软件学报 ISSN 1000-9825, CODEN RUXUEW
[doi: 10.13328/j.cnki.jos.007475] [CSTR: 32375.14.jos.007475]
©中国科学院软件研究所版权所有.
E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于秘密分享的高效隐私保护卷积神经网络预测*

白 浩[1,2], 何 琨[1,2], 陈 晶[1,2], 赵陈斌[1,2], 杜瑞颖[1,2]

[1](空天信息安全与可信计算教育部重点实验室 (武汉大学), 湖北 武汉 430040)

[2](武汉大学 国家网络安全学院, 湖北 武汉 430040)

通信作者: 何琨, E-mail: hekun@whu.edu.cn

摘　要: 针对隐私保护卷积神经网络预测, 先前的研究采用同态加密、安全多方计算等方法来保护客户端隐私敏感数据. 然而, 这些方法通常面临预测时间开销过大的问题. 为了解决此问题, 提出一个高效的隐私保护卷积神经网络预测方案. 该方案根据卷积神经网络中线性层和非线性层不同计算特点, 设计矩阵分解计算协议和参数化二次多项式近似 ReLU 激活函数方法, 从而实现了线性层和非线性层高效安全计算, 并缓解了近似处理而导致的预测准确率损失. 在线性层和非线性层中的计算都可以通过轻量级密码原语秘密分享来完成. 理论分析和实验结果表明, 在保证安全性前提下, 所提方案将预测速度提高了 2–15 倍, 同时预测准确率损失仅约为 2%.

关键词: 隐私保护预测; 卷积神经网络; 秘密分享; 非线性近似; 深度学习

中图法分类号: TP309

中文引用格式: 白浩, 何琨, 陈晶, 赵陈斌, 杜瑞颖. 基于秘密分享的高效隐私保护卷积神经网络预测. 软件学报. http://www.jos.org.cn/1000-9825/7475.htm

英文引用格式: Bai H, He K, Chen J, Zhao CB, Du RY. Efficient Privacy-preserving Inference Based on Secret Sharing for Convolutional Neural Network. Ruan Jian Xue Bao/Journal of Software (in Chinese). http://www.jos.org.cn/1000-9825/7475.htm

Efficient Privacy-preserving Inference Based on Secret Sharing for Convolutional Neural Network

BAI Hao[1,2], HE Kun[1,2], CHEN Jing[1,2], ZHAO Chen-Bin[1,2], DU Rui-Ying[1,2]

[1](Key Laboratory of Aerospace Information Security and Trusted Computing (Wuhan University), Ministry of Education, Wuhan 430040, China)

[2](School of Cyber Science and Engineering, Wuhan University, Wuhan 430040, China)

Abstract: In privacy-preserving inference using convolutional neural network (CNN) models, previous research has employed methods such as homomorphic encryption and secure multi-party computation to protect client data privacy. However, these methods typically suffer from excessive prediction time overhead. To address this issue, an efficient privacy-preserving CNN prediction scheme is proposed. This scheme exploits the different computational characteristics of the linear and non-linear layers in CNNs and designs a matrix decomposition computation protocol and a parameterized quadratic polynomial approximation for the ReLU activation function. This enables efficient and secure computation of both the linear and non-linear layers, while mitigating the prediction accuracy loss caused by the approximations. The computations in both the linear and non-linear layers can be performed using lightweight cryptographic primitives, such as secret sharing. Theoretical analysis and experimental results show that, while ensuring security, the proposed scheme improves prediction speed by a factor of 2 to 15, with only about a 2% loss in prediction accuracy.

Key words: privacy-preserving inference; convolutional neural network (CNN); secret sharing; non-linear approximation; deep learning

---

* 基金项目: 国家重点研发计划 (2022YFB3102100); 国家自然科学基金 (62172303, 62302343, 62076187); 湖北省重点研发计划 (2021BAA190, 2022BAA039); 山东省重点研发计划 (2022CXPT055)
收稿时间: 2024-11-11; 修改时间: 2025-03-17; 采用时间: 2025-05-26; jos 在线出版时间: 2025-09-28

基于云服务器的预测服务被广泛运用于许多领域, 例如模式识别[1,2]和医学诊断[3,4]. 在这种服务中, 客户端将自己的数据上传到云服务器, 云服务器使用训练好的深度神经网络模型进行预测. 但是存在的一个主要问题是客户端的输入和预测结果将暴露给云服务器[5,6]. 由于数据隐私的担忧和法律限制 (如《通用数据保护条例》[7]), 设计实用的隐私保护深度神经网络预测方案至关重要[8]. 在隐私保护预测中, 云服务器无法获取客户端的输入隐私数据和预测结果.

卷积神经网络 (convolutional neural network, CNN) 作为一种重要的深度神经网络模型, 已经广泛应用于图像识别、文本分类等任务中, 给人们的生活带来了许多便利. 然而, CNN 预测过程中会涉及复杂的线性层和非线性层计算. 现有基于 CNN 隐私保护预测的研究工作一般会针对 CNN 中不同层的特性设计了加密协议. 在线性层中, 现有研究工作采用了秘密分享[9,10]或同态加密[11−14]. 在非线性层中, 它们依赖于安全多方计算原语, 如混淆电路和不经意传输协议[15−17]. 然而, 这些研究工作仍然存在计算速度缓慢和通信开销巨大的问题[18]. 例如, Cheetah[14]利用 ResNet50 深度神经网络模型在 ImageNet 数据集上完成隐私保护预测, 客户端和云服务器需要花费 2.3 min, 并传输约 4.8 GB 的数据.

当前研究工作低效的原因有两个方面. 首先, 在线性层中, 为了完成矩阵乘法的安全计算, 通常会涉及复杂的加密原语 (例如, 同态加密), 导致计算效率较低. 其次, 非线性层中的修正线性单元激活函数 (rectified linear unit, ReLU) 需要使用计算效率低效的混淆电路进行计算. 例如, 约有 80% 左右的在线隐私预测时间开销会花费在混淆电路上[9]. 此外, 使用混淆电路通常需要双方进行多轮交互, 这会带来较高的通信开销, 进一步影响了整体隐私预测的效率.

针对卷积神经网络的特点, 隐私保护卷积神经网络预测除了满足保护客户端隐私敏感数据之外, 还需要具备以下特性. 第一, 较低计算和通信开销的密码学原语: 同态加密和安全多方计算等密码学原语能够处理密文数据, 但通常伴随高昂的计算和通信开销, 不适用于卷积神经网络预测. 第二, 保证隐私预测准确率: 为了高效完成非线性层的安全计算, 目前的研究方案通常对非线性函数进行近似处理, 并需要设计方法以减轻因近似处理导致的隐私保护预测准确率损失.

为此, 本文提出了高效的隐私保护卷积神经网络预测方案. 本文的主要贡献包括 3 个方面.

1) 设计了矩阵分解计算协议, 用于完成线性层矩阵乘法的安全计算, 这是线性层的核心. 客户端和云服务器都拥有矩阵乘法的秘密分享结果. 在该协议中仅使用计算高效的秘密分享密码学原语, 避免了同态加密等复杂密码学原语, 因此提高了线性层计算效率.

2) 设计了参数化的二次多项式来近似计算 ReLU 激活函数. 在非线性层中, 客户端和云服务器使用自己的份额来获取非线性层计算结果的秘密分享, 完全避免了混淆电路的使用. 为了确保隐私保护预测准确率, 构建了一个简单的神经网络来学习二次多项式近似的参数.

3) 线性层和非线性层都以秘密分享的方式完成, 确保了该方案的正确性和安全性. 并且, 该方案符合离线-在线范式, 保证了整体预测效率. 在 3 个数据集上进行了实验, 结果验证了该框架的有效性. 与现有工作相比, 隐私保护预测速度提高了 2–15 倍. 此外, 预测准确率损失仅约为 2%.

本文第 1 节介绍隐私预测的相关方法和研究现状. 第 2 节介绍本文所需的预备知识, 包括秘密分享和隐私保护预测, 以及本文的系统模型和威胁模型. 第 3 节介绍本文构建的方案, 包括安全线性层和安全非线性层. 第 4 节通过正确性分析、安全性分析和效率分析验证所提方案的正确性、安全性和高效性. 第 5 节通过实验测试验证本文所提出的方案的有效性. 最后总结全文.

1 隐私保护预测相关工作

隐私保护预测明确提出了允许两方在不泄露自己隐私的前提下, 通过协作的方式对隐私数据进行预测[19]. 当隐私数据持有方利用部署在服务器上的神经网络模型进行预测时, 可以借助加密技术包括同态加密和安全多方计算来完成隐私保护预测. 此类相关技术能保证输入隐私性与计算的正确性, 是用于隐私保护预测的一种潜在技术.

在基于同态加密的方案中, 例如 CryptoNets[11]、CrytoDL[20]和 Lola[21], 客户端将加密数据发送到云服务器, 云服务器执行同态多项式计算以得到加密结果. 然后, 客户端解密密文以输出预测结果. 然而, 这些方案很难准确计算 ReLU 激活函数. 一般采用近似多项式的方法计算非线性层, 但是这会导致预测准确率降低. 仅基于安全多方计算的方案较少见. XONN[22]提出了基于混淆电路的二值神经网络预测框架. 然而, 二进制权重对预测准确率产生负面影响, 并且该方案需要数千轮通信. 然而, 以上方案并未考虑到神经网络不同层的计算特点, 导致计算和通信开销过大, 难以得到实际应用.

