文章编号: 1000-5641(2023)05-0077-13

支持隐私保护的端云协同训练

高祥云[1], 孟 丹[2], 罗明凯[2,3], 王 俊[2], 张丽平[1], 孔 超[1,4]

(1. 安徽工程大学 计算机与信息学院, 安徽 芜湖 241000; 2. OPPO 研究院, 广东 深圳 518000;
3. 同济大学 电子与信息工程学院, 上海 201804;
4. 安徽工程大学 可重构与智能计算实验室, 安徽 芜湖 241000)

摘要: 我国在数据资源上具有规模化和多样化的优势, 在移动互联网数据应用上具有后发优势, 在丰富的应用场景下产生了海量数据, 推荐系统可以从大规模数据中挖掘有价值的信息, 缓解信息过载问题. 已有的工作聚焦于集中式推荐, 数据在云侧训练. 随着数据安全和隐私保护问题的日益突出, 从端侧设备收集用户数据变得越发困难, 这使得集中式推荐变得不可行. 以去中心化的方式, 利用端侧设备和云服务器的优势, 充分考虑数据安全与隐私保护问题, 面向推荐系统, 提出了一个基于联邦机器学习 (federated machine learning, FedML) 与移动神经网络 (mobile neural network, MNN) 的端云协同训练方法 FedMNN (federated machine learning and mobile neural network). 具体分为 3 部分: 首先, 将多种深度学习框架实现的云侧模型以 ONNX (open neural network exchange) 作为中间框架通过 MNN 模型转换工具转换成通用 MNN 模型供端侧设备训练; 然后, 云侧将模型下发给端侧设备, 端侧初始化后, 获取本地数据进行训练并计算损失, 再执行梯度反向传播; 最后, 端侧训练后的模型反馈给云侧, 通过联邦学习框架进行模型聚合与更新, 再根据不同需求, 将云侧模型按需部署到端侧设备上, 实现端云协同. 实验通过对比 FedMNN 和 FLTFlite (flower and TensorFlow lite) 框架在基准任务上的功耗, 发现 FedMNN 比 FLTFlite 低 32% ∼ 51%, 并以 DSSM (deep structured semantic model) 和 Deep and Wide 这 2 个推荐模型为例, 实验验证了端云协同训练的有效性.

关键词: 隐私保护; 联邦学习; 机器学习; 端云协同训练
中图分类号: TP181 **文献标志码**: A **DOI**: 10.3969/j.issn.1000-5641.2023.05.007

Privacy-preserving cloud-end collaborative training

GAO Xiangyun[1], MENG Dan[2], LUO Mingkai[2,3], WANG Jun[2],
ZHANG Liping[1], KONG Chao[1,4]

(1. *School of Computer and Information, Anhui Polytechnic University, Wuhu, Anhui* 241000, *China*;
2. *OPPO Research Institute, Shenzhen, Guangdong* 518000, *China*; 3. *College of Electronic and Information Engineering, Tongji University, Shanghai* 201804, *China*; 4. *Reconfigurable and Intelligent Computing Laboratory, Anhui Polytechnic University, Wuhu, Anhui* 241000, *China*)

Abstract: China has the advantages of scale and diversity in data resources, and mobile internet data applications, which generate massive amounts of data in diverse application scenarios, recommendation systems have the capability to extract valuable information from this massive amounts of data, thereby mitigating the problem of information overload. Most existing research on recommendation systems focused on centralized recommender systems, training the data on the cloud centrally. However, with increasingly

收稿日期: 2023-06-30
基金项目: 国家自然科学基金 (61902001) ; 安徽工程大学本科教学质量提升计划项目 (2022lzyybj02)
通信作者: 孔 超, 男, 副教授, 硕士生导师, 研究方向为海量数据挖掘. E-mail: kongchao@ahpu.edu.cn

prominent data security and privacy protection issues, collecting user data has become increasingly difficult, making centralized recommendation methods infeasible. This study focuses on privacy-preserving cloud-end collaborative training in a decentralized manner for personalized recommender systems. To fully utilize the advantages of end devices and cloud servers while considering privacy and security issues, a cloud-end collaborative training method named FedMNN (federated machine learning and mobile neural network) is proposed for recommender systems based on federated machine learning (FedML) and a mobile neural network (MNN). The proposed method was divided into three parts: First, cloud-based models implemented in various deep learning frameworks were converted into general MNN models for end-device training using the ONNX (open neural network exchange) intermediate framework and a MNN model conversion tool. Second, the cloud server sends the model to the end-side devices, which initialized and obtain local data for training and loss calculation, followed by gradient back-propagation. Finally, the end-side models are fed back to the cloud server for model aggregation and updating. Depending on different requirements, the cloud model was deployed on end-side devices as required, achieving end-cloud collaboration. Experiments comparing power consumption of the proposed FedMNN and FLTFlite (flower and TensorFlow lite) frameworks on benchmark tasks identified that FedMNN is 32% to 51% lower than FLTFlite. Using DSSM (deep structured semantic model) and deep and wide recommendation models, the experimental results demonstrated the effectiveness of the proposed cloud-end collaborative training method.

