面向深度神经网络的安全计算协议设计方法[J]. 网络与信息安全学报, 2020, 6(4): 130-139.

BI R W, CHEN Q X, XIONG J B, et al. Design method of secure computing protocol for deep neural network[J]. Chinese Journal of Network and Information Security, 2020, 6(4): 130-139.

to array computation. Theoretical analysis ensures the correctness, efficiency and security of these protocols. The experimental results show that the error of these protocols is negligible, and the computational costs and communication overhead are better than the existing schemes.

Key words: deep neural network, additive secret sharing, secure computing protocol, add-multiply transformation, array unit

1 引言

随着人工智能和物联网时代的全面到来,智能终端设备被赋予自主采集各种类型数据的能力,如高清摄像机捕捉图像数据、声学设备采录语音数据、LiDAR 设备探测点云数据^[1]等。为了进一步挖掘、分析出有意义的信息,相比传统的机器学习算法,采用深度神经网络模型处理数据可以获得更高的预测精度。例如, Ren 等^[2]利用卷积神经网络提取图像目标的特征信息, Zhou 等^[1]采用稀疏卷积神经网络提取云目标的特征信息等。然而,深度神经网络由于模型深度、参数数量等特征会耗费高额的计算开销,资源受限的智能终端选择将数据和任务需求提交给边缘服务器进行处理。随着数据控制权的迁移,包含在数据内的隐私信息存在极高的泄露风险,如何在不泄露隐私信息的前提下快速准确地将原始数据分析转化为有价值的内容,是当前应用研究的热点。

实际上,隐私信息是否安全可以归结于网络函数计算的安全性,目前,主要结合同态加密^[3]和加性秘密共享^[4]方案构思解决方法。同态加密概念自提出以来被广泛应用于密文处理, Hesamifard 等^[5]通过构造非线性函数的低阶近似多项式,采用线性同态加密可以实现线性修正单元(ReLU)、双曲正切(Tanh)、Sigmoid 等网络激活函数的密态计算,但烦琐的加密计算开销难以满足智能终端低延迟性需求。针对这一问题, Mohassel 等^[6]最早将加性秘密共享概念引入线性回归、逻辑回归和神经网络等模型训练任务中,基于安全两方计算^[7]提出了一系列隐私保护协议,相比同态加密算法提高了计算效率。随后,唐春明等^[8]针对两台非共谋服务器训练线性回归模型过程中的信息泄露问题,结合加性秘密共享设计了安全加法、乘法交互计算协议,并将 Sigmoid 激活函数中的指数函数展开为泰勒多项式形式,利用加法和乘法计算协议可以获得其近似值。然而,这些协议采用不经意传输机制^[9]传

递数据,并且需要多轮近似迭代计算,计算效率较低,并且没有分析协议的误差范围。显然,神经网络模型的正确性和效率依赖于底层计算函数,刘新等^[10]探索基本初等函数的安全高效计算方法,终端将模糊化的数据传递给云服务器,由两者共同计算目标函数的结果,计算复杂度降低为常数阶,并且设计的计算协议可以扩展至数组计算,但计算过程需要终端同步参与数据传递。

近年来,研究学者结合加性秘密共享在隐私保护神经网络方面取得较大进展, Huang 等^[11]提出了一种隐私保护图像特征提取方案,设计的安全比较函数可以正确实现 ReLU 激活函数功能。 Ma 等^[12]提出了一种隐私保护语音识别方案,利用长短期记忆网络提取关联语音特征信息,设计的安全 Sigmoid 函数和 Tanh 函数可以实现门控的迭代计算。 Liu 等^[13]提出了一种隐私保护图像目标检测方案,设计的安全指数、对数和倒数计算协议可以正确实现检测网络的底层函数功能。这些安全计算协议可以为网络模型提供足够的安全性,遗憾的是,需要多轮迭代^[14]降低误差以达到计算精度需求,计算开销依赖于迭代次数。

为了克服上述安全计算协议的缺陷,本文面向深度神经网络底层函数探索安全高效的计算协议设计方法,主要贡献总结如下。

1) 考虑到非线性函数拆分困难的特性,设计了安全乘法-加法转换(STMA, secure transforming multiply-add)和安全加法-乘法转换(STAM, secure transforming add-multiply)协议,实现加性副本与乘性副本的安全等值转换。

2) 在安全转换协议的基础上,结合基本运算性质可以实现对非线性函数的计算拆分,设计的安全幂函数、比较、指数、对数、除法等计算协议不需要引入复杂迭代操作,并且可以扩展至数组计算。协议的输入和输出仅需要加法计算完成拆分与合并,构建安全的深度神经网络模型可以自由选择 and 替换固有的函数模块。

3) 详细的理论分析证明了所提协议的正确性

和安全性, 并且计算和通信复杂度可以控制在常数阶。实际性能测试结果表明, 协议的误差在可接受范围内, 计算和通信效率均优于最新的研究工作。

2 系统模型与安全模型

2.1 系统模型

本文系统模型包含 5 个实体: 发送终端 $\mathcal{D}_1$ 、接收终端 $\mathcal{D}_2$ 、可信第三方服务器 $\mathcal{T}$ 以及两台边缘服务器 $\mathcal{S}_1$ 和 $\mathcal{S}_2$, 如图 1 所示, 其中智能终端 $\mathcal{D}_1$ 和 $\mathcal{D}_2$ 根据发送数据和接收数据进行区分, $\mathcal{T}$ 、 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 负责执行安全计算协议, 具体如下。

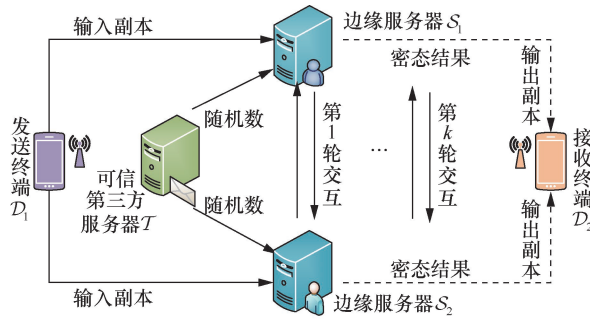


图 1 系统模型
Figure 1 System model

1) $\mathcal{D}_1$ 负责以加法形式随机拆分数据, 并将两份输入副本和计算需求提交给 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 。

2) $\mathcal{T}$ 负责为 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 生成和传递随机数, 不直接参与协议计算, 因此可以离线执行。

3) $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 负责在线执行安全计算协议, 将 $\mathcal{D}_1$ 发送的输入副本隐藏在随机数中, 双方利用密态结果进行 k 轮关联交互, 计算任务完成后将输出副本发送给 $\mathcal{D}_2$ 。