为适应深度神经网络不同层的计算需求, 许多方案集成了多种加密技术, 包括同态加密和安全多方计算. Gazelle[12]采用了同态加密用于线性层安全计算, 混淆电路用于非线性层安全计算, 并且优化了同态计算方式. 这种范式已经被许多方案采纳, 如 FALCON[23]和 ENSEI[24]. Delphi[9]将隐私保护预测分为离线和在线两个阶段, 利用秘密分享构建了一个高效的线性层计算协议. 所设置的离线阶段集中进行 Gazelle 中繁重的同态加密计算. CrypTFlow[25]提出了一个用于非线性层计算的安全比较协议, 需要较少的带宽但更多的通信回合, 解决了现有工作在实现 ReLU 激活函数时通信开销较大的问题. Circa[26]将 ReLU 激活函数分解为分段线性函数和符号函数, 利用乘法三元组并使用近似符号来降低混淆电路的成本. COINN[27]设计了一种新的低位量化方法, 并通过使用定制的加密协议构建了新的协议. Cheetah[14]通过基于格的同态加密、不经意传输协议和秘密分享设计了一组新的加密协议, 用于线性层和非线性层计算. FastSecNet[19]通过函数秘密共享设计了一种新的非线性层的 ReLU 协议, 并提出了一种优化的线性层的乘法协议. Kim 等人[28]提出了一种更有效的使用全同态加密技术计算卷积的方法, 无论内核大小如何, 代价都保持不变, 从而进一步减少在各种内核大小的计算时间. 然而, 这些方案仍然具有较高的计算和通信复杂性. 本文所提出方案利用基于秘密分享的矩阵分解计算协议用于线性层矩阵乘法计算, 以及参数化二次多项式近似计算非线性层激活函数, 完全避免了混淆电路的使用. 整个隐私保护预测都是在秘密分享方式完成的. 由于秘密分享具有计算高效的特点, 因此提高了系统的隐私保护预测性能.

2 预备知识

简要介绍提出方案中使用的秘密分享密码学原语, 以及隐私保护预测中的重要概念. 最后, 提出了系统模型和威胁模型, 这是该方案的基础.

2.1 秘密分享

(T,N) 秘密分享是一种加密原语, 它将一个秘密值 $\langle X\rangle$ 分成 N 个秘密份额 $\langle X\rangle_1,\langle X\rangle_2,\ldots,\langle X\rangle_N$. 然后, 这些份额分发给参与者, 并且对任何 $T-1$ 个参与者都不知道原始的秘密值 $\langle X\rangle$ [29]. 在两方加法秘密共享中, 为了秘密分享一个值 $\langle X\rangle$, 生成一对秘密份额 $(\langle X\rangle_1,\langle X\rangle_2)$, 使得 $\langle X\rangle=\langle X\rangle_1+\langle X\rangle_2$. 秘密分享保证持有其中一个秘密份额的参与者无法获知任何关于秘密值 $\langle X\rangle$ 的信息. 在该框架中, 客户端和云服务器持有卷积神经网络每一层计算结果的秘密份额, 共同计算而不泄露自己的隐私数据.

2.2 隐私保护预测

隐私保护预测确保客户端可以利用云服务器的深度神经网络模型进行预测, 而不泄露私有数据. 卷积深度神经网络模型由一系列层组成 (例如, 卷积层、ReLU 激活函数层、池化层和全连接层). 前一层的输出是下一层的输入. 当前的研究依赖于多种密码学原语 (例如, 秘密分享、同态加密和混淆电路) 来设计每个层的两方隐私计算协议. 一般来说, 在执行协议之前, 每一层的输入在两个参与者之间进行秘密分享, 并且在运行协议后, 计算结果也在它们之间进行秘密分享, 如图 1 所示. 客户端的输入被秘密分享在云服务器和客户端之间, 表示为 $\langle input\rangle_0$ 和 $\langle input\rangle_1$. $\langle output^i\rangle_0$ 和 $\langle output^i\rangle_1$ 表示第 i 层输出结果的秘密分享, 它们也是第 $i+1$ 层的输入. 然后, 这些协议被组合在一起来完成隐私保护预测. 卷积深度神经网络的线性层包括卷积层和全连接层. 其非线性层包括 ReLU 激活函数层和池化层.

● 卷积层. 卷积层的输入是三维张量 $\boldsymbol{T}$, 输出是三维张量 $\boldsymbol{T}'$. 输出张量是通过线性计算 $\boldsymbol{T}'=\boldsymbol{K}(\boldsymbol{T},\boldsymbol{W})$ 获得的, 其中 $\boldsymbol{W}$ 是四维张量, $\boldsymbol{K}(\cdot)$ 表示卷积计算操作. 在隐私保护预测过程中, 云服务器和客户端都持有 $\boldsymbol{T}$ 的秘密共享份额,

并完成卷积操作. 最后, 双方都持有 $\boldsymbol{T}'$ 的秘密共享份额.

● 全连接层. 全连接层的输入是向量 $\boldsymbol{x}$, 该向量在云服务器和客户端之间被秘密分享, 输出是权重矩阵 $\boldsymbol{W}$ 与向量 $\boldsymbol{x}$ 的矩阵乘法的秘密共享结果. 该过程可以表示为 $\boldsymbol{y}=\boldsymbol{W}\cdot\boldsymbol{x}$.

● ReLU 激活函数层. ReLU 激活函数定义为 $f(x)=\max(x,0)$, 其中 x 是激活函数的输入. 在隐私保护预测中, 在计算 ReLU 激活函数之前, x 在两方之间被秘密分享. 然后, 双方共同计算 ReLU 函数并秘密分享计算结果.

● 池化层. 平均池化和最大池化是卷积深度神经网络中常用的两种池化操作. 与 ReLU 激活函数层类似, 池化层通过云服务器和客户端之间的安全计算协议完成其操作. 池化操作的计算结果也被秘密分享给双方. 本文选择平均池化进行操作.

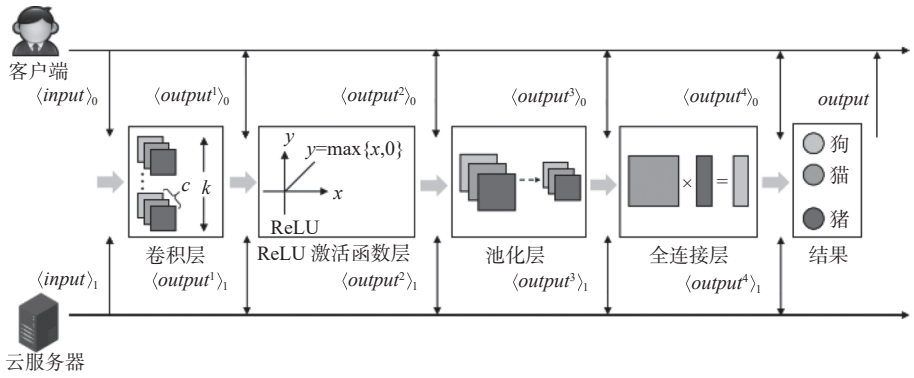

图 1　隐私保护预测的流程示意图

2.3 系统模型

隐私保护预测确保客户端可以利用云服务器的深度神经网络模型进行预测, 而不泄露私有数据. 卷积深度神经网络模型由一系列层组成 (例如, 卷积层、ReLU 激活函数层、池化层和全连接层). 假设客户端 C 希望使用云服务器 S 持有的卷积深度神经网络模型进行预测, 但客户端的输入数据和预测结果是其隐私. 本文目标是在实现隐私保护预测. 该系统分为离线阶段和在线阶段, 如图 2 所示.