Keywords: privacy protection; federated learning; machine learning; cloud-end collaborative training

0 引 言

随着大数据和物联网应用的不断普及, 大量数据分散在多种设备和系统中. 这些数据蕴含着丰富的价值, 与用户密切相关, 同时也可能包含诸多敏感信息, 如姓名、性别、家庭住址等. 传统的机器学习方法需要将数据集中到一个地方进行训练, 集中式训练可能导致用户隐私泄露和数据安全等问题. 因此, 支持隐私保护的机器学习方法得到越来越多的重视. 传统的数据加密方法在一定程度上保护了数据隐私, 也在一定程度上阻碍了数据的共享与流通, 出现“数据孤岛”现象, 由此, 联邦学习[1] 应运而生. 在联邦学习框架中, 数据以分布式方式存储, 本地训练出局部模型, 以通信的方式进行各设备的局部模型聚合与更新, 从而获得全局模型. 这种方式充分利用了分布式计算的优势, 提升了模型的准确性和泛化能力. 联邦学习是一种分布式机器学习技术, 能够让多个参与方在不披露底层数据的前提下, 通过交换加密的机器学习中间结果实现联合建模. 这一分布式机器学习架构涉及多个客户端和一个聚合服务器. 客户端可以是个人终端设备 (如手机、平板电脑等) 、不同部门或企业等, 负责保存用户个人数据或组织私有数据. 客户端在本地训练模型, 将训练后的模型参数发送给聚合服务器. 聚合服务器负责聚合部分或全部客户端的模型参数, 并将聚合后的模型同步到客户端以开始新一轮的训练. 这种联合协作训练方式可以在保证模型性能的前提下, 避免个人隐私数据泄露, 并有效解决数据孤岛问题.

近年来研究发现, 联邦学习框架中, 在梯度传输时仍然会泄露隐私信息, 甚至会泄露原始数据[2], 由此发展出了梯度安全聚合技术, 主要包括差分隐私[3]、同态加密[4] 等. 然而, 这些技术仍然存在一些问题: 本地差分隐私需要对数据进行足量的扰动以保证每个用户所上传数据的隐私不被泄露, 隐私保护程度与模型性能之间难以平衡是个巨大的挑战; 同态加密的计算及通信代价比较高, 导致难以被应用到资源受限的环境中; 分布式设备间的数据通常是非独立同分布的, 数据量也是不均衡的, 这意味着数据存在异构性[5-6]. SCAFFOLD[7] 在服务器和客户端添加了控制变量以调整模型训练过程的更新方向, 但每轮通信时, 控制变量会随着模型一同传输, 增加了额外的通信成本. 此外, 当设备参与率很

低时, 客户端可能无法参与每轮的模型训练, 导致其控制变量陈旧, 从而降低算法性能.

端云协同训练为解决上述问题提供了新思路. 端云协同训练是一种利用端侧设备和云服务器协同计算的机器学习范式, 能够在分布式数据上高效、安全、协作地训练模型. 端云协同训练能充分利用端侧设备的计算能力和数据多样性, 弥补云侧计算资源不足、突破端侧数据安全与隐私的瓶颈, 同时达到提升模型训练效率和性能的目的. 目前, 主流的端云协同训练框架有 FedML[8] 和 Flower[9], 它们分别采用 MNN[10] 和 TFlite (TensorFlow lite)(https://www.tensorflow.org/lite) 作为端侧训练框架, 通过端侧设备和云服务器的协同工作, 实现高效的模型训练和保护用户数据隐私. 本文以安卓设备为例, 对这两个框架在基准任务上的功耗情况进行了对比, 发现 FedML 与 MNN 组合的 FedMNN 框架, 和 Flower 与 TFlite 组合的 FLTFlite 框架在进行端云协同训练过程中, FedMNN 在端侧设备上的单轮平均功耗比 FLTFlite 低 32% ∼ 51%. 因此, 本文以 FedMNN 框架为例, 首先, 介绍端云协同训练的流程; 其次, 总结了实现端云协同训练的通用方法; 最后, 以 DSSM (deep structured semantic model) 和 Deep and Wide 两个推荐模型为例, 通过实验验证了端云协同训练的有效性.

1　相关工作

1.1　联邦学习框架

联邦学习是一种保护隐私和安全的数据共享方法, 通过中央服务器 (如服务提供商) 的协调, 使多个数据持有方 (如移动设备) 能够进行模型训练. 在联邦学习训练过程中, 各参与方的原始数据始终保留在本地, 服务器主要通过加密机制在参数交换中建立共享模型[11]. 根据数据分布的特征, 联邦学习分为横向联邦学习、纵向联邦学习与联邦迁移学习[12]. 横向联邦学习又被称为基于样本的联邦学习, 横向联邦学习表示数据按行划分, 特征不变; 纵向联邦学习又被称为基于特征的联邦学习, 数据按列划分, 特征在变, 样本 ID 空间不变; 联邦迁移学习在应用场景中, 两个参与方的数据集特征不同且样本也不同. 随着联邦学习研究的不断深入, 适用于不同场景的联邦学习框架相继被推出, 以满足不同应用需求.

中国信息通信研究院发布的报告显示, 截至 2020 年共有 18 款联邦学习产品通过评测, 超过 60 家企业拥有联邦学习框架和产品. 除了 TensorFlow federated、PySyft[13]、FedML 和 Flower, 国内较为成熟的联邦学习框架还有微众银行开发的 FATE[14] 和百度开发的 PaddleFL[15].

谷歌于 2019 年推出了基于 TensorFlow 构建的移动设备端联邦学习框架 (TensorFlow federated, TFF), 推动联邦学习在移动智能设备上的实验. TFF 支持横向联邦学习, 目前尚未支持纵向联邦学习和联邦迁移学习. TFF 提供了多种算法, 包括 FedAvg (federated averaging algorithm)[16] 和 Fed-SGD (federated stochastic gradient descent) 等, 可支持神经网络和线性模型. TFF 可在单机设备模拟和移动设备训练, 并采用差分隐私来保证数据安全. 然而, TFF 不支持基于拓扑结构的分布式训练.

FATE (federated AI technology enabler) 是微众银行于 2019 年开源的联邦学习框架, 旨在实现支持数据安全与隐私的协同学习. FATE 采用密钥共享、散列[17] 和同态加密技术, 实现多方安全模式下的机器学习、深度学习和迁移学习. FATE 覆盖了横向联邦学习, 纵向联邦学习, 联邦迁移学习和同步、异步模型融合, 并提供联邦特征工程、模型评估、在线推理等一站式解决方案. 此外, FATE 还提供了诸如数据对齐和模型聚合等其他功能.