4) 接收到两份输出副本后, $\mathcal{D}_2$ 只需要执行简单的加法就可以获得完整的输出结果。

2.2 安全模型

在系统模型中, 考虑隐私信息为计算协议完整的输入和输出, 若协议双方($\mathcal{S}_1$ 和 $\mathcal{S}_2$)和潜在的概率多项式时间敌手 $\mathcal{A}$ 均不能获得隐私信息, 则认为计算协议是安全的。由于 $\mathcal{T}$ 不直接参与 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 间的交互计算, 不拥有输入和输出副本, 因此不会影响计算协议的安全性。假设 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 是不共谋的, 不能直接传递己方拥有的副本, 即不能直接共谋获取隐私信息。并且, $\mathcal{T}$ 、 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 之间均通

过安全信道传递信息。类似于之前工作^[11-13, 15-16]中的安全模型, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 为半可信的服务器, 严格执行计算协议步骤, 不会中途恶意掉线或传递虚假数据, 同时, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 被允许记录所有的中间结果, 期望在计算协议完成后推断出完整的输入和输出信息。此外, $\mathcal{A}$ 具备下述能力: 不能干扰协议参与方的正常信息传递; 不能同时窃取 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 拥有的信息; 至多窃取一条传递输入或输出的通信链路。值得注意的是, 这种约束被认为是典型的, 文献[11-13]中也有类似的敌手约束定义。

3 基本安全转换协议

接收输入 ξ_1 和 ξ_2 ($\xi = \xi_1 + \xi_2$), $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 交互计算函数 $g(\xi)$, 各自输出 $g_1(\xi_1)$ 和 $g_2(\xi_2)$, 满足 $g(\xi) = g_1(\xi_1) + g_2(\xi_2)$ 。显然, 一般非线性函数不能直接拆分, 却可以转化为乘法与加法组合的形式, 如 $g_{(+)}(g'_{(+)}(\xi_1, \xi_2))$ 和 $g'_{(+)}(g_{(+)}(\xi_1, \xi_2))$ 等, $g'_{(+)}$ 表示输入 ξ_1 和 ξ_2 以加法形式连接, $g_{(+)}$ 表示输入 ξ_1 和 ξ_2 以乘法形式连接, 如果 ξ_1 和 ξ_2 可以在加法与乘法形式之间自由转换, 直接拆分非线性函数将成为可能。针对这种需求, 本节介绍一组安全转换协议及其构造过程。

3.1 安全乘法-加法转换协议

考虑到浮点数的小数点浮动、存储格式等, 本文采用固定点格式存储数据。对于固定点数据 u , 计算 $\bar{u} = [u \cdot 10^p]$ 且 $\bar{u} \in \mathbb{Z}_m$, $[\cdot]$ 为取整操作, 以整数形式参与协议计算。一般来说, 为了维持计算协议精度, 同时预防数据产生溢出, 习惯取 $5 \leq p \leq 10$ 。若没有特殊说明, 下文将省略“上横线”标识, 详细的误差影响在第 6 节进行讨论。

已知 $u_1 \in \mathbb{Z}_m$ 和 $u_2 \in \mathbb{Z}_m$, 分别发送给 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 作为输入, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 协同执行 STMA 协议得到输出 f_1 和 f_2 , 满足 $f_1 + f_2 = u_1 \cdot u_2$, 实现了乘法输入形式到加法输出形式的转换。如协议 1 所示, $\mathcal{T}$ 建立加法与乘法运算间的联系, 随机生成一组随机数, 使 $c_1 + c_2 = a_1 \cdot a_2$, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 将输入副本 u_1 和 u_2 隐藏在 p_1 和 p_2 中进行传递, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 只需要独立计算 f_1 和 f_2 。

协议 1 安全乘法-加法转换

输入 $\mathcal{S}_1$ 拥有 $u_1 \in \mathbb{Z}_m$, $\mathcal{S}_2$ 拥有 $u_2 \in \mathbb{Z}_m$

输出 $\mathcal{S}_1$ 返回 f_1 , $\mathcal{S}_2$ 返回 f_2

1) $\mathcal{T}$ 随机生成 $a_1, a_2 \in \mathbb{Z}_m$, 计算 $c \leftarrow a_1 \cdot a_2$, 并随机拆分 $c \leftarrow c_1 + c_2$, 将 a_1 和 c_1 发送给 $\mathcal{S}_1$, 将 a_2 和 c_2 发送给 $\mathcal{S}_2$;

2) $\mathcal{S}_1$ 计算 $p_1 \leftarrow u_1 - a_1$, 并发送 p_1 给 $\mathcal{S}_2$;

3) $\mathcal{S}_2$ 计算 $p_2 \leftarrow u_2 - a_2$, 并发送 p_2 给 $\mathcal{S}_1$;

4) $\mathcal{S}_1$ 返回 $f_1 \leftarrow c_1 + a_1 \cdot p_2$, $\mathcal{S}_2$ 返回 $f_2 \leftarrow c_2 + a_2 \cdot p_1 + p_1 \cdot p_2$ 。

3.2 安全加法-乘法转换协议

已知 $u_1 \in \mathbb{Z}_m$ 和 $u_2 \in \mathbb{Z}_m$, 分别发送给 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 作为输入, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 协同执行安全加法-乘法转换协议得到输出 f_1 和 f_2 , 满足 $f_1 \cdot f_2 = u_1 + u_2$ 。STAM 协议是 STMA 协议的逆操作, 实现了加法输入形式到乘法输出形式的转换。如协议 2 所示, $\mathcal{T}$ 同样需要随机生成 $c_1 + c_2 = a_1 \cdot a_2$ 关联加法与乘法运算, $\mathcal{S}_1$ 将 u_1 隐藏在 p_1 中传递给 $\mathcal{S}_2$, $\mathcal{S}_2$ 随机生成非零分母 f_2 混淆 u_2 , $\mathcal{S}_1$ 获得 λ' 和 λ'' , 可以计算得到 $u_1 + u_2$ 与 f_2 的除法结果。注意, 步骤 3 的除法操作存在不能整除的情况, 因此需要引入 $\lceil \cdot \rceil$ 取整操作。实际上, 这意味着 f_1 在第 P 位小数存在误差, 可以表示为 $f_1 \cdot f_2 = (u_1 + u_2) \pm \varepsilon_p$, 若 ε_p 可以忽略, 那么有理由认为协议输出满足 $f_1 \cdot f_2 = u_1 + u_2$ 。

协议 2 安全加法-乘法转换

输入 $\mathcal{S}_1$ 拥有 $u_1 \in \mathbb{Z}_m$, $\mathcal{S}_2$ 拥有 $u_2 \in \mathbb{Z}_m$