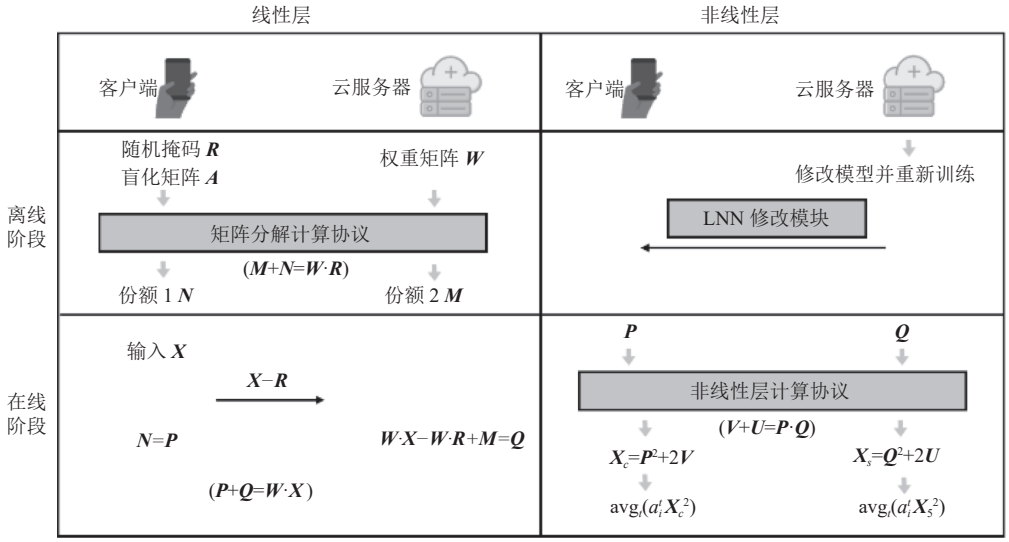

图 2　框架示意图, 包括离线阶段和在线阶段

2.4 威胁模型

本文采用了与之前工作相同的半诚实模型[12,27]. 在这个模型下, 参与者在遵循协议的同时可能会尝试获取彼此的额外信息. 假设模型的结构是公开的, 包括层的类型和数量以及内核的大小[13]. 攻击者可能会采用更高级的攻击方式, 比如模型提取、模型入侵和成员推理. 然而, 该方案并不旨在防御这些攻击, 这超出了该框架的范围, 就如同现有研究工作[9,10,14]. 此外, 我们假设客户端和云服务器之间的所有数据通信都通过安全信道进行, 从而有效阻止窃听攻击.

3 方案描述

本节提出方案框架, 该框架以秘密分享的方式计算线性和非线性层. 对于线性层, 提出了矩阵分解计算协议来进行矩阵乘法. 对于非线性层, 设计了参数化二次多项式来近似计算 ReLU 激活函数.

3.1 安全线性层

卷积深度神经网络中的层大都是卷积层, 例如 ResNet50 中的 49 个卷积层. 卷积层关键计算可以转换为矩阵乘法 $\boldsymbol{F}=\boldsymbol{W}\cdot\boldsymbol{X}$, 其中 $\boldsymbol{W}$ 是通过重塑原始的四维张量得到的二维矩阵, $\boldsymbol{X}$ 是通过将三维张量滑动成矩阵列而形成的. 与此同时, 卷积深度神经网络中的全连接层也是通过矩阵乘法实现的. 然而, 现有工作为了完成矩阵乘法的安全计算, 通常会涉及复杂的加密原语 (例如, 同态加密), 导致计算效率较低.

为了解决这个问题, 我们提出了基于秘密分享矩阵分解计算协议. 通过采用轻量级秘密分享, 加快了线性层矩阵乘法的安全计算效率. 具体地, 在该框架中, 将线性层计算被分为两个阶段: 离线阶段和在线阶段. 在离线阶段, 客户端 C 和云服务器 S 预先计算数据. 这个阶段独立于客户端的输入执行, 这使得双方都能在客户端发起隐私保护预测请求之前完成整个过程.

离线阶段: 此阶段包括 3 个步骤. 首先, 客户端生成随机掩码 $\boldsymbol{R}\in\mathbb{R}^{y\times z}$ 和盲化矩阵 $\boldsymbol{A}\in\mathbb{R}^{y\times z}$. 然后, 客户端持有 $\boldsymbol{R}\in\mathbb{R}^{y\times z}$ 和 $\boldsymbol{A}\in\mathbb{R}^{y\times z}$, 而云服务器持有 $\boldsymbol{W}\in\mathbb{R}^{x\times y}$, 它们作为矩阵分解计算协议的输入 (见协议 1). 最后, 客户端和云服务器都持有 $\boldsymbol{W}\cdot\boldsymbol{R}$ 的秘密分享份额 (即 $\boldsymbol{M}+\boldsymbol{N}=\boldsymbol{W}\cdot\boldsymbol{R}$). 具体过程如下.

步骤 1: 客户端和云服务器需要进行本地数据处理, 即客户端生成随机矩阵 $\boldsymbol{R}'\in\mathbb{R}^{y\times z}$, 并将 $\boldsymbol{R}'$ 分成两个矩阵 $\boldsymbol{R}'_f\in\mathbb{R}^{y/2\times z}$ 和 $\boldsymbol{R}'_s\in\mathbb{R}^{y/2\times z}$, 它们分别是 $\boldsymbol{R}'$ 的前半行和后半行; 云服务器生成随机矩阵 $\boldsymbol{W}'\in\mathbb{R}^{x\times y}$, 并将 $\boldsymbol{W}'$ 分成两个矩阵 $\boldsymbol{W}'_f\in\mathbb{R}^{x\times y/2}$ 和 $\boldsymbol{W}'_s\in\mathbb{R}^{x\times y/2}$, 它们分别是 $\boldsymbol{W}'$ 的前半列和后半列.

步骤 2: 客户端和云服务器进行数据交换, 即客户端计算 $\boldsymbol{A}\cdot\boldsymbol{R}'^{-1}$, $\boldsymbol{R}_1=\boldsymbol{R}+\boldsymbol{R}'+\boldsymbol{A}$ 和 $\boldsymbol{R}_2=\boldsymbol{R}'_s-\boldsymbol{R}'_f$ 并发送云服务器; 云服务器计算 $\boldsymbol{B}=\boldsymbol{W}\cdot\boldsymbol{A}\cdot\boldsymbol{R}'^{-1}$, $\boldsymbol{W}_1=\boldsymbol{W}'-\boldsymbol{W}-\boldsymbol{B}$, $\boldsymbol{W}_2=\boldsymbol{W}'_s+\boldsymbol{W}'_f$ 并发送给客户端.

步骤 3: 客户端和云服务器完成本地计算, 即云服务器计算 $\boldsymbol{M}=\boldsymbol{W}\cdot\boldsymbol{R}_1+\boldsymbol{W}'_f\cdot\boldsymbol{R}_2$; 客户端计算 $\boldsymbol{N}=\boldsymbol{W}_1\cdot\boldsymbol{R}'-\boldsymbol{W}_2\cdot\boldsymbol{R}'_s$.

协议 1. 矩阵分解计算协议.

输入: S 输入矩阵 $\boldsymbol{W}\in\mathbb{R}^{x\times y}$, C 输入随机掩码 $\boldsymbol{R}\in\mathbb{R}^{y\times z}$ 和盲化矩阵 $\boldsymbol{A}\in\mathbb{R}^{y\times z}$;
输出: S 得到矩阵 $\boldsymbol{M}\in\mathbb{R}^{x\times z}$, C 得到矩阵 $\boldsymbol{N}\in\mathbb{R}^{x\times z}$, 其中 $\boldsymbol{M}+\boldsymbol{N}=\boldsymbol{W}\cdot\boldsymbol{R}$.

// 数据准备
1. C 生成随机矩阵 $\boldsymbol{R}'\in\mathbb{R}^{y\times z}$, 并将 $\boldsymbol{R}'$ 分成两个矩阵 $\boldsymbol{R}'_f\in\mathbb{R}^{y/2\times z}$ 和 $\boldsymbol{R}'_s\in\mathbb{R}^{y/2\times z}$, 它们分别是 $\boldsymbol{R}'$ 的前半行和后半行.
2. C 将 $\boldsymbol{A}\cdot\boldsymbol{R}'^{-1}$, $\boldsymbol{R}_1=\boldsymbol{R}+\boldsymbol{R}'+\boldsymbol{A}$ 和 $\boldsymbol{R}_2=\boldsymbol{R}'_s-\boldsymbol{R}'_f$ 发送给 S.
3. S 生成随机矩阵 $\boldsymbol{W}'\in\mathbb{R}^{x\times y}$, 并将 $\boldsymbol{W}'$ 分成两个矩阵 $\boldsymbol{W}'_f\in\mathbb{R}^{x\times y/2}$ 和 $\boldsymbol{W}'_s\in\mathbb{R}^{x\times y/2}$, 它们分别是 $\boldsymbol{W}'$ 的前半列和后半列.
4. S 计算 $\boldsymbol{B}=\boldsymbol{W}\cdot\boldsymbol{A}\cdot\boldsymbol{R}'^{-1}$.
5. S 将 $\boldsymbol{W}_1=\boldsymbol{W}'-\boldsymbol{W}-\boldsymbol{B}$ 和 $\boldsymbol{W}_2=\boldsymbol{W}'_s+\boldsymbol{W}'_f$ 发给 C.
// 结果计算
6. S 计算 $\boldsymbol{M}=\boldsymbol{W}\cdot\boldsymbol{R}_1+\boldsymbol{W}'_f\cdot\boldsymbol{R}_2$.
7. C 计算 $\boldsymbol{N}=\boldsymbol{W}_1\cdot\boldsymbol{R}'-\boldsymbol{W}_2\cdot\boldsymbol{R}'_s$.

在线阶段: 这个阶段包括两个步骤. 首先, 客户端将 $\widetilde{\boldsymbol{X}}=\boldsymbol{X}-\boldsymbol{R}$ 发送给服务器, 因此 $\boldsymbol{X}$ 被秘密分享在客户端和服务器之间. 最后, 服务器计算 $\boldsymbol{W}\cdot(\boldsymbol{X}-\boldsymbol{R})+\boldsymbol{M}$ 作为输出, 而客户端持有在离线阶段获得的 $\boldsymbol{N}$ 作为输出. 在线阶段结束之后, 客户端和服务器获得了 $\boldsymbol{W}\cdot\boldsymbol{X}$ 的秘密分享份额.