FedML 是由美国南加州大学、MIT (Massachusetts Institute of Technology) 和 Stanford 等高校和公司联合发布的联邦学习开源框架, 支持 3 种不同的计算范例, 分别是单机模拟、分布式训练和移动设备训练. FedML 主要由高级接口 FedML-API (federated machine learning application programming interface) 和低级接口 FedML-core 组成, 其中 FedML-API 建立在 FedML-core 之上, 包括模型、数据

集和算法. FedML 支持的算法包括线性模型 (如逻辑回归) 和神经网络 (如 CNN (convolutional neural network) 和 RNN (recurrent neural network)), 根据训练设备的异构性, 采用了当前最流行的消息队列遥测传输协议, 实现云服务器与终端设备之间的通信.

Flower 是一款由英国牛津大学于 2020 年发布的联邦学习框架, 其最大的优点是可以模拟真实场景下的大规模联邦训练, 充分考虑计算资源、内存空间和通信资源, 高效利用移动设备和无线客户端下异构资源[18]. 此外, Flower 具有兼容性和易用性, 分别表现在可以跨平台和跨设计语言, 可以支持已有机器学习框架, 同时提供抽象的框架封装, 这使得用户可以快速高效地搭建联邦学习训练流程, Flower 适用于项目研究, 也方便进行生产部署.

现有的支持端云协同训练框架主要有 FedML 和 Flower, 支持在分布式数据上进行安全和协作的机器学习, 提供了多种联邦学习算法和应用场景以及易于使用的 PyThon 应用程序编程接口. 它们的不同点主要体现在以下两个方面:

(1) FedML 是一个集研究和生产于一体的边缘-云平台, 支持在任何规模和任何地点进行联邦学习, 可以让用户在公有云上部署、监控、改进和共享 AI (artificial intelligence) 模型;

(2) Flower 是一个轻量级的联邦学习框架, 支持在多种设备和环境上进行联邦学习, 它允许用户自定义联邦学习算法和模型, 并与 TensorFlow、PyTorch 等深度学习框架深度兼容.

综上所述, FedML 和 Flower 都是优秀的端云联邦学习平台, FedML 更注重平台化、社区化, Flower 更注重灵活性、兼容性和可扩展性.

1.2 端侧训练框架

随着大数据和物联网技术的快速发展、相关应用的快速普及, 应用数据的爆炸式增长给云侧带来了极大的负载, 同时大量数据集中存储在云侧也可能导致数据安全与隐私问题, 边缘计算提供了一种可能的解决方案, 它能够更接近本地数据和管理本地实体[19]. 边缘计算将任务部分或全部写在边缘计算服务器上, 将计算能力延伸到边缘层[20], 这种方式能够减小数据传输和处理的延迟, 提高计算效率. MPDA (model personalization with large-scale cloud-coordinated domain adaption)[21] 从云侧提取与端侧设备上类似的样本, 在端侧设备上进行训练来提高模型整体性能. 文献 [22] 面向推荐系统设计了多个具有相同功能但训练过程不同的模型, 并使用元控制器来自适应选择相应模型, 优化推荐性能.

随着端侧设备计算能力的显著提高, 越来越多的任务开始在端侧执行, 大多数已有的工作聚焦于处理响应延迟或隐私保护方面. 然而, 端侧设备的计算能力和存储能力有限, 当涉及的数据量非常大时, 通常需要传输到云侧进行处理. 已有研究中, 对端侧设备和云侧采用单独存储计算数据的模型较多, 然而端侧设备和云侧之间的联合建模并使双方共同受益的工作较少[23]. 端云协同训练充分结合了端侧设备和云服务器的优势, 达到提升模型训练的效率和性能的目的.

端云协同训练通过云侧下发统一训练框架供端侧训练, 为了满足端侧训练的需求, FedML 以及 Flower 分别使用 MNN 和 TFlite 作为端侧训练框架. MNN 是由阿里巴巴开发的一款轻量级深度学习推理框架, 旨在为移动设备和嵌入式设备提供高效的神经网络推理能力, 支持多种硬件平台 (包括 Central Processing Unit、Graphics Processing Unit、Neural Processing Unit 等) 和多种操作系统 (包括 Android、iOS 和 Linux 等). MNN 基本工作流程由离线转换和设备推理两部分组成. 对于离线转换部分, 不同深度学习框架模型输入转换器中, 转换器将其转换为 MNN 框架的模型格式, 同时进行了一些基本的图优化. 设备推理部分由 3 个模块组成: 预推理、算子级优化和后端抽象. 对于每个算子, 在预推理模块中, 通过成本评估机制, 结合输入大小和硬件属性等信息, 动态选择最佳的计算解决方案. 算子级优化模块利用诸如 SIMD (single instruction multiple data) (单指令多数据)、流水线等技术进一步提高性能. 此外, MNN 支持多种硬件架构作为后端, 由于不同硬件规格存在差异, 没有统一的

标准, MNN 提供了多种软件解决方案, 如 Open Computing Language、Open Graphics Library、Vulkan 和 Metal 等. 所有的后端都被实现为独立的组件, 并提供了一组统一的接口, 通过后端抽象模块来隐藏原始的细节.

TensorFlow lite 是由 Google 开发的一款轻量级深度学习推理框架. 与 MNN 类似, TFlite 也支持多种硬件平台, 并且支持多种操作系统. TFlite 可以在 TensorFlow 模型转换为 TFlite 格式后, 在本地进行模型推理, 这使得 TFlite 适用于多种应用场景, 包括人脸识别、图像识别、语音识别等. 同时, TFlite 也支持使用自定义运算符, 扩展 TensorFlow 模型的功能, 提供更多的操作支持.