输出 $\mathcal{S}_1$ 返回 f_1 , $\mathcal{S}_2$ 返回 f_2

1) $\mathcal{T}$ 随机生成 $a_1, a_2 \in \mathbb{Z}_m$, 计算 $c \leftarrow a_1 \cdot a_2$, 并随机拆分 $c \leftarrow c_1 + c_2$, 将 a_1 和 c_1 发送给 $\mathcal{S}_1$, 将 a_2 和 c_2 发送给 $\mathcal{S}_2$;

2) $\mathcal{S}_1$ 计算 $p_1 \leftarrow u_1 - a_1$, 并发送 p_1 给 $\mathcal{S}_2$;

3) $\mathcal{S}_2$ 随机选择 $f_2 \in \frac{\mathbb{Z}_m}{\{0\}}$, 计算 $\lambda' \leftarrow \left\lceil \frac{(p_1 + u_2)}{f_2} \right\rceil - c_2$ 和 $\lambda'' \leftarrow \left\lceil \frac{1}{f_2} \right\rceil + a_2$, 并将 λ' 和 λ'' 发送给 $\mathcal{S}_1$;

4) $\mathcal{S}_1$ 返回 $f_1 \leftarrow \lambda' + a_1 \cdot \lambda'' - c_1$, $\mathcal{S}_2$ 返回 f_2 。

4 安全协议设计方法

本节针对深度神经网络中常用的幂函数、比较函数、指数函数、对数函数、除法函数设计对应的安全计算协议。类似于第 3 节, 协议输入

$\{x, y\}$ 在参与计算之前, 通过乘法和取整操作保留 P 位小数, 转化为整数表示。

4.1 安全幂计算协议

幂函数的表达式为 $z = x^\sigma$, 输入 $x \in \mathbb{R}$, $\sigma \in \mathbb{R}$ 为指数。特别地, 若 $\sigma = \frac{1}{2}$, 则表现为平方根函数, 在卷积神经网络中充当 Square 激活函数的核心计算。完整输入 x 随机拆分为 x_1 和 x_2 分发给 $\mathcal{S}_1$ 和 $\mathcal{S}_2$, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 输出 z_1 和 z_2 , 满足 $z_1 + z_2 = (x_1 + x_2)^\sigma$ 。安全幂计算 (SPow, secure power calculation) 协议描述如下:

1) $\mathcal{S}_1$ 协同 $\mathcal{S}_2$ 计算 $(\zeta_1, \zeta_2) \leftarrow \text{STAM}(x_1, x_2)$;

2) $\mathcal{S}_1$ 协同 $\mathcal{S}_2$ 计算 $(z_1, z_2) \leftarrow \text{STMA}(\zeta_1^\sigma, \zeta_2^\sigma)$ 。

若 $\sigma = -1$, 幂函数 x^σ 表现为倒数函数 $\frac{1}{x}$ 。

深度神经网络中, 倒数运算是分类和回归训练目标函数的重要组成部分, 这里设计一种安全倒数计算 (SRec, secure reciprocal calculation) 协议, 输入 x 随机拆分为 x_1 和 x_2 , 分别交由 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 持有, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 协作计算 SRec 协议后输出 z_1 和 z_2 , 满足 $z_1 + z_2 = \frac{1}{x_1 + x_2}$ 。SRec 协议具体执行如下:

1) $\mathcal{S}_1$ 协同 $\mathcal{S}_2$ 计算 $(\eta_1, \eta_2) \leftarrow \text{STAM}(x_1, x_2)$;

2) $\mathcal{S}_1$ 协同 $\mathcal{S}_2$ 计算 $(z_1, z_2) \leftarrow \text{STMA}\left(\frac{1}{\eta_1}, \frac{2}{\eta_2}\right)$ 。

若 $\sigma = 2$, 幂函数 x^σ 退化为二次方函数 x^2 , 不失一般性, 将乘法表示为 $z = x \cdot y$, $x, y \in \mathbb{R}$, 一种安全高效的乘法计算 (SMul, secure multiplication calculation) 协议对于安全深度学习具有很强的积极意义。随机拆分 $x = x_1 + x_2$ 和 $y = y_1 + y_2$, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 分别持有 (x_1, y_1) 和 (x_2, y_2) , 分别输出的 z_1 和 z_2 满足 $z_1 + z_2 = (x_1 + x_2) \cdot (y_1 + y_2)$ 。SMul 协议具体执行如下:

1) $\mathcal{S}_1$ 协同 $\mathcal{S}_2$ 计算 $(\psi'_1, \psi'_2) \leftarrow \text{STMA}(x_1, y_2)$ 和 $(\psi''_1, \psi''_2) \leftarrow \text{STMA}(y_1, x_2)$;

2) $\mathcal{S}_1$ 计算 $z_1 \leftarrow x_1 \cdot y_1 + \psi'_1 + \psi''_1$, $\mathcal{S}_2$ 计算 $z_2 \leftarrow x_2 \cdot y_2 + \psi'_2 + \psi''_2$ 。

4.2 安全比较计算协议

在深度神经网络中, 比较运算十分常见, 如 ReLU 激活函数需要比较输入与 0 的大小, 目标检测中的边界框排序也必须使用比较操作。意图比较 x 和 y ($x, y \in \mathbb{R}$) 的大小, 首先随机拆分

$x = x_1 + x_2$ 和 $y = y_1 + y_2$, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 分别持有 (x_1, y_1) 和 (x_2, y_2) 。通俗来说, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 可以通过传递双方独立计算的差值 $x_1 - y_1$ 和 $x_2 - y_2$, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 计算 $(x_1 - y_1) + (x_2 - y_2)$ 可以获得 $x - y$, 这个差值的正负代表了 x 和 y 的比较结果。遗憾的是, 这种比较方法不能推广至单个输入与 0 的比较。例如, 当上述输入 $y = y_1 = y_2 = 0$ 时, $x_1 - y_1$ 或 $x_2 - y_2$ 相当于双方直接将己方输入进行传递, 这显然是不安全的。相比差值传递这种朴素方法, 本文设计一种安全比较计算 (SComp, secure comparison calculation) 协议, 可以安全实现两个输入的比较以及任意输入与 0 的比较。 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 具体执行如下:

- 1) $\mathcal{S}_1$ 计算 $\theta_1 \leftarrow x_1 - y_1$, $\mathcal{S}_2$ 计算 $\theta_2 \leftarrow x_2 - y_2$;
- 2) $\mathcal{S}_1$ 协同 $\mathcal{S}_2$ 计算 $(\phi_1, \phi_2) \leftarrow \text{STAM}(\theta_1, \theta_2)$;
- 3) $\mathcal{S}_1$ 随机选择 $\rho_1 \in \mathbb{Z}_m$ 且 $\rho_1 > 0$, 计算 $\phi_1 \leftarrow \rho_1 \cdot \phi_1$, 并将 ϕ_1 发送给 $\mathcal{S}_2$;
- 4) $\mathcal{S}_2$ 随机选择 $\rho_2 \in \mathbb{Z}_m$ 且 $\rho_2 > 0$, 计算 $\phi_2 \leftarrow \rho_2 \cdot \phi_2$, 并将 ϕ_2 发送给 $\mathcal{S}_1$;
- 5) $\mathcal{S}_1$ 计算 $\chi \leftarrow \text{sign}(\phi_1 \cdot \phi_2)$, $\mathcal{S}_2$ 计算 $\chi \leftarrow \text{sign}(\phi_2 \cdot \phi_1)$ 。

协议最后, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 计算得到 $x - y$ 的符号 χ , 若 $\chi = 0$, 意味着 $x - y \geq 0$, 即 $x \geq y$; 否则 $\chi = 1$, $x < y$ 。SComp 协议的正确性和安全性将在第 5 节详细介绍。

若比较 x 和 0 的大小, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 调用 STAM 协议执行 $(\phi_1, \phi_2) \leftarrow \text{STAM}(x_1, x_2)$ 获得输入的乘积变换结果 $\phi_1 \cdot \phi_2 = x_1 + x_2$, 然后顺序执行 SComp 协议的步骤 3) 至步骤 5), $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 计算得到 x 的符号 χ , 若 $\chi = 0$, 意味着 $x \geq 0$; 否则 $\chi = 1$, $x < 0$ 。

4.3 安全指数与对数计算协议

指数函数是 Sigmoid 激活函数、Softmax 归一化等常用操作的核心计算, 已知输入 $x \in \mathbb{R}$, 计算 $z = \tau^x$, 其中底数 $\tau \in \mathbb{R}$ 。特别地, 若 $\tau = e$, 则表现为自然指数函数。 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 拥有 x_1 和 x_2 , 满足 $x = x_1 + x_2$, 执行安全指数计算 (SExp, secure exponent calculation) 协议后输出 z_1 和 z_2 , 满足 $z_1 + z_2 = \tau^{x_1 + x_2}$, 协议描述如下。

- 1) $\mathcal{S}_1$ 计算 $g_1 \leftarrow \tau^{x_1}$, $\mathcal{S}_2$ 计算 $g_2 \leftarrow \tau^{x_2}$;
- 2) $\mathcal{S}_1$ 协同 $\mathcal{S}_2$ 计算 $(z_1, z_2) \leftarrow \text{STMA}(g_1, g_2)$ 。

对数函数是网络训练目标函数的主要组成部分, 如 L1 损失函数和交叉熵函数, 已知输入

$x \in \mathbb{R}$, 计算 $z = \log_\tau(x)$, 其中底数 $\tau \in \mathbb{R}$ 。特别地, 若 $\tau = e$, 则表现为自然对数函数。 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 拥有 x_1 和 x_2 , 满足 $x = x_1 + x_2$ 。执行安全对数计算 (Slog, secure logarithm calculation) 协议后输出 z_1 和 z_2 , 满足 $z_1 + z_2 = \log_\tau(x_1 + x_2)$, 协议描述如下:

- 1) $\mathcal{S}_1$ 协同 $\mathcal{S}_2$ 计算 $(\zeta_1, \zeta_2) \leftarrow \text{STAM}(x_1, x_2)$;
- 2) $\mathcal{S}_1$ 计算 $z_1 \leftarrow \log_\tau(\zeta_1)$, $\mathcal{S}_2$ 计算 $z_2 \leftarrow \log_\tau(\zeta_2)$ 。

4.4 安全除法计算协议

除法计算是基本运算方式之一, 针对 4 种类型的除法形式 $\frac{f(x_1, x_2)}{g(y_1, y_2)}$, $f(+)$ 和 $g(\cdot)$ 分别表示加法和乘法运算, 考虑将加法形式的分母或分子转换为乘法形式, 然后执行除法分割操作。下面详细讨论安全除法计算协议的设计方法。

对于第一种除法形式 $z_1 + z_2 = \frac{x_1 + x_2}{y_1 + y_2}$,

$\mathcal{S}_i (i=1, 2)$ 提供输入 $x_i, y_i \in \mathbb{R}$, 协作执行 (SDiv, secure division calculation) 协议后输出 z_i 。具体协议步骤如下:

- 1) $\mathcal{S}_1$ 协同 $\mathcal{S}_2$ 计算 $(\mu_1, \mu_2) \leftarrow \text{STAM}(x_1, x_2)$ 和 $(\eta_1, \eta_2) \leftarrow \text{STAM}(y_1, y_2)$;

- 2) $\mathcal{S}_1$ 协同 $\mathcal{S}_2$ 计算 $(z_1, z_2) \leftarrow \text{STMA}\left(\frac{\mu_1}{\eta_1}, \frac{\mu_2}{\eta_2}\right)$ 。

对于第二种除法形式 $z_1 + z_2 = \frac{x_1 \cdot x_2}{y_1 + y_2}$,

$\mathcal{S}_i (i=1, 2)$ 首先需要利用 STAM 协议将加法形式的分母转换为乘积形式, 再利用 STMA 协议实现输入的加法拆分, 即获得输出 z_1 和 z_2 ; 对于第三种除法形式 $z_1 + z_2 = \frac{x_1 + x_2}{y_1 \cdot y_2}$, $\mathcal{S}_i (i=1, 2)$ 只需要利

用 STAM 协议将加法形式的分子转换为乘积形式, 剩余操作类似于 SDiv 协议的步骤 2); 对于第 4 种除法形式 $z_1 + z_2 = \frac{x_1 \cdot x_2}{y_1 \cdot y_2}$, $\mathcal{S}_i (i=1, 2)$ 直接