基于协议 1 完成线性层 $\boldsymbol{W}\cdot\boldsymbol{X}$ 的安全计算, 且计算结果正确, 具体见第 4.1 节正确性分析, 因此基于秘密分享完成线性层安全计算并不会影响隐私保护预测结果.

3.2 安全非线性层

由于安全计算非线性层的 ReLU 激活函数通常需要依赖计算耗时的混淆电路, 并且会导致系统开销进一步增加. 现有的研究工作大多采用线性近似替换 ReLU 激活函数. 然而, 这种近似处理会导致预测准确率下降. 为了克服这一问题, 提出了使用参数化二次多项式进行近似处理的方案. 通过这种方法, 可以在保持预测准确率的同时, 以秘密共享的方式安全地完成近似计算.

ReLU 激活函数可以表示为 $\max(x,0)$. 与之前的研究工作类似[9,11], 将用二次多项式函数 x^2 来近似替换 ReLU. 完成协议 1 后, 云服务器 S 获得 $\boldsymbol{M}$, 客户端 C 获得 $\boldsymbol{N}$. 将云服务器的 $\boldsymbol{W}\cdot(\boldsymbol{X}-\boldsymbol{R})+\boldsymbol{M}$ 表示为 $\boldsymbol{Q}$, 客户端的 $\boldsymbol{N}$ 表示为 $\boldsymbol{P}$.

为了安全地计算非线性层中的 $(\boldsymbol{P}+\boldsymbol{Q})^2$, 展开表达式得到 $\boldsymbol{P}^2+2\boldsymbol{P}\cdot\boldsymbol{Q}+\boldsymbol{Q}^2$. 使用线性层的输出, 云服务器 S 本地计算 $\boldsymbol{Q}^2$, 客户端 C 本地计算 $\boldsymbol{P}^2$. 云服务器和客户端运行协议 2 安全计算 $\boldsymbol{P}\cdot\boldsymbol{Q}$, 使得双方都拥有秘密共享, 即 $\boldsymbol{U}+\boldsymbol{V}=\boldsymbol{P}\cdot\boldsymbol{Q}$. 最后, 非线性计算的结果进行秘密分享, 以确保非线性层的安全性 (即, 云服务器持有 $\boldsymbol{Q}^2+2\boldsymbol{U}$, 客户端持有 $\boldsymbol{P}^2+2\boldsymbol{V}$). 协议 2 中所述的转换过程分为: 降维转换 (即, 步骤 1), 升维转换 (即, 步骤 5 和步骤 6) 和扩充转换 (即, 步骤 2). 具体地, 降维转换是将矩阵按行顺序转换到一维向量; 而升维转换是将向量按行顺序转换到矩阵. 扩充转换则是将矩阵按行顺序填充到新矩阵对角位置, 其余位置填 0. 如图 3 所示.

协议 2. 非线性层计算协议.

输入: S 输入矩阵 $\boldsymbol{Q}\in\mathbb{R}^{x\times y}$, C 输入矩阵 $\boldsymbol{P}\in\mathbb{R}^{x\times y}$;
输出: S 得到矩阵 $\boldsymbol{U}\in\mathbb{R}^{x\times y}$, C 得到矩阵 $\boldsymbol{V}\in\mathbb{R}^{x\times y}$, 其中 $\boldsymbol{V}+\boldsymbol{U}=\boldsymbol{P}\cdot\boldsymbol{Q}$.

// 数据准备
1. S 将 $\boldsymbol{Q}\in\mathbb{R}^{x\times y}$ 降维转换为一维行向量 $\hat{\boldsymbol{Q}}\in\mathbb{R}^{1\times xy}$.
2. C 将 $\boldsymbol{P}\in\mathbb{R}^{x\times y}$ 扩充转换为矩阵 $\hat{\boldsymbol{P}}\in\mathbb{R}^{xy\times xy}$, 其中将 $\boldsymbol{P}\in\mathbb{R}^{x\times y}$ 的元素填充到对角位置, 其余位置填 0.
3. C 生成盲化矩阵 $\boldsymbol{Z}\in\mathbb{R}^{xy\times xy}$.
// 结果计算
4. $\hat{\boldsymbol{U}}\in\mathbb{R}^{1\times xy}$, $\hat{\boldsymbol{V}}\in\mathbb{R}^{1\times xy}$ 矩阵分解计算协议 $(\hat{\boldsymbol{Q}},\hat{\boldsymbol{P}},\boldsymbol{Z})$.
5. S 将 $\hat{\boldsymbol{U}}\in\mathbb{R}^{1\times xy}$ 升维转换为矩阵 $\boldsymbol{U}\in\mathbb{R}^{x\times y}$.
6. C 将 $\hat{\boldsymbol{V}}\in\mathbb{R}^{1\times xy}$ 升维转换为矩阵 $\boldsymbol{V}\in\mathbb{R}^{x\times y}$.

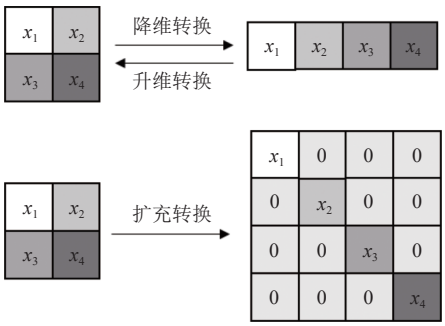

图 3 降维转换、升维转换和扩充转换示意图

为了缓解近似处理预测准确率的负面影响, 提出了一种参数化二次多项式方法来近似激活函数 (即, ax^2). 这些参数允许在近似函数中具有更大的灵活性, 以提高隐私保护预测准确性. 为了学习参数 $a=[a_1^1,\ldots,a_I^1,\ldots,a_1^T,\ldots,a_I^T]$, 其中 T 表示参数个数, I 表示通道个数. 将卷积深度神经网络中 ReLU 层使用轻量级神经网络 (light neural network, LNN) 和求和层进行训练, 以替换图 4 中所示的每个 ReLU 函数. LNN 使用压缩和激励 (squeeze and excitation, SE)[30,31]模块来建模学习参数. 它首先执行平均池化以压缩输入向量 $\boldsymbol{X}$ 的空间信息, 然后通过两个全连接层激励, 并通过归一化层缩放输出. LNN 模型输出 $2TI$ 个元素, 对应于 a_i^t 的残差 Δa_i^t. 之后将残差归一化到 $(-1,1)$. 最后, 通过加权和求和初始值 α^t 和残差来获取最终输出, 即 $a_i^t=\lambda_a\Delta a_i^t+\alpha^t$, 其中 λ_a 是控制残差范围的标量. λ_a 和 α^t 是需要在训练过程中调整的超参数. 在求和层中, 将计算 $\mathrm{avg}_i(a_i^t\boldsymbol{X}^2)$.

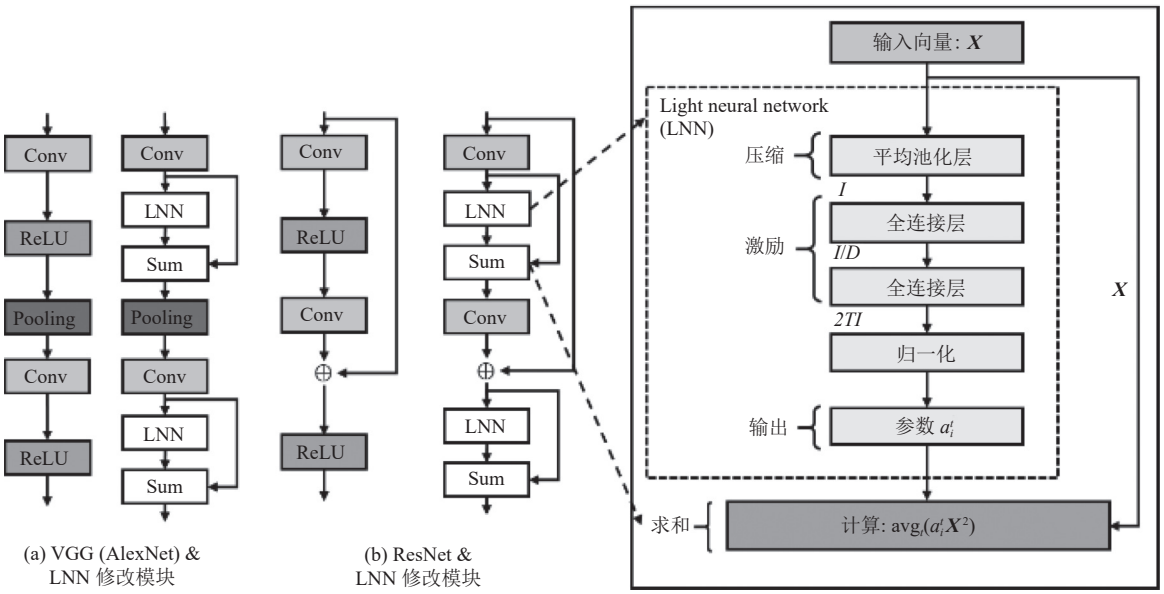

图 4 VGG、AlexNet 和 ResNet 架构使用 LNN 块进行修改示意图

在修改后的模型使用训练数据集进行训练 (与原始卷积深度神经网络模型训练过程相同) 后, 云服务器将每个轻量级神经网络发送给客户端, 可以将其放置在离线阶段. 在隐私保护预测期间, 客户端和云服务器分别获取线性层的输出, 并将它们作为输入传递给 LNN 以安全近似计算近似激活函数. 由于收集了不同通道的信息, 此方法可以更加准确的调整参数, 因此避免近似处理对预测准确率带来的负面影响. 对于池化层, 利用了平均池化, 这在两个秘密份额上很容易评估. 两个参与者只需要在各自的份额上取平均值, 这使得非线性计算能够在秘密份额下完成. 针对非线性层近似处理对于预测结果的影响以及提出的安全非线性层计算协议的有效性, 在第 5.3 节进行分析说明.