从功能角度来看, MNN 和 TFlite 等端侧训练框架都是为了支持在移动设备和嵌入式设备上进行深度学习模型的推理和训练, 实现端侧智能, 都提供了模型转换、优化、执行、解释等功能以及丰富的算子库和应用程序编程接口, 可以让用户在端侧设备上运行多种深度学习应用; 从性能角度来看, MNN 和 TFlite 等端侧训练框架都是为了提高端侧设备上的模型运行效率和质量, 实现高性能端侧智能, 都支持 CPU、GPU、NPU 等多种硬件加速, 以及量化、剪枝、蒸馏等模型压缩技术, 可以让用户在端侧设备上运行更快、更小、更准的模型. 此外, MNN 和 TFlite 等框架不仅支持端侧训练, 还是实现端云协同、分布式训练、联邦学习等高级功能的关键步骤. 端侧训练框架提供的端侧训练能力使用户可以在端侧设备上进行模型更新, 当搭配联邦学习、协作学习等技术, 可进一步实现协作模型, 无须上传用户隐私数据, 从而有效保障了数据安全和隐私.

2　端云协同训练

本章将介绍基于 FedMNN 框架进行端云协同训练的通用流程, 并对训练过程涉及的模块进行详细阐述. 此外, 以经典推荐算法模型 DSSM 与 Deep and Wide 为例, 介绍端云协同场景下, 实现不同算法模型的一般流程.

2.1　整体流程

基于 FedMNN 的端云协同训练流程分为 3 个部分: 云侧模型准备、端侧模型部署与训练、端云服务.

2.1.1　云侧模型准备

TensorFlow、PyTorch 和 CAFFE (convolutional architecture for fast feature embedding) 等常见的深度学习框架通常被部署在云侧进行模型训练, 不能直接支持端侧设备训练. FedMNN 使用移动神经网络框架 MNN 作为端侧统一训练框架进行端侧训练. 从通用性角度出发, 为了实现端侧训练, 本文将云侧模型转换为端侧可训练的 MNN 模型. 模型转换过程如图 1 所示, 使用开源框架 ONNX (open neural network exchange) 作为中间表示, 将不同深度学习框架实现的模型转换为 ONNX 模型, 并利用 MNN 提供的模型转换工具将 ONNX 模型转换为 MNN 模型. 转换后的 MNN 模型可以直接发送给不同客户端用于端侧训练.

2.1.2　端侧模型部署与训练

端侧模型训练需要完成两个准备阶段: 一是模型准备; 二是数据准备. 在模型准备阶段, 任务初始化时, 云侧将初始化模型直接下发到端侧. 在训练过程中, 当每轮训练结束后, 云侧对端侧上传的模型进行聚合, 并将聚合模型再次下发给端侧. 在数据准备阶段, 需要将端侧数据存放到指定路径. 此外, 基于 MNN 框架进行端侧训练需要针对特定训练数据进行数据集的定义并实现数据读取操作. 特此说明, 在实验验证阶段, 将预处理好的数据上传到手机中. 在实际应用中, 用户数据均来自本地, 并且在训练过程中无须上传至云侧, 在一定程度上保护了用户数据安全与隐私.

总体来说, 端侧训练的流程与云侧训练类似. 端侧模型的训练流程如图 2 所示. 在初始化阶段, 首先, 端侧获取云侧下发的模型并初始化优化器; 然后, 创建 Executor 对象, 给用户提供配置推理后端、

线程数等属性; 最后, 创建数据集以及 DataLoader. 在训练阶段, 先从 DataLoader 中获取训练数据, 并执行模型前向操作, 再根据训练任务的不同来计算损失, 并执行梯度反向传播, 便完成了一个批次的训练.

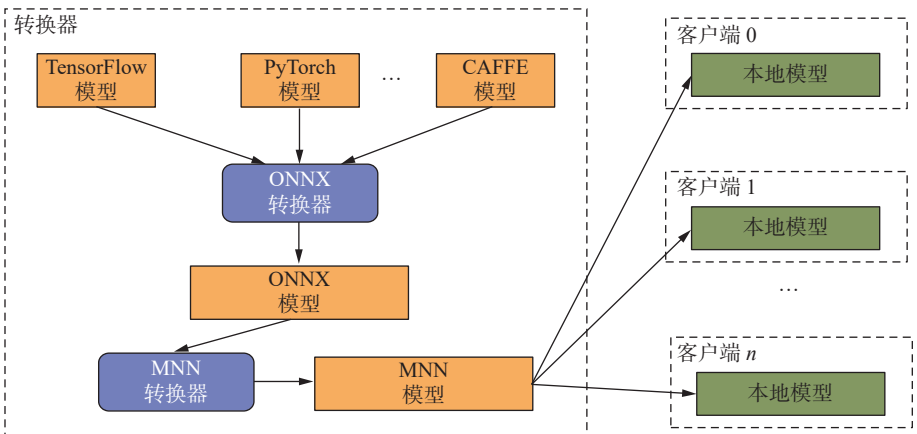


图 1 云侧模型转换

Fig. 1 Cloud-side model transformation on cloud side

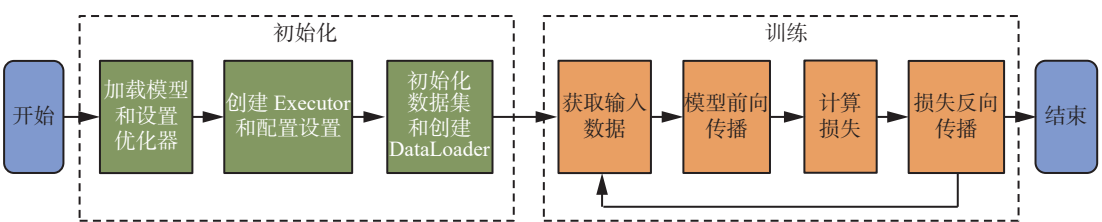


图 2 端侧模型训练流程

Fig. 2 End-side model training process

2.1.3 端云服务

端云服务是实现端云模型协同优化的重要途径, 主要包括 3 个方面: 一是将端侧模型的训练结果反馈给云侧, 进行模型的聚合和更新, 以提高云侧模型的泛化能力和鲁棒性; 二是根据业务需求, 将云侧模型按需部署到不同的设备上, 实现端云协同服务; 三是根据云侧的负载情况对端侧设备进行调度, 选择参与训练的设备.