可以利用 STMA 协议实现输入的加法拆分运算。

5 理论分析

5.1 正确性分析

第 3 和第 4 节的诸多协议是针对深度神经网络模型需求而设计的, 正确性是计算协议可行性的一部分, 接下来, 本节提供详细的推导过程证

明协议的正确性。

STMA 协议根据多项式进行拆分, STAM 协议传递除法结果关联双方信息, 完整输出分别为 $f_1 + f_2 = (a_1 + p_1) \cdot (a_2 + p_2) = u_1 \cdot u_2$ 和 $f_1 \cdot f_2 = \left[\frac{[(u_1 + u_2)]}{f_2} + a_1 \cdot a_2 - (c_1 + c_2) \right] \cdot f_2 = u_1 + u_2$, 显然 STMA 和 STAM 协议是正确的。在此基础上进行线性组合扩展, SPow、SRec 和 SMul 协议的完整输出分别为 $z_1 + z_2 = (\zeta_1 \cdot \zeta_2)^\sigma = (x_1 + x_2)^\sigma$ 、 $z_1 + z_2 = \frac{1}{\eta_1 \cdot \eta_2} = \frac{1}{x_1 + x_2}$ 和 $z_1 + z_2 = x_1 \cdot y_1 + x_2 \cdot y_2 + (\psi'_1 + \psi'_2) + (\psi''_1 + \psi''_2) = (x_1 + x_2) \cdot (y_1 + y_2)$, 这些推导过程可以证明 SPow、SRec 和 SMul 协议是正确的。在 SComp 协议中, 通过调用 STAM 协议得到 $\phi_1 \cdot \phi_2 = x - y$, $\mathcal{S}_1$ 计算 $\text{sign}(\phi_1 \cdot \phi_2) = \text{sign}(\phi_1 \cdot \phi_2 \cdot \rho_2)$, 由于 $\rho_2 > 0$, $\phi_1 \cdot \phi_2$ 与 $x - y$ 的正负性相同; 同理, $\mathcal{S}_2$ 能判断 $x - y$ 的正负性, 若 $\chi = 0$, 意味着 $x \geq y$; 否则 ($\chi = 1$), $x < y$ 。SExp 和 SLog 协议的输出分别为 $z_1 + z_2 = \mathcal{S}_1 \cdot \mathcal{S}_2 = \tau^{x_1 + x_2}$ 和 $z_1 + z_2 = \log_r(\zeta_1 \cdot \zeta_2) = \log_r(x_1 + x_2)$, 可见 SExp 和 SLog 协议是正确的。针对安全除法计算协议, 利用 STAM 协议将加法形式的分子或分母转换为乘积形式, 再利用 STMA 协议可以正确实现输入的加法分割。就 SDiv 协议而言, 其输出结果为 $z_1 + z_2 = \frac{\mu_1 \cdot \mu_2}{\eta_1 \cdot \eta_2} = \frac{x_1 + x_2}{y_1 + y_2}$, 可见计算过程是正确的。

5.2 复杂度分析

本节从计算和通信两方面讨论协议的复杂度。假设协议输入数组长度为 n , STMA 和 STAM 协议顺序执行关于输入数组和随机数组的线性计算, 计算复杂度均为 $O(n)$ 。表 1 将第 4 节提出的

计算协议与文献[11,12,16]进行了比较, 文献[11]提出的 SMul 协议是线性的, 其计算复杂度为 $O(n)$, SComp 协议将加法输入形式转换为异或形式, 需要引入进位循环结构, 循环次数依赖于输入的二进制位长度 l , 其计算复杂度为 $O(nl)$, 而本文只需要执行一次 STAM 协议和一些线性计算, 计算复杂度可以维持在 $O(n)$ 。文献[12]提出的 SRec、SExp 和 SLog 协议将函数泰勒展开为多项式, 迭代调用 m 次 SMul 协议, 而本文只需要顺序调用若干次 STMA 和 STAM 协议, 计算复杂度仍为 $O(n)$ 。文献[12]提出的 SPow 协议需要迭代调用 m 次 SRec 和 SMul 协议, 其计算复杂度高达 $O(nm^2)$, 而本文只需要执行一次 STAM 协议和一次 STMA 协议, 计算复杂度仅为 $O(n)$ 。关于 SDiv 除法协议, 文献[16]将乘积形式的分子或分母转化为加法形式, 而本文将加法形式的分子或分母转化为乘积形式, 虽然计算复杂度均为 $O(n)$, 但在实际测试中, 本文的 SDiv 协议计算开销较低。

接下来, 讨论协议中 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 之间的通信轮数, STMA 和 STAM 协议分别需要一轮和二轮通信传递包含输入副本的随机数组, 表 2 比较了本文协议与相关工作的通信轮数。具体而言, 文献[11]提出的 SComp 协议需要迭代执行 SMul 乘法协议, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 需要完成 $2l + 4$ 轮通信, 而本文只需要 3 轮通信完成数据传递。本文提出的 SPow、SRec、SExp 和 SLog 协议至多执行两次 STMA 和 STAM 协议, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 至多需要 3 轮通信, 而文献[12]需要迭代执行 m 次 SMul 协议, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 至少需要 m 轮通信, 为了满足精度需求, $m \gg 3$ 。关于第一种除法类型, 文献[12]与本文提出的 SDiv 协议均执行 5 轮通信, 针对第 2、3 和 4 种除法类型, 文献[12]需要额外执行 STMA 协议将

表 1 协议的计算复杂度比较
Table 1 Comparison of computational complexity of the proposed protocols

相关工作	SPow	SRec	SMul	SComp	SExp	SLog	SDiv
文献[11]	N.A.	N.A.	$O(n)$	$O(nl)$	N.A.	N.A.	N.A.
文献[12]	$O(nm^2)$	$O(nm)$	N.A.	N.A.	$O(nm)$	$O(nm)$	N.A.
文献[16]	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	$O(n)$
本文	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$

注: n 表示输入数组长度; l 表示二进制位长度; m 表示子协议迭代次数

表2 协议的通信复杂度比较
Table 2 Comparison of communication complexity of the proposed protocols

相关工作	SPow	SRec	SMul	SComp	SExp	SLog	SDiv
文献[11]	N.A.	N.A.	1	$2l+4$	N.A.	N.A.	N.A.
文献[12]	$2m^2+m$	$2m$	N.A.	N.A.	$m+1$	$3m+2$	N.A.
文献[16]	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	5
本文	3	3	2	3	1	2	5

注: l 表示二进制位长度; m 表示子协议迭代次数

乘积形式的分子或分母转化为加法形式,相对地, SDiv 协议可以节省 STAM 协议的通信轮数。

5.3 安全性分析

为了形式化证明提出协议的安全性, 首先定义半可信模型下的协议安全^[17-18], 如果存在一个概率多项式时间的模拟器 $\mathcal{M}$, 它可以为现实敌手 $\mathcal{A}$ 生成一组模拟视图, 实质上是一些均匀分布的随机数, 若 $\mathcal{A}$ 在计算上不能区分真实视图和模拟视图, 那么认为模型下的协议是安全的。此外, 针对协议的组合安全和协议内部的随机数安全, 存在如下引理^[17-18]。

引理 1 若协议调用的所有子协议在多项式时间内是可模拟的, 那么该协议是可模拟的。

引理 2 若 $a \in \mathbb{Z}_m$ 是均匀分布的, 并且与 $b \in \mathbb{Z}_m$ 相互独立, 那么认为 $a \pm b$ 也是均匀分布的, 并且与 b 相互独立。