4 理论分析

4.1 正确性分析

为了证明协议 1 的正确性, 按照以下步骤进行:

$$\begin{aligned}\boldsymbol{M}+\boldsymbol{N}&=\boldsymbol{W}\cdot\boldsymbol{R}_1+\boldsymbol{W}_f'\cdot\boldsymbol{R}_2+\boldsymbol{W}_1\cdot\boldsymbol{R}'-\boldsymbol{W}_2\cdot\boldsymbol{R}_s'\\&=\boldsymbol{W}\cdot\boldsymbol{R}+\boldsymbol{W}\cdot\boldsymbol{R}'+\boldsymbol{W}\cdot\boldsymbol{A}+\boldsymbol{W}_f'\cdot\boldsymbol{R}_s'-\boldsymbol{W}_f'\cdot\boldsymbol{R}_f'+\boldsymbol{W}'\cdot\boldsymbol{R}'-\boldsymbol{W}\cdot\boldsymbol{R}'-\boldsymbol{B}\cdot\boldsymbol{R}'-\boldsymbol{W}_s'\cdot\boldsymbol{R}_s'-\boldsymbol{W}_f'\cdot\boldsymbol{R}_s'\\&=\boldsymbol{W}\cdot\boldsymbol{R}.\end{aligned}$$

在协议 1 中, 有 $\boldsymbol{W}_1=\boldsymbol{W}'-\boldsymbol{W}-\boldsymbol{B}$, $\boldsymbol{W}_2=\boldsymbol{W}_s'+\boldsymbol{W}_f'$, $\boldsymbol{R}_1=\boldsymbol{R}+\boldsymbol{R}'+\boldsymbol{A}$, $\boldsymbol{R}_2=\boldsymbol{R}_s'-\boldsymbol{R}_f'$ 以及 $\boldsymbol{W}\cdot\boldsymbol{A}-\boldsymbol{B}\cdot\boldsymbol{R}'=0$. 此外, $\boldsymbol{W}_f'\cdot\boldsymbol{R}_f'$ 和 $\boldsymbol{W}_s'\cdot\boldsymbol{R}_s'$ 对应于 $\boldsymbol{W}'\cdot\boldsymbol{R}'$ 的前一半和后一半计算结果. 根据矩阵乘法的定义, 有 $\boldsymbol{W}'\cdot\boldsymbol{R}'=\boldsymbol{W}_f'\cdot\boldsymbol{R}_f'+\boldsymbol{W}_s'\cdot\boldsymbol{R}_s'$. 基于这个

等式, 并将值代入简化后, 可以推导出正确的计算结果 $\boldsymbol{W}\cdot\boldsymbol{R}$.

4.2 安全性分析

为了正式证明协议是安全的, 将采用半诚实威胁模型, 即每个参与者都真实地遵守协议, 同时对其他各方的原始数据感到好奇. 在基于现实世界和理想世界模拟的证明下, 任何一方可以计算的东西都只能在协议期间接收到的消息的情况下进行模拟, 这意味着每一方除了从协议中接收到的消息中得到的东西之外, 从协议执行中没有学到任何东西. 对协议 1 进行安全性分析如下. 执行完协议 1 后, S 从 C 处得到 $\boldsymbol{R}_1$ 和 $\boldsymbol{R}_2$. S 提取 $\boldsymbol{R}_1$ 的前一半和后一半列得到 $\boldsymbol{R}_1^f$ 和 $\boldsymbol{R}_1^s$, 然后计算 $\hat{\boldsymbol{R}}=\boldsymbol{R}_1^f-\boldsymbol{R}_1^s+\boldsymbol{R}_2=\boldsymbol{R}_f+\boldsymbol{R}'_f+\boldsymbol{A}_f-\boldsymbol{R}_s-\boldsymbol{R}'_s-\boldsymbol{A}_s+\boldsymbol{R}'_s-\boldsymbol{R}'_f=\boldsymbol{R}_f-\boldsymbol{R}_s+\boldsymbol{A}_f-\boldsymbol{A}_s$. 在线阶段, 客户端向服务器发送 $\widetilde{\boldsymbol{X}}=\boldsymbol{X}-\boldsymbol{R}$. S 提取 $\widetilde{\boldsymbol{X}}$ 的前一半和后一半列以得到 $\widetilde{\boldsymbol{X}}_f$ 和 $\widetilde{\boldsymbol{X}}_s$, 并计算 $\widetilde{\boldsymbol{X}}_f-\widetilde{\boldsymbol{X}}_s+\hat{\boldsymbol{R}}=\boldsymbol{X}_f-\boldsymbol{R}_f-\boldsymbol{X}_s+\boldsymbol{R}_s+\boldsymbol{R}_f-\boldsymbol{R}_s+\boldsymbol{A}_f-\boldsymbol{A}_s=\boldsymbol{X}_f-\boldsymbol{X}_s+\boldsymbol{A}_f-\boldsymbol{A}_s=\hat{\boldsymbol{X}}$. 因此, 云服务器在协议结束时获得 $\hat{\boldsymbol{X}}$. 然而, 盲化矩阵 $\boldsymbol{A}$ 为客户端 C 随机生成的, 并且对云服务器 S 保密, $\hat{\boldsymbol{X}}$ 不会泄露有关 C 的隐私数据 $\boldsymbol{X}$ 的任何信息.

定理 1. 协议 1 保证客户端隐私数据 $\boldsymbol{X}$ 不会泄露给云服务器.

证明: 从 S 角度来看, 它拥有 $\boldsymbol{W},\boldsymbol{W}',\boldsymbol{R}_1,\boldsymbol{R}_2$ 和 $\hat{\boldsymbol{R}}$. 证明的关键在于需要建立一个模拟器, 以模拟 $\boldsymbol{R}_1$ 和 $\boldsymbol{R}_2$ 的分布. 首先, 模拟器将 $\hat{\boldsymbol{R}}$ 视为先验知识, 并构造一个随机矩阵 $\widetilde{\boldsymbol{R}}\in\mathbb{R}^{y\times z}$. 其次, 模拟器计算 $\widetilde{\boldsymbol{R}}_2=\widetilde{\boldsymbol{R}}_f-\widetilde{\boldsymbol{R}}_s-\hat{\boldsymbol{R}}$, 其中 $\widetilde{\boldsymbol{R}}_f$ 和 $\widetilde{\boldsymbol{R}}_s$ 分别是 $\widetilde{\boldsymbol{R}}$ 的前一半和后一半行. 并且声称 $(\widetilde{\boldsymbol{R}},\widetilde{\boldsymbol{R}}_2)$ 和 $(\boldsymbol{R}_1,\boldsymbol{R}_2)$ 具有相同的分布, 因此在计算上不可区分. 为了理解这一点, 首先注意到 $\boldsymbol{R}_1$ 是固定矩阵 $\boldsymbol{R}$ 和随机矩阵 $\boldsymbol{R}'$ 之间的和, 因此本质上等同于一个随机矩阵, 即 $\widetilde{\boldsymbol{R}}$. 由此可知, 可以将 $\boldsymbol{R}_s-\widetilde{\boldsymbol{R}}_s$ 和 $\boldsymbol{R}'_s$ 视为具有相同的分布. $\boldsymbol{R}_f-\widetilde{\boldsymbol{R}}_f$ 和 $\boldsymbol{R}'_f$ 也视为具有相同的分布. 因此, $\widetilde{\boldsymbol{R}}_2$ 和 $\boldsymbol{R}_2$ 具有相同的分布. 最后, $(\widetilde{\boldsymbol{R}},\widetilde{\boldsymbol{R}}_2)$ 和 $(\boldsymbol{R}_1,\boldsymbol{R}_2)$ 具有相同的分布. 在理想世界中, 模拟器成功地为 S 创建了所有数据, 这意味着在现实世界中只有 $\hat{\boldsymbol{R}}$ 被透露给了 S. 然而, 盲化矩阵 $\boldsymbol{A}$ 为客户端 C 随机生成的, 并且对云服务器 S 保密, $\hat{\boldsymbol{X}}$ 不会泄露有关客户端 C 的隐私数据 $\boldsymbol{X}$ 的任何信息. 证毕.