模型聚合与更新. 端云协同训练的关键之一在于云侧模型的聚合, 将多个端侧模型的参数或梯度进行加权平均或通过其他方式的融合, 得到一个新的云侧模型. 云侧模型聚合服务的整体流程如图 3 所示. FedMNN 使用 FedML 作为云侧联邦学习框架, FedML 提供聚合服务, 可以选择不同的联邦学习聚合算法. 本文选择 FedAvg (federated averaging algorithm) 进行后续实验, 其实现原理是将端侧上传的 MNN 模型读取为 PyTorch Tensor 字典进行聚合, 并将聚合模型下发至端侧对端侧模型进行更新.

端云通信. FedMNN 完成端云通信依赖 FedML 提供的通信链路, 该通信链路基于消息队列遥测传输协议进行消息传输, 在数据传输过程中, 模型均以 MNN 的形式存储. 端云通信链路中的数据以及模型流向如图 3 所示. 训练初始阶段, 由云侧向所有加入训练的端侧设备下发初始化 MNN 模型, 端侧设备基于此模型以及本地数据进行训练. 训练完成后, 所有端侧设备将本地训练得到的 MNN 模型上传至云侧进行聚合, 并等待云侧完成聚合, 将聚合后的模型发送给所有端侧设备.

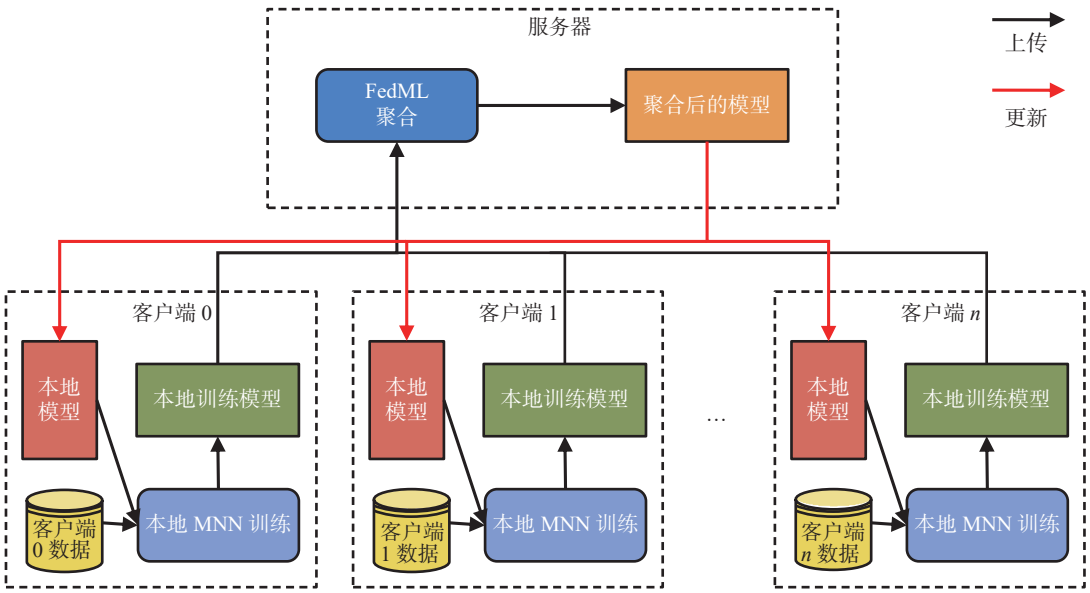


图 3　端云模型聚合与更新

Fig. 3　Cloud-end model aggregation and update

端侧设备调度. 训练由云侧发起, 并在云侧选择好参与训练的端侧设备. 在每轮通信中, 必须保证通信链路的畅通, 当参与训练的任意客户端发生意外断开连接, 或者模型上传、下载的过程出现丢失, 都将导致本次训练无法继续进行, 这也是 FedML 目前的局限性. 此外, 研究发现, 目前 FedMNN 中的端侧设备和云服务器还无法根据网络状况、计算能力、数据特征等动态调整通信频率、更新步长、聚合策略等参数, 实现自适应的端云协同训练. 因此, 在适应更大规模的端云协同训练上仍然有很多工作要做.

2.2　端云协同训练算法实现

为了说明 FedMNN 在端云协同场景下的应用, 本文对 FedMNN 实现端云协同训练的算法进行总结. 与传统的 FedAvg 算法相比, 在 FedMNN 框架下, 需要先完成模型的转换, 将其他模型转换为 MNN 模型进行端云协同训练. 此外, FedMNN 在进行端侧设备调度需要从 MLOps (machine learning operations) 平台中选择参与训练的设备, 并将初始化模型发送给所有选中的设备. 在训练过程中, 需要将端侧上传的模型转换为 Torch Tensor 字典进行聚合, 再将聚合后的参数存储到 MNN 模型. 端云协同训练的算法实现详见算法 1.