定理 1 在半可信模型中, STMA 和 STAM 协议是安全的。

证明 在 STMA 协议中, $\mathcal{S}_1$ 的真实视图为 $\{u_1, a_1, c_1, p_1, p_2, f_1\}$, 其中 a_1 是从 $\mathbb{Z}_m$ 中随机选择的, 根据引理 2 可知 $p_1 = u_1 - a_1$ 是均匀分布的, 接收到 p_1 后, $\mathcal{S}_2$ 不能推测出 $\mathcal{S}_1$ 的隐私信息 u_1 。对于概率多项式时间敌手 $\mathcal{A}$ 而言, 模拟器 $\mathcal{M}$ 为 $\mathcal{S}_1$ 生成对应的模拟视图, $\mathcal{A}$ 在计算意义上无法进行区分。同理, $\mathcal{S}_2$ 的真实视图为 $\{u_2, a_2, c_2, p_1, p_2, f_2\}$, 根据引理 2 可知 p_2 是均匀分布的随机数, $\mathcal{S}_1$ 不能推测出 $\mathcal{S}_2$ 的隐私信息 u_2 , 并且, $\mathcal{A}$ 在计算上无法区分真实视图和 $\mathcal{M}$ 为 $\mathcal{S}_2$ 生成的模拟视图。在 STAM 协议中, $\mathcal{S}_1$ 的真实视图为 $\{u_1, a_1, c_1, p_1, \lambda', \lambda'', f_1\}$, 根据引理 2 可知 p_1 是均匀分布的, $\mathcal{S}_2$ 根据已有信息不能推测出 u_1 ; $\mathcal{S}_2$ 的真实视图为 $\{u_2, a_2, c_2, p_1, \lambda', \lambda'', f_2\}$, λ' 和 λ'' 是均匀分布的, $\mathcal{S}_1$ 不能推测出 u_2 。此外, $\mathcal{M}$ 可以为 $\mathcal{S}_1$ 和

$\mathcal{S}_2$ 生成均匀的模拟视图, $\mathcal{A}$ 在计算上不能区分真实视图和模拟视图。

证毕。

定理 2 在半可信模型中, SPow、SRec 和 SMul 协议是安全的。

证明 在 SPow 协议中, $\mathcal{S}_1$ 的真实视图为 $\{x_1, \varsigma_1, z_1\}$, ς_1 和 z_1 分别为 STAM 和 STMA 协议的输出, 由于 STAM 和 STMA 协议在定理 1 中证明是安全的, 根据引理 1, 模拟器 $\mathcal{M}$ 可以为 $\mathcal{S}_1$ 生成对应的模拟视图, 并且对于概率多项式敌手 $\mathcal{A}$ 而言, 在计算上无法区分真实视图和模拟视图。同理, $\mathcal{S}_2$ 的真实视图为 $\{x_2, \varsigma_2, z_2\}$, $\mathcal{A}$ 在计算上无法区分真实视图和 $\mathcal{M}$ 生成的模拟视图。在 SRec 协议中, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 的真实视图分别为 $\{y_1, \eta_1, z_1\}$ 和 $\{y_2, \eta_2, z_2\}$, η_1 和 η_2 来自于 STAM 协议的输出, 根据引理 2 可知 η_1 和 η_2 是均匀分布的, 并且由于 STAM 和 STMA 协议的安全性, $\mathcal{A}$ 在计算上无法区分 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 的真实视图与模拟视图。在 SMul 协议中, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 的真实视图分别为 $\{x_1, y_1, \psi'_1, \psi''_1, z_1\}$ 和 $\{x_2, y_2, \psi'_2, \psi''_2, z_2\}$, $\{\psi'_1, \psi'_2, \psi''_1, \psi''_2\}$ 来自 STMA 协议的输出, 根据引理 1, 模拟器 $\mathcal{M}$ 可以为 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 生成模拟视图, $\mathcal{A}$ 在计算上无法区分真实视图和模拟视图。

证毕。

定理 3 在半可信模型中, SComp、SExp、SLog 和 SDiv 协议是安全的。

证明 在 SComp 协议中, $\mathcal{S}_1$ 的真实视图为 $\{x_1, y_1, \theta_1, \phi_1, \rho_1, \varphi_1, \varphi_2, \chi\}$, ϕ_1 为 STAM 协议的输出, 由于 ρ_1 是随机选择的, 根据引理 2 可知 φ_1 是均匀的随机数, $\mathcal{S}_2$ 从 φ_1 中不能推测出 x_1 和 y_1 , 甚至不能获得 ϕ_1 , 根据引理 1, 模拟器 $\mathcal{M}$ 可以为 $\mathcal{S}_1$ 生成对应的模拟视图, $\mathcal{A}$ 在计算上无法区分真实视图和模拟视图。同理, $\mathcal{S}_2$ 的真实视图为 $\{x_2, y_2, \theta_2, \phi_2,$

$\rho_2, \phi_1, \phi_2, \chi\}$, $\mathcal{A}$ 在计算上无法区分真实视图和 $\mathcal{M}$ 为 $\mathcal{S}_2$ 生成的模拟视图。在 SExp 协议中, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 的真实视图分别为 $\{x_1, g_1, z_1\}$ 和 $\{x_2, g_2, z_2\}$, z_1 和 z_2 为 STMA 协议的输出, 由于 STMA 协议在定理 1 中已经证明是安全的, 根据引理 1, 模拟器 $\mathcal{M}$ 可以为 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 生成模拟视图, $\mathcal{A}$ 在计算上无法区分真实视图和模拟视图。在 SLog 协议中, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 的真实视图分别为 $\{x_1, \zeta_1, z_1\}$ 和 $\{x_2, \zeta_2, z_2\}$, ζ_1 和 ζ_2 为 STAM 协议的输出, 由于 STAM 协议在定理 1 中证明是安全的, 根据引理 1, 模拟器 $\mathcal{M}$ 可以为 $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 生成模拟视图, $\mathcal{A}$ 在计算上无法区分真实视图和模拟视图。在 SDiv 协议中, $\mathcal{S}_1$ 的真实视图为 $\{x_1, y_1, \mu_1, \eta_1, z_1\}$, μ_1 和 η_1 是 STAM 协议的输出, 根据引理 2 可知 μ_1 和 η_1 是均匀分布的随机数, 然后调用 STMA 协议获得输出, 由于 STAM 和 STMA 协议在定理 1 中证明是安全的, 根据引理 1, 模拟器 $\mathcal{M}$ 可以为 $\mathcal{S}_1$ 生成模拟视图, 并且 $\mathcal{A}$ 在计算上无法区分真实视图和模拟视图。同理, $\mathcal{A}$ 在计算上无法区分 $\mathcal{S}_2$ 的真实视图 $\{x_2, y_2, \mu_2, \eta_2, z_2\}$ 和 $\mathcal{M}$ 均匀生成的模拟视图。关于其他除法输入形式的计算安全性, 均可以由 STAM 和 STMA 协议及其线性组合的安全性提供保证。