定理 2. 协议 2 保证客户端隐私数据 $\boldsymbol{X}$ 不会泄露给云服务器.

证明: 由于协议 2 中的 $\hat{\boldsymbol{U}}\in\mathbb{R}^{1\times xy}$ 和 $\hat{\boldsymbol{V}}\in\mathbb{R}^{1\times xy}$ 是基于协议 1 完成计算得到的, 它不会泄露关于客户端 C 的隐私数据 $\boldsymbol{X}$ 的任何信息. 此外, 协议 2 中其他操作均在本地完成, 无需客户端和云服务器进行交互, 因此保证客户端隐私数据 $\boldsymbol{X}$ 不会泄露给云服务器. 证毕.

4.3 效率分析

首先分析协议 1 的计算和通信复杂度. 计算复杂度主要来自于协议 1 中的第 6 行和第 7 行, 其复杂度为 $\mathrm{O}(x\times y\times z)$. 从 S 到 C 的通信复杂度取决于矩阵 $\boldsymbol{W}_1$ 和 $\boldsymbol{W}_2$, 复杂度均为 $\mathrm{O}(x\times y)$. 从 C 到 S 的通信复杂度取决于矩阵 $\boldsymbol{A}\cdot\boldsymbol{R}'^{-1}$, $\boldsymbol{R}_1$ 和 $\boldsymbol{R}_2$, 其复杂度为 $\mathrm{O}(y^2)$, $\mathrm{O}(y\times z)$ 和 $\mathrm{O}(y\times z)$. 因此协议 1 总体通信复杂度为 $\mathrm{O}(y\times(x+y+z))$. 协议 2 由于调用了一次协议 1 的执行, 并且其余操作均在本地完成不涉及额外的通信开销, 且不涉及矩阵乘法计算, 因此协议 2 的总体计算开销为 $\mathrm{O}(x\times y\times z)$, 总体通信开销为 $\mathrm{O}(y\times(x+y+z))$.

假设神经网络模型 L 层网络结构, 其中包括 L_l 层线性层, L_n 层非线性层, 即 $L=L_l+L_n$. 因此该框架总体计算开销为 $\mathrm{O}(L\times x\times y\times z)$, 总体通信开销为 $\mathrm{O}(L\times y\times(x+y+z))$.

5 实验分析

首先, 提供了实验设置. 然后, 展示了所提出框架的效率, 并将其与之前的方案进行了比较. 最后, 评估了该框架的预测准确性.

5.1 实验设置

该框架使用 TensorFlow 库训练卷积深度神经网络, 并应用所提出方案进行隐私保护预测. 所有实验均在硬件设备为 Intel Core i7 CPU, 6 个 3.19 GHz 核心和 15.8 GB 的 RAM 的计算机上进行. 实验在局域网环境中进行. 使用 LeNet、AlexNet、VGG16 和 ResNet18/50 这 5 种不同的卷积深度神经网络架构. 所选的 5 种架构涵盖了从轻量级到深度复杂卷积神经网络模型, 代表了当前卷积神经网络设计中的主流方向. 这有助于验证我们方案在不同

模型复杂度下的有效性. 并且在 MNIST、CIFAR-10 和 CIFAR-100 数据集上进行实验. MNIST 数据集包含 60 000 张 28×28 大小的手写数字 0–9 的灰度图像, 其中 50 000 张用于训练, 10 000 张用于测试. CIFAR-10 和 CIFAR-100 数据集包含 50 000 个训练图像和 10, 000 个测试图像, 大小为 32×32. 这两个数据集之间唯一的区别是类别数量, CIFAR-10 有 10 个类别, 而 CIFAR-100 有 100 个类别. 3 个数据集分别具有不同的规模和应用场景, 能够充分反映在多样化环境中方案的有效性. 并且所选择数据集与之前研究工作[9,12,13,25,26]所选择的保持一致. 这样的设计确保我们的评估结果既具有代表性, 又能展示方案在不同任务下的性能表现.

5.2 线性层计算测试

设计了用于矩阵乘法的高效线性计算方法, 对具有不同输入尺寸的卷积层和全连接层进行测试, 并与之前的研究工作方案进行比较. 每个层的所有输入数据都是随机采样的, 并且卷积层和全连接层的大小来自第 5.1 节中描述的模型.

我们进行了 4 个全连接层测试, 矩阵维度为 (n_0, n_1). 表 1 展示了使用不同方法进行矩阵乘法的计算时间和通信开销. 与先前基于同态加密的工作相比, 提出的协议减少了 2–10 倍计算时间, 而且通信开销大致相同.

表 1　矩阵乘法的基准测试

方法	1×2048		16×1024		128×1024		512×1024	
	通信开销 (MB)	时间开销 (ms)	通信开销 (MB)	时间开销 (ms)	通信开销 (MB)	时间开销 (ms)	通信开销 (MB)	时间开销 (ms)
Gazelle[12]	0.30	5.81	0.55	9.41	0.90	14.86	2.39	17.26
Delphi[9]	0.36	4.63	0.48	7.31	0.89	11.95	2.56	14.34
CrypTFlow[25]	0.38	3.48	0.43	6.63	0.85	10.88	2.62	12.58
GALA[13]	0.30	3.06	0.46	5.85	0.88	8.02	2.33	10.72
Cheetah[14]	0.32	0.87	0.52	2.66	0.85	3.63	2.66	8.94
FastSecNet[19]	0.27	2.13	0.33	3.46	0.81	6.34	1.96	11.39
Ours	**0.10**	**0.43**	**0.12**	**1.05**	**0.72**	**2.61**	**1.31**	**6.93**

对于卷积层计算基准测试, 测试了不同的输入和核大小. 输入尺寸表示为三维张量 $(u_w \times u_h \times c_i)$, 其中 u_w 和 u_h 分别是输入的宽度和高度, c_i 是输入的通道数. 卷积核的大小为 $(k_w \times k_h \times k_i)$, 其中 k_w 和 k_h 是卷积核的宽度和高度, k_i 是卷积核的通道数. 将卷积计算转换为矩阵乘法后, 该框架计算时间减少了大约 3 倍并且通信开销大致相同, 如表 2 所示. 由于卷积深度神经网络通常涉及大量的卷积操作, 所提出方案在卷积方面具有更好的计算效率, 更适合于隐私保护预测.

表 2　卷积计算的基准测试

方法	16×16×2048 1×1×128		16×16×2048 3×3×128		32×32×32 3×3×32		16×16×2048 5×5×2048	
	通信开销 (MB)	时间开销 (ms)	通信开销 (MB)	时间开销 (ms)	通信开销 (MB)	时间开销 (ms)	通信开销 (MB)	时间开销 (ms)
Gazelle[12]	0.50	45.05	0.50	74.61	0.21	109.42	2.47	852.16
Delphi[9]	0.57	48.21	0.57	88.36	0.18	111.63	1.98	873.81
GALA[13]	0.40	36.45	0.40	77.48	0.19	104.26	2.33	803.93
Cheetah[14]	0.45	25.34	0.45	52.45	0.17	65.32	1.65	703.45
FastSecNet[19]	0.43	30.19	0.46	64.92	0.20	86.32	1.84	842.13
Ours	**0.33**	**16.13**	**0.33**	**32.98**	**0.14**	**40.01**	**1.43**	**395.93**

5.3 性能评估

时间开销和通信开销: 首先使用 4 个经典简单卷积深度神经网络模型在 MNIST 数据集上评估所提出的方案, 这些模型由全连接层和卷积层的组合组成: A: 3 个全连接层, 采用 ReLU 激活函数; B: 1 个卷积层和 1 个全连接

层, 采用 ReLU 激活函数; C: 1 个卷积层和 2 个全连接层, 采用 ReLU 激活函数; D: 2 个卷积层和 3 个全连接层, 采用 ReLU 激活函数和平均池化 (即, LeNet). 该方案在时间上优于其他方法, 时间开销减少了 3 倍, 并且通信开销大致相同, 如图 5 所示.