算法 1 端云协同训练

输入: 初始化全局模型 m_1, 每个客户端的数据集 $\mathcal{D}_k$

输出: 聚合全局模型 m_T

1: K 个客户端编号为 k, B 是本地批量大小 (batch_size), E 是客户端训练轮数 (epoch), T 是通信的轮数, η 是学习率

2: 服务器执行

3: 将 torch 模型转换为 MNN 格式模型 m_1

4: **For** 每轮 $t = 1, 2, \cdots, T-1$ **do**

5:　　$S_t \leftarrow$ 选择参与训练的 K 个客户端 // S_t 是第 t 轮参与训练的客户端集合

6:　　**For** 每个客户端 $k \in S_t$ **do** //K 个客户端并行处理

发送模型 m_t 到客户端 k
  $m_{t+1}^k \leftarrow$ 调用客户端更新 (k, m_t) //得到客户端 k 上传的新模型 m_{t+1}^k
  $w_{t+1}^k \leftarrow m_{t+1}^k$ //从模型 m_{t+1}^k 中提取模型的参数 w_{t+1}^k
$w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$ //模型聚合, n_k 为客户端 k 上的样本数量, n 为所有被选中客户端的总样本数量
$m_{t+1} \leftarrow w_{t+1}$ //将聚合后的模型参数 w_{t+1} 存储到 MNN 模型中, 得到 m_{t+1}
客户端更新 (k, m_t) //在客户端 k 上运行
将 $\mathcal{D}_k$ 分成 B 个大小为 B 的批次
从模型 m_t 中提取本地模型的参数 w_t^k
  For 本地第 i 轮, $i \in (1, E)$ **do**
    For 每个批次 $b \in (1, B)$ **do**
      $w_{t+1}^k \leftarrow w_t^k \leftarrow \eta \nabla l(w_t^k; b)$ //∇ 为计算梯度, $l(\cdot)$ 为损失函数
保存 w_{t+1}^k 为本地 MNN 格式模型 m_{t+1}^k
返回 m_{t+1}^k 至服务器

3 实　　验

为了验证 FedMNN 在端云协同场景下端侧设备的功耗以及模型性能, 本文使用安卓手机进行了框架功耗测试. 此外, 选取了推荐系统中的经典算法 DSSM 与 Deep and Wide 来验证端云协同训练模型在推荐场景中的性能.

3.1 框架功耗测试

为了对比 FedMNN 与 FLTFlite 两种端云协同训练框架在真实端侧设备运行的功耗情况, 本文选择了相同的模型以及联邦学习常用的基准数据集 MNIST (https://yann.lecun.com/exdb/mnist) 进行了训练功耗测试. MNIST 是一个手写体数字的图片数据集, 该数据集来自美国国家标准与技术研究所. 数据集由来自 250 个不同的人手写的数字构成, 其中 50% 是高中学生, 50% 来自人口普查局的工作人员. 训练集一共包含了 60 000 张图像, 而测试集一共包含了 10 000 张图像. 实验中, Flower 和 FedML 云侧服务均部署在 PC (personal computer) 设备上, 实验过程均使用 FedAvg 算法, 端云通信轮数均设置为 20. 使用相同的 Android 手机作为端侧测试设备, 并保证训练时间有间隔, 避免连续实验对手机原始性能产生影响. 在训练过程中, 使用 PowerMoniter 功耗仪进行测试, 运用 PowerTool 工具进行数据分析.

表 1 展示了 FLTFlite 与 FedMNN 在端侧设备上的功耗情况. 首先, 直接统计训练完成一轮端云通信的端侧平均功耗, 总体来说, FedMNN 的平均每轮通信功耗比 FLTFlite 低 0 ∼ 20%. 为了更严谨统计端侧训练过程带来的实际功耗, 本文统计了端云协同训练相对不训练产生的平均功耗增量, 具体操作是统计使用 APP (application) 在不进行训练时的平均功耗, 并用每轮通信的实际功耗减去平均功耗的差再乘以平均每轮训练时间. 通过表 1 可以发现, 使用 FedMNN 相较于 FLTFlite 进行端侧训练时, 每轮通信的平均功耗低 32% ∼ 51%. 此外, 对比了端侧本地训练 epoch 数量以及 batch_size 的差异对训练功耗的影响, 实验结果表明, 增大端侧本地训练 epoch 会显著提升整体训练时长以及每轮通信的功耗. 增大 batch_size 后, 会降低端侧训练的时长, 同时导致训练功耗上涨, FLTFlite 训练流程对端侧训练功耗增长不超过 2%, 而 FedMNN 对 batch_size 的变化则比较敏感, 功耗增长接近 13%. 实验证明了整体使用 FedML + MNN 端云协同训练在端侧实际功耗优于 Flower + TFlite. 在深度学习中, 通常使用 epoch 来表示将整个数据集迭代一遍的过程.

表 1　Flower + TFlite 与 FedML + MNN 训练流程端侧功耗对比

Tab. 1　End-side power consumption comparison between Flower + TFlite and FedML + MNN training processes

训练框架	epoch	batch_size	收敛时间/min	每轮功耗/mA · h	每轮功耗增量/mA · h
FLTFlitelocal	3	100	17.82	8.858 9	4.318 1
FLTFlitelocal	1	100	8.99	4.529 2	2.264 9
FLTFlitelocal	1	1 000	8.04	4.309 6	2.290 8
FedMNNlocal	3	100	18.08	8.658 4	2.946 5
FedMNNlocal	1	100	9.63	4.151 1	1.128 3
FedMNNlocal	1	1 000	8.64	3.992 2	1.283 2

3.2　DSSM 与 Deep and Wide 模型端云协同训练

现有的 FedMNN 通常应用在图像相关任务. 为了验证基于 FedMNN 的端云协同训练的有效性, 以 DSSM 以及 Deep and Wide 两种推荐算法在联邦学习的设置下进行了实验. 云侧服务均部署在 PC 设备上, 实验过程使用 FedAvg 算法, 端云通信轮数均设置为 5, 端侧本地训练 epoch 均设置为 2. 实验结果均使用最后一次聚合的模型进行评估, 使用 AUC (area under the curve) 作为评价指标.