证毕。

6 协议性能分析

本节使用 Intel Core i7-8565U CPU 1.80 GHz, 20 GB RAM 硬件配置的 64 位计算机, 在 Pycharm 仿真平台上进行实验, 协议涉及的数组操作利用 Numpy 工具完成。实际数据从 $\mathbb{R}$ 中随机选取, 为了符合协议输入要求, 将这些数据保留 P 位小数, 并乘上 10^P 转换为整数形式。接下来, 首先讨论提出协议的计算误差, 然后从计算和通信开销方面评估所提安全计算协议的性能。

6.1 误差分析

计算误差是引起安全计算协议输出结果不精确的重要原因, 本文主要讨论误差与协议输入范围的关系, 设置 $P=8$, 即保留 8 位小数参与计算, 并定义两个误差评价指标: 绝对误差 (AE, absolute error) 和相对误差 (RE, relative error)。已知实际输出数组与理论输出数组的绝对差值, AE 表示数组差值中的最大值, 象征实际输出与

理论输出的最大偏离程度; 已知数组差值与理论输出数组的比值, RE 表示比值中的最大值, 象征偏离程度占理论输出的比例。

将 $n < 10^5$ 长度的数组作为输入, 使实际探测得到的 AE 和 RE 足够客观, 由于个别协议的增长幅度和增长方向不一致, 将根据实际的 AE 和 RE 提供文字说明, 大部分协议的计算误差如表 3 所示。STMA 协议的 AE 随着输入范围增大而增大, 输入在 10^4 范围内取值时, AE 仍能控制在 10^{-4} 量级, RE 只有 10^{-8} 量级, 意味着误差对于本身输出而言可以忽略不计。STAM 协议的 AE 随着输入范围增大维持在 10^{-5} 不变, 具备优良的计算性能。SPow 协议执行一次 STAM 协议和一次 STMA 协议, AE 可以控制在 10^{-5} , RE 可以控制在 10^{-4} ; SMul 执行两次 STMA 协议, AE 可以控制在 10^{-4} , RE 可以控制在 10^{-7} , 这种细微的误差在可接收范围内。SComp 协议调用一次 STAM 协议, 由于 STAM 协议不会导致整数部分出现误差, 进而不会影响比较结果的正负性, SComp 协议可以实现比较计算零误差。SLog 协议的 AE 随着输入范围增大而减小, 控制在 10^{-3} 内。由于 SExp 协议随着输入呈指数增长, 实际探测的 AE 随着输入范围增大而增大, 当输入在 10^4 h 以内时, AE 可以控制在 10^{-4} 量级, RE 可以维持在 10^{-8} 量级, 意味着误差不会对计算结果形成影响。SRec 协议接收 10^{-3} 范围内的输入, AE 可以维持在 10^{-3} 左右, SDiv 协议调用 STAM 和 SRec 协议, AE 的变化情况类似于 SRec 协议。

表 3 协议的绝对误差和相对误差
Table 3 The absolute error and relative error of the proposed protocol

输入范围		10^1	10^2	10^3	10^4
STMA	AE	10^{-7}	10^{-6}	10^{-5}	10^{-4}
	RE	10^{-6}	10^{-7}	10^{-7}	10^{-8}
STAM	AE	10^{-5}	10^{-5}	10^{-5}	10^{-5}
	RE	10^{-4}	10^{-5}	10^{-7}	10^{-8}
SPow	AE	10^{-5}	10^{-6}	10^{-6}	10^{-7}
	RE	10^{-4}	10^{-6}	10^{-7}	10^{-8}
SMul	AE	10^{-7}	10^{-6}	10^{-5}	10^{-4}
	RE	10^{-7}	10^{-8}	10^{-10}	10^{-9}
SLog	AE	10^{-3}	10^{-4}	10^{-4}	10^{-5}
	RE	10^{-3}	10^{-4}	10^{-5}	10^{-6}

6.2 实际开销分析

6.2.1 计算开销

所提安全计算协议的计算开销如表4所示,当输入数组长度 $n < 10^4$ 时,协议的计算开销可以忽略不计。为了更好地显示协议运行时间与数组长度的关系,测试 n 在 $10^4 \sim 10^7$ 时所提协议的运行时间。宏观来看,本文所有协议的运行时间与 n 呈线性增长关系,这与表1所示的计算复杂度一致。具体地,即使STMA和STAM协议接收 10^7 长度的输入数组,仍能控制其计算开销在91.16 ms以内。SExp协议执行一次STMA协议, SLog协议执行一次STAM协议,可见产生的计算开销大致相等;当输入数组长度 $n = 10^5$ 时,计算开销均为0.80 ms。SPow、SRec和SMul协议均执行一次STAM协议和一次STMA协议,产生的计算开销约为SExp和SLog协议的两倍。其中,当 n 一定时,SMul协议产生的计算开销略高于文献[11],但处理 10^5 长度的数组也仅为1.70 ms。关于安全比较计算协议,文献[11]的复杂度依赖于输入的二进制位长度 l ,处理 10^6 长度数组需要481.71 ms,而本文所提SComp协议的计算开销仅为前者的 $\frac{1}{30}$ 。关于安全除法计算协议,相比文献[16],本文提出的SDiv协议大概可以节省 $\frac{1}{2}$ 计算开销;针对其他3种除法类型,文献[16]需要额外附加STMA协议

的计算开销,而SDiv协议却可以节省STAM协议的计算开销,可知SDiv协议的执行效率高于文献[16]。

6.2.2 通信开销

输入数组和随机数组元素均由定点数转换得到,在内存中占用8个字节,安全计算协议实际产生的通信开销与通信轮数以及传递消息大小有关。假设协议接收长度为 $n = 10^5$ 的输入数组,这足够处理大部分深度学习网络的数组长度,本文提出协议的通信开销如表5所示,对于STMA和STAM协议, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 的通信开销维持在1.53 kB内, $\mathcal{T}$ 产生的通信开销略大于 $\mathcal{S}_1$ 和 $\mathcal{S}_2$,但是 $\mathcal{T}$ 可以离线完成计算和通信任务,进一步压缩了通信开销。对于SPow、SRec、SMul、SComp、SExp和SLog协议, $\mathcal{S}_1$ 和 $\mathcal{S}_2$ 的通信开销基本对称,维持在2.29 kB内。其中,SMul协议产生的通信开销略高于文献[11],但SComp协议产生的通信开销远远低于文献[11],这是因为SComp协议的通信开销与 l 无关,仅依赖于输入数组长度。关于安全除法计算,SDiv协议产生的通信开销近似于文献[16],参考表2的通信复杂度,其他除法类型产生的通信开销明显低于文献[16]。