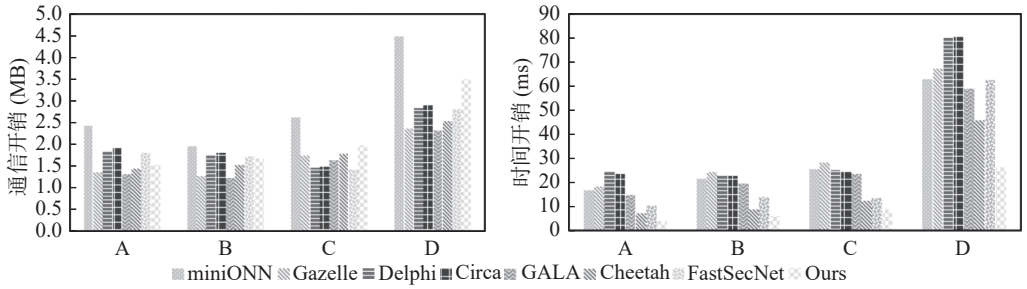

图 5 经典卷积深度神经网络时间和通信开销

然而, 这些模型在实际应用中并不常用. 卷积深度神经网络模型通常采用大通道和小内核来捕获更多的特征, 所提出的框架在这些结构中表现更好, 例如 AlexNet、VGG 和 ResNet18/50. 表 3 展示了在 CIFAR-10 上使用不同方案的 CNN 时间和通信开销. 与 FastSecNet 相比, 预测时间平均减少 8 倍. 我们的方案比 Cheetah 快约 2 倍, 并且与 CrypTFlow 相比, 通过完全避免混淆电路, 平均加速了 12 倍. 与基于同态加密和混淆电路的 Gazelle 和 GALA 相比, 我们的方案分别提供了大约 15 倍和 10 倍的时间加速, 这是由于我们的高效线性层计算协议和参数化二次多项式近似计算非线性层. 相较于同态加密和传统安全多方计算方法, 我们采用了轻量级密码原语秘密分享来设计安全计算协议, 显著降低了计算开销和通信开销. 此外, 我们对线性层和非线性层分别进行了针对性优化, 特别是非线性层采用了适当的近似技术, 有效降低了计算复杂度且保证了预测准确率.

表 3 在 CIFAR-10 数据集上完成隐私保护卷积神经网络预测的时间和通信开销

方法	AlexNet		VGG16		ResNet18		ResNet50	
	通信开销 (MB)	时间开销 (s)	通信开销 (MB)	时间开销 (s)	通信开销 (MB)	时间开销 (s)	通信开销 (MB)	时间开销 (s)
Gazelle[12]	463.18	12.10	551.95	18.94	652.64	23.21	4226.13	71.79
Delphi[9]	962.13	7.65	1396.62	9.52	2536.92	15.63	7803.6	43.26
CrypTFlow[25]	833.16	9.85	1629.65	14.25	3186.36	22.63	9915.32	56.24
Circa[26]	613.72	3.73	676.31	5.16	709.28	12.92	5413.29	25.17
GALA[13]	420.12	4.84	530.13	7.29	631.15	14.42	4210.78	32.10
COINN[27]	862.16	1.68	956.23	2.27	1576.35	4.97	4276.13	7.23
Cheetah[14]	377.83	2.05	615.38	4.56	1493.62	3.74	4305.78	5.93
FastSecNet[19]	563.16	6.32	842.93	10.39	2263.93	19.96	6346.93	48.32
Ours	**353.24**	**0.98**	**467.51**	**1.16**	**588.72**	**2.51**	**3620.34**	**3.86**

在表 4 中, 我们的方案在不同的卷积神经网络中计算速度比快约 2 倍. 此外, 将我们的方案与其他工作进行了计算速度提升的比较. 与现有方法相比, 所提出的方案在隐私保护预测方面速度提升了 2–13 倍, 通信成本降低了 50%–75%. 这是因为我们的方案整个隐私保护预测过程都是在秘密分享方式完成的, 避免了复杂的加解密计算, 因此提高了系统的性能. 这些计算效率的提高不仅使得隐私保护预测更加实用, 而且增强了系统的可用性, 节省了大量的时间.

预测准确率: 在非线性激活函数领域的先前工作中, 通常依赖混淆电路来计算 ReLU. 虽然混淆电路确保了预测准确率, 但它带来了高昂的通信成本, 使得在实际场景中变得不切实际. 在该框架中, 采用参数化二次多项式来近似计算 ReLU. 然而, 使用近似方法来计算激活函数必然会导致预测准确率的降低. 为了解决这个问题, 通过训练 LNN 来学习一组系数. 然后, 将近似表达式乘以相应的系数来确保隐私保护预测准确性.

表 4　在 CIFAR-100 数据集上完成隐私保护卷积神经网络预测的时间和通信开销

方法	AlexNet		VGG16		ResNet18		ResNet50	
	通信开销 (MB)	时间开销 (s)	通信开销 (MB)	时间开销 (s)	通信开销 (MB)	时间开销 (s)	通信开销 (MB)	时间开销 (s)
Gazelle[12]	424.29	13.40	532.63	19.46	664.16	27.60	4 323.29	73.17
Delphi[18]	1 236.32	7.76	1 679.33	9.76	3 036.45	16.85	8 974.60	45.62
CrypTFlow[25]	851.75	11.56	1 632.95	16.89	3 698.62	24.90	9 827.95	58.67
Circa[26]	603.11	3.47	693.26	5.71	733.99	14.23	5 673.12	26.93
GALA[13]	413.18	5.03	524.17	7.35	642.51	14.68	4 128.39	38.16
COINN[27]	989.46	1.92	1 156.36	2.75	1 426.32	3.03	4 423.52	7.41
Cheetah[14]	407.23	2.65	618.23	3.56	635.72	4.56	4 901.54	9.64
FastSecNet[19]	596.26	8.23	996.21	13.26	2 563.16	26.32	6 965.16	52.13
Ours	**389.96**	**1.38**	**503.36**	**1.71**	**606.31**	**2.59**	**3 793.28**	**6.86**

为了展示参数化二次多项式近似计算非线性层激活函数与基于混淆电路的方案 (例如, Delphi 和 Circa) 对预测准确率的影响, 我们使用 4 种不同的数据集 (即 MNIST、CIFAR-10 和 CIFAR-100), 在 LeNet、ResNet18 和 ResNet50 中进行实验评估, 并将其与明文模型的预测准确率进行比较. 如表 6 所示, 所提出框架导致预测准确率下降约 2%. 在隐私保护预测中, 该准确率的损失是可以忽略的.

表 5　卷积神经网络在不同数据集上的预测准确率 (%)

数据集	Ours	Delphi[9]	Circa[26]	明文
MNIST (LeNet)	97.38	97.06	97.52	98.75
CIFAR-10 (ResNet18)	79.32	78.88	79.26	80.68
CIFAR-100 (ResNet50)	70.33	69.23	70.89	72.78

参数化二次多项式近似的影响: 为了展示所提出的用于计算 ReLU 激活函数的参数化二次多项式近似方法的有效性, 进行了消融实验, 如图 6 所示. 结果表明, 参数化近似方法缓解了近似对预测准确率带来的负面影响, 并将预测准确率提高了 10%, 并更接近明文训练准确率. 这种方法允许参数适应输入元素, 并考虑到不同通道信息, 从而更准确地近似 ReLU 激活函数. 总体来说, 我们的方案在保证数据隐私的前提下, 计算效率和通信效率取得了显著提升, 同时对预测准确率的影响极小.

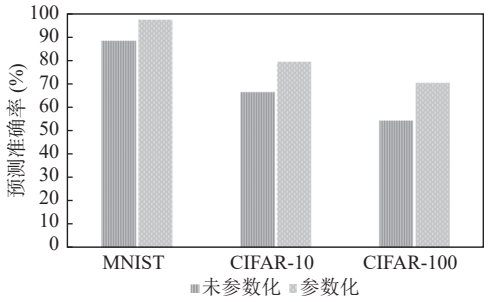

图 6　未参数化和参数化二次多项式近似表达式预测准确率

修改模型架构的影响: 在安全非线性层使用一种轻量级神经网络和求和层替代每一层 ReLU 层来训练模型, 以此缓解近似处理 ReLU 导致预测准确率降低的问题. 然而, 这将会导致额外的重新训练时间以及存储开销. 表 7 展示了修改模型后重新训练时间和额外存储开销的实验结果. 例如, 在 CIFAR-10 上训练 ResNet18 模型需要大约 2 h. 由于只替换了非线性层, 重新训练时间仅有 28.93 min. 原始 ResNet18 模型的存储大小为 450 MB, 而修改后的模型仅增加了 17.92 MB 的存储开销. 通过对比, 可以认为重新训练时间以及额外的存储开销可以接受. 由于重新训练可以在离线阶段完成, 这对在线阶段的隐私预测效率没有影响.

表 6 修改模型架构后重新训练时间以及额外存储开销

开销类别	MNIST (LeNet)	CIFAR-10 (ResNet18)	CIFAR-100 (ResNet50)
时间开销 (min)	5.23	28.93	41.63
存储开销 (MB)	2.39	17.62	22.36

6 总 结

在本研究工作中, 提出了一种新的隐私保护预测框架, 确保了客户端的隐私数据. 为了提高效率, 设计了一种基于秘密共享的矩阵分解计算协议以及参数化二次多项式近似计算 ReLU 方法. 实验结果表明, 所提出的框架比现有方案快 2–15 倍, 同时保持了预测准确率. 然而, 我们仅关注在卷积神经网络的隐私预测. 在未来, 将扩展该框架到 Transformer 等不同模型架构上, 计划探索模块化设计以增强方法的泛化能力, 以及进一步在更多不同类型的数据集上验证和优化该方法. 此外, 当前的研究集中在两个关键领域: 优化神经网络结构和设计更高效的协议. 探索这些领域可以进一步提高隐私保护预测的安全性和效率.