Movielens1M (https://grouplens.org/datasets/movielens) 数据集数据划分按照时间以晚于 2003 年 1 月 1 日的数据为测试集, 早于 2003 年 12 月 31 日的数据集为训练集. 同时为确保测试集中的用户在训练集中有历史行为, 需要过滤掉在训练集中没有行为的用户. 对训练数据进行随机负采样, 将用户评分的电影作为正样本, 在所有的电影中随机采样若干个电影作为用户的负样本, 默认正负样本比例为 1∶3. 随后在采样后的数据中, 随机抽取互相独立的 50 万条数据作为不同端侧设备的训练数据.

Avazu (https://www.kaggle.com/c/avazu-ctr-prediction/data) 数据集预处理参考 torch-rechub (https://github.com/morningsky/Torch-RecHub) 中关于 Avazu 数据集的预处理. 由于训练集数量过大, 从训练集中随机抽取互相独立的 30 万条数据作为不同端侧设备的训练数据. 从表 2 和表 3 的结果可以看出, 将端云协同训练用于推荐领域, 在端侧设备持有不同数据的情况下, 增加多台端侧设备进行端云协同训练, 可以提高模型收敛速度, 在更少的通信轮数下实现相对更高的性能.

表 2　在 Movielens1M 上 DSSM 每轮的 AUC

Tab. 2　DSSM AUC on Movielens1M for each round

客户端数量	通信轮数	AUC
1	1	70.47
1	2	72.63
1	3	74.18
1	4	74.85
1	5	75.18
3	1	71.10
3	2	73.31
3	3	74.82
3	4	75.57
3	5	75.86

4　端云协同训练原型系统

4.1　系统架构

端云协同训练原型系统旨在通过分布式计算和数据处理, 在多个设备上实现模型训练. 该系统由

两个主要部分组成: 端侧设备和云服务器, 其架构如图 4 所示. 云服务器提供模型的聚合与更新服务, 端侧设备提供数据存储与模型训练功能.

表 3 在 Avazu 上 Deep and Wide 每轮的 AUC

Tab. 3 Deep and Wide AUC on Avazu each round

客户端数量	通信轮数	AUC
1	1	53.45
1	2	57.08
1	3	59.27
1	4	60.68
1	5	61.67
3	1	53.97
3	2	57.70
3	3	59.85
3	4	61.19
3	5	62.11

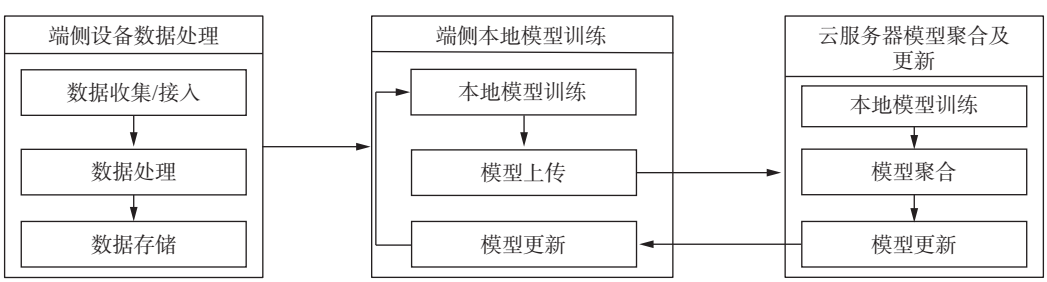


图 4 端云协同训练原型系统架构

Fig. 4 Architecture of cloud-end collaborative training prototype system

端侧设备: 端侧设备可以是智能手机、平板电脑等, 其客户端负责数据的搜集、处理、存储和计算, 同时负责更新本地模型.

云服务器: 云服务器负责协调端侧设备的计算任务, 执行全局模型训练, 存储全局模型权重, 并将更新后的模型发送至端侧设备.

4.2 系统功能及演示

端云协同训练原型系统功能主要分为端侧设备数据处理、端侧本地模型训练和云服务器模型聚合及更新.

4.2.1 端侧设备数据处理

端侧设备负责收集用户的数据, 并对数据进行预处理后存储在设备中. 系统提供数据接入的功能, 如图 5 所示, 选定项目 (以 ai.fedml 为例) 后, 右击并选择 Upload 按钮, 即可完成数据接入.

完成数据接入后, 可在端侧设备上选择并设置数据读取、处理的路径. 如图 6 所示, 点击图 6(a) 顶部绿色一栏, 会出现图 6(b) 所示界面. 继续点击图 6(b) 中右上角绿色按钮 (Set Private Path), 设备数据存储的路径 (/storage/emulated/0/ai.fedml/movielens1m/client_0), 即可完成用于本地隐私数据读取、处理的数据路径设置.

4.2.2 端侧本地模型训练

端侧设备在本地数据预处理后的基础上利用 MNN 框架进行本地模型训练, 训练完成后更新本地模型权重. 端侧设备依赖 FedML 提供的通信链路, 其基于消息队列遥测传输协议进行消息传输, 在数据传输过程中, 模型均以 MNN 的形式存储. 端侧设备将训练好的模型框架上传到云侧.

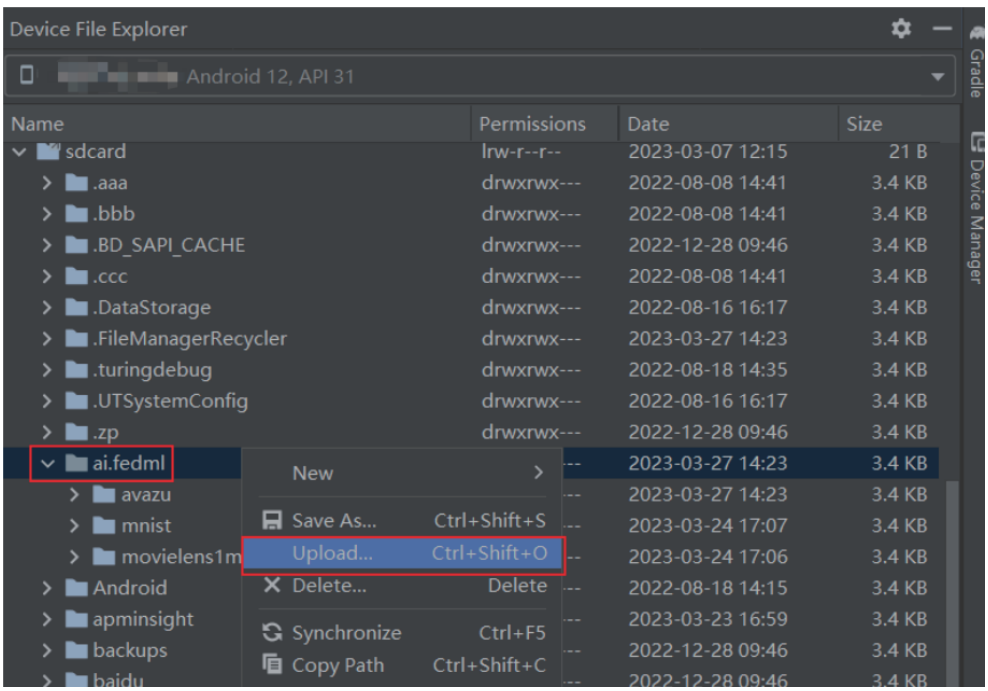
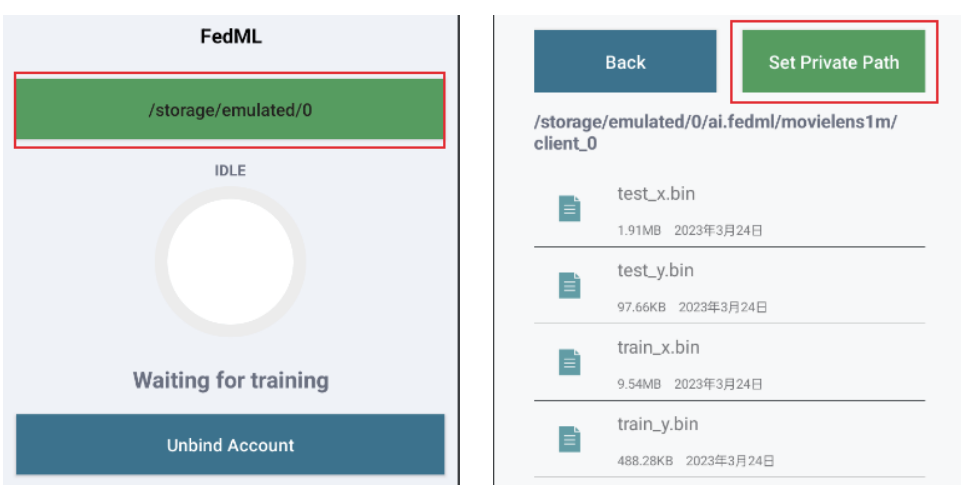


图 5　端侧设备数据接入功能界面

Fig. 5　End-side device data access interface



(a) 路径选择　　(b) 路径设置

图 6　端侧设备数据路径设置

Fig. 6　End-side device data path setting

如图 7 所示, 进入 MLOps 平台后, 按照 create group→create project→create run→select application→start run 的顺序开启训练. 开启训练后, 端侧设备进行本地模型训练的迭代, 并在端侧设备上显示如图 8 所示的训练进度及评价指标.

4.2.3　云服务器模型聚合及更新

上传后的模型通过联邦学习框架 FedML 提供聚合服务, 使用 FedAvg 算法对端侧设备上传的模型进行聚合操作 (图 9), 聚合后的模型以更新全局权重. 更新后的模型下发到端侧设备, 以便其在下一轮训练中使用.

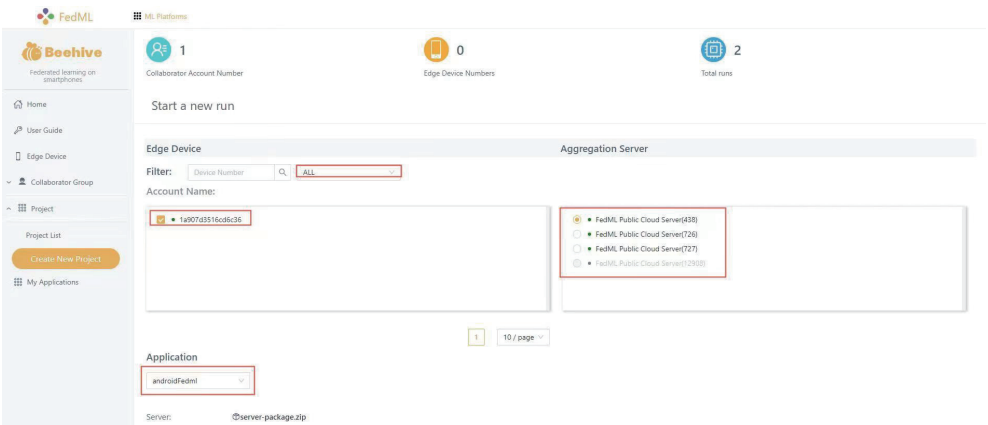


图 7 开启端云协同训练界面

Fig. 7 Cloud-end collaborative training start-up interface