7 结束语

智能终端由于自身资源受限,选择将复杂的深度神经网络模型计算任务卸载至边缘节点。考

表4 协议的计算开销
Table 4 The computational overhead of the proposed protocol

输入数量 n	运行时间/ms											
	STMA	STAM	SPow	SRec	SMul ^[11]	SMul	SComp ^[11]	SComp	SExp	SLog	SDiv ^[16]	SDiv
10^4	≈ 0	0.10	0.10	0.10	0.20	0.10	7.88	0.20	0.10	0.10	0.30	0.10
10^5	0.60	0.70	1.70	1.40	0.99	1.70	39.20	1.20	0.80	0.80	3.39	2.09
10^6	8.28	8.08	19.64	16.25	11.87	20.85	481.71	13.97	10.18	8.87	43.08	25.73
10^7	87.17	91.16	206.95	166.85	110.61	214.63	4770.44	147.51	106.02	92.15	417.18	252.13

表5 协议的通信开销
Table 5 The communication overhead of the proposed protocol

参与方	通信开销/kB											
	STMA	STAM	SPow	SRec	SMul ^[11]	SMul	SComp ^[11]	SComp	SExp	SLog	SDiv ^[16]	SDiv
$\mathcal{S}_1$	0.76	0.76	1.53	1.53	1.53	1.53	24.41	1.53	0.76	0.76	3.81	2.29
$\mathcal{S}_2$	0.76	1.53	2.29	2.29	1.53	1.53	24.41	2.29	0.76	1.53	3.81	3.81
$\mathcal{T}$	1.53	1.53	3.05	3.05	2.29	3.05	1.53	1.53	1.53	1.53	6.10	4.58

虑到计算过程的安全性, 本文为网络底层函数量身设计了一系列安全计算协议, 可以扩展至数组计算, 不需要迭代操作即可实现函数的线性分割, 两台非共谋边缘服务器及潜在敌手均不能获得完整的输入和输出。经过实际性能评估, 协议的计算误差可以忽略不计, 并且计算和通信开销均优于现有工作, 十分适合深度神经网络函数的安全替换。在未来工作中, 将继续寻找进一步提升协议计算和通信效率的方法, 同时设计更多通用函数对应的安全计算协议。

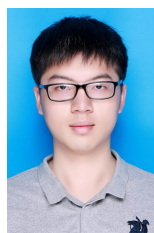
参考文献:

- [1] ZHOU Y, TUZEL O. Voxelnet: end-to-end learning for point cloud based 3d object detection[C]//The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2018: 4490-4499.
- [2] REN S Q, HE K, GIRSHICK R, et al. Faster R-CNN: towards real-time object detection with region proposal networks[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2015, 39(6): 91-99.
- [3] YANN L, BERNHARD B, JOHN S D, et al. Backpropagation applied to handwritten zip code recognition[J]. Neural Computation, 1989, 1(4): 541-551.
- [4] XIONG J B, BI R W, SZHAO M F, et al. Edge-assisted privacy-preserving raw data sharing framework for connected autonomous vehicles[J]. IEEE Wireless Communications, 2020, 27(3): 24-30.
- [5] HESAMIFARD E, TAKABI H, GHASEMI M. CryptoDL: deep neural networks over encrypted data[J]. arXiv preprint arXiv:1711.05189, 2017.
- [6] MOHASSEL P, ZHANG Y P. SecureML: a system for scalable privacy-preserving machine learning[C]//IEEE Symposium on Security and Privacy (S&P). 2017: 19-38.
- [7] YAO A C. Protocols for secure computations[C]//The 23rd Annual Symposium on Foundations of Computer Science (SFCS). 1982: 160-164.
- [8] 唐春明, 魏伟明. 基于安全两方计算的具有隐私性的回归算法[J]. 信息安全学报, 2018(10): 10-16.
TANG C M, WEI W M. Regression algorithm with based on secure two-party computation[J]. Information Network Security, 2018(10): 10-16.
- [9] RABIN M O. How to exchange secrets with oblivious transfer[J]. IACR Cryptology ePrint Archive, 2005:187.
- [10] 刘新, 李顺东, 陈振华. 基本初等函数的保密云计算服务协议[J]. 计算机科学, 2015, 42(10):159-163.
LIU X, LI S D, CHEN Z H. Secure cloud computing service protocols of elementary functions[J]. Computer Science, 2015, 42(10): 159-163.
- [11] HUANG K, LIU X M, FU S J, et al. A lightweight privacy-preserving CNN feature extraction framework for mobile sensing[J]. IEEE Transactions on Dependable and Secure Computing, 2019: 1.
- [12] MA Z, LIU Y, LIU X M, et al. Privacy-preserving outsourced speech recognition for smart IoT devices[J]. IEEE Internet of Things Journal, 2019, 6(5): 8406-8420.
- [13] LIU Y, MA Z, LIU X M, et al. Privacy-preserving object detection for medical images with Faster R-CNN[J]. IEEE Transactions on Information Forensics and Security, 2019: 1.
- [14] VOLDER J E. The cordic trigonometric computing technique[J]. IRE Transactions on Electronic Computers, 1959, 8(3): 330-334.
- [15] ZHU Y W, HUANG L S, YANG W, et al. Three new approaches to privacy-preserving add to multiply protocol and its application[C]//The 2nd International Workshop on Knowledge Discovery and Data Mining. 2009: 554-558.
- [16] 李禾, 王述洋. 关于除法的安全双方计算协议[J]. 计算机工程与应用, 2010, 46(6): 86-88.
LI H, WANG S Y. Several secure two-party protocols of division[J]. Computer Engineering and Applications, 2010, 46(6): 86-88.
- [17] BOGDANOV D, LAUR S, WILLEMSON J. Sharemind: a framework for fast privacy-preserving computations[C]//European Symposium on Research in Computer Security. 2008: 192-206.
- [18] BOGDANOV D, NIITSOO M, TOFT T, et al. High-performance secure multi-party computation for data mining applications[J]. International Journal of Information Security, 2012, 11(6): 403-418.

[作者简介]



毕仁万 (1996-), 男, 湖南常德人, 福建师范大学硕士生, 主要研究方向为安全深度学习、安全多方计算。



陈前昕 (1996-), 男, 福建泉州人, 福建师范大学硕士生, 主要研究方向为安全深度学习、隐私保护技术。



熊金波 (1981-), 男, 湖南益阳人, 博士, 福建师范大学教授, 主要研究方向为安全深度学习、移动群智感知、隐私保护技术。



刘西蒙 (1988-), 男, 陕西西安人, 博士, 福州大学教授, 主要研究方向为云安全、应用密码学、大数据安全。