References

[1] Deng JK, Guo J, Xue NN, Zafeiriou S. ArcFace: Additive angular margin loss for deep face recognition. In: Proc. of the 2019 IEEE/CVF Conf. on Computer Vision and Pattern Recognition. Long Beach: IEEE, 2019. 4685–4694. [doi: 10.1109/CVPR.2019.00482]

[2] Schroff F, Kalenichenko D, Philbin J. FaceNet: A unified embedding for face recognition and clustering. In: Proc. of the 2015 IEEE Conf. on Computer Vision and Pattern Recognition. Boston: IEEE, 2015. 815–823. [doi: 10.1109/CVPR.2015.7298682]

[3] Richens JG, Lee CM, Johri S. Improving the accuracy of medical diagnosis with causal machine learning. Nature Communications, 2020, 11(1): 3923. [doi: 10.1038/s41467-020-17419-7]

[4] Topol EJ. High-performance medicine: The convergence of human and artificial intelligence. Nature Medicine, 2019, 25(1): 44–56. [doi: 10.1038/s41591-018-0300-7]

[5] Shokri R, Shmatikov V. Privacy-preserving deep learning. In: Proc. of the 22nd ACM SIGSAC Conf. on Computer and Communications Security. Denver: ACM, 2015. 1310–1321. [doi: 10.1145/2810103.2813687]

[6] Bonawitz K, Ivanov V, Kreuter B, Marcedone A, McMahan HB, Patel S, Ramage D, Segal A, Seth K. Practical secure aggregation for privacy-preserving machine learning. In: Proc. of the 2017 ACM SIGSAC Conf. on Computer and Communications Security. Dallas: ACM, 2017. 1175–1191. [doi: 10.1145/3133956.3133982]

[7] Official Journal of the European Union. General data protection regulation. 2021. https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679

[8] Song L, Ma CG, Duan GH. Machine learning security and privacy: A survey. Chinese Journal of Network and Information Security, 2018, 4(8): 1–11 (in Chinese with English abstract). [doi: 10.11959/j.issn.2096-109x.2018067]

[9] Mishra P, Lehmkuhl R, Srinivasan A, Zheng WT, Popa RA. Delphi: A cryptographic inference service for neural networks. In: Proc. of the 29th USENIX Conf. on Security Symp. ACM, 2020. 141.

[10] Mohassel P, Rindal P. ABY^3: A mixed protocol framework for machine learning. In: Proc. of the 2018 ACM SIGSAC Conf. on Computer and Communications Security. Toronto: ACM, 2018. 35–52. [doi: 10.1145/3243734.3243760]

[11] Dowlin N, Gilad-Bachrach R, Laine K, Lauter K, Naehrig M, Wernsing J. CryptoNets: Applying neural networks to encrypted data with high throughput and accuracy. In: Proc. of the 33rd Int'l Conf. on Machine Learning. New York: ACM, 2016. 201–210.

[12] Juvekar C, Vaikuntanathan V, Chandrakasan A. Gazelle: A low latency framework for secure neural network inference. In: Proc. of the 27th USENIX Conf. on Security Symp. Baltimore: ACM, 2018. 1651–1668.

[13] Zhang Q, Xin CS, Wu HY. GALA: Greedy computation for linear algebra in privacy-preserved neural networks. In: Proc. of the 28th Annual Network and Distributed System Security Symp. The Internet Society, 2021.

[14] Huang ZC, Lu WJ, Hong C, Ding JS. Cheetah: Lean and fast secure two-party deep neural network inference. In: Proc. of the 31st USENIX Security Symp. Boston: USENIX Association, 2022. 809–826.

[15] Ball M, Carmer B, Malkin T, Rosulek M, Schimanski N. Garbled neural networks are practical. 2019. https://eprint.iacr.org/2019/338.pdf

[16] Liu J, Juuti M, Lu Y, Asokan N. Oblivious neural network predictions via miniONN transformations. In: Proc. of the 2017 ACM

SIGSAC Conf. on Computer and Communications Security. Dallas: ACM, 2017. 619–631. [doi: 10.1145/3133956.3134056]

[17] Knott B, Venkataraman S, Hannun A, Sengupta S, Ibrahim M, van der Maaten L. CrypTen: Secure multi-party computation meets machine learning. In: Proc. of the 35th Int'l Conf. on Neural Information Processing Systems. ACM, 2021. 379.

[18] Avci O, Abdeljaber O, Kiranyaz S, Hussein M, Gabbouj M, Inman DJ. A review of vibration-based damage detection in civil structures: From traditional methods to machine learning and deep learning applications. Mechanical Systems and Signal Processing, 2021, 147: 107077. [doi: 10.1016/j.ymssp.2020.107077]

[19] Hao M, Li HW, Chen HX, Xing PZ, Zhang TW. FastSecNet: An efficient cryptographic framework for private neural network inference. IEEE Trans. on Information Forensics and Security, 2023, 18: 2569–2582. [doi: 10.1109/TIFS.2023.3262149]

[20] Hesamifard E, Takabi H, Ghasemi M. CryptoDL: Deep neural networks over encrypted data. arXiv:1711.05189, 2017.

[21] Brutzkus A, Gilad-Bachrach R, Elisha O. Low latency privacy preserving inference. In: Proc. of the 36th Int'l Conf. on Machine Learning. Long Beach: PMLR, 2019. 812–821.

[22] Riazi MS, Samragh M, Chen H, Laine K, Lauter K, Koushanfar F. XONN: XNOR-based oblivious deep neural network inference. In: Proc. of the 28th USENIX Conf. on Security Symp. Santa Clara: ACM, 2019. 1501–1518.

[23] Li SH, Xue KP, Zhu B, Ding CK, Gao XD, Wei D, Wan T. FALCON: A Fourier transform based approach for fast and secure convolutional neural network predictions. In: Proc. of the 2020 IEEE/CVF Conf. on Computer Vision and Pattern Recognition. Seattle: IEEE, 2020. 8702–8711. [doi: 10.1109/CVPR42600.2020.00873]

[24] Bian S, Wang TC, Hiromoto M, Shi YY, Sato T. ENSEI: Efficient secure inference via frequency-domain homomorphic convolution for privacy-preserving visual recognition. In: Proc. of the 2020 IEEE/CVF Conf. on Computer Vision and Pattern Recognition. Seattle: IEEE, 2020. 9400–9409. [doi: 10.1109/CVPR42600.2020.00942]

[25] Kumar N, Rathee M, Chandran N, Gupta D, Rastogi A, Sharma R. CrypTFlow: Secure TensorFlow inference. In: Proc. of the 2020 IEEE Symp. on Security and Privacy. San Francisco: IEEE, 2020. 336–353. [doi: 10.1109/SP40000.2020.00092]

[26] Ghodsi Z, Jha NK, Reagen B, Garg S. Circa: Stochastic ReLUs for private deep learning. In: Proc. of the 35th Int'l Conf. on Neural Information Processing Systems. ACM, 2021. 172.

[27] Hussain SU, Javaheripi M, Samragh M, Koushanfar F. COINN: Crypto/ML codesign for oblivious inference via neural networks. In: Proc. of the 2021 ACM SIGSAC Conf. on Computer and Communications Security. ACM, 2021. 3266–3281. [doi: 10.1145/3460120.3484797]

[28] Kim D, Guyot C. Optimized privacy-preserving CNN inference with fully homomorphic encryption. IEEE Trans. on Information Forensics and Security, 2023, 18: 2175–2187. [doi: 10.1109/TIFS.2023.3263631]

[29] Beimel A. Secret-sharing schemes: A survey. In: Proc. of the 3rd Int'l Conf. on Coding and Cryptology. Qingdao: Springer, 2011. 11–46. [doi: 10.1007/978-3-642-20901-7_2]

[30] Hu J, Shen L, Albanie S, Sun G, Wu EH. Squeeze-and-excitation networks. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2020, 42(8): 2011–2023. [doi: 10.1109/TPAMI.2019.2913372]

[31] Liu JX, Meng XF. Survey on privacy-preserving machine learning. Journal of Computer Research and Development, 2020, 57(2): 346–362 (in Chinese with English abstract). [doi: 10.7544/issn1000-1239.2020.20190455]

附中文参考文献

[8] 宋蕾, 马春光, 段广晗. 机器学习安全及隐私保护研究进展. 网络与信息安全学报, 2018, 4(8): 1–11. [doi: 10.11959/j.issn.2096-109x.2018067]

[31] 刘俊旭, 孟小峰. 机器学习的隐私保护研究综述. 计算机研究与发展, 2020, 57(2): 346–362. [doi: 10.7544/issn1000-1239.2020.20190455]

作者简介

白浩, 博士生, 主要研究领域为隐私保护机器学习.

何琨, 博士, 副教授, CCF 专业会员, 主要研究领域为应用密码学, 网络安全, 云计算安全, 人工智能安全, 区块链安全.

陈晶, 博士, 教授, 博士生导师, CCF 高级会员, 主要研究领域为网络安全, 分布式系统安全, 区块链.

赵陈斌, 博士生, 主要研究领域为应用密码学, 可搜索加密.

杜瑞颖, 博士, 教授, 博士生导师, 主要研究领域为网络安全, 隐私保护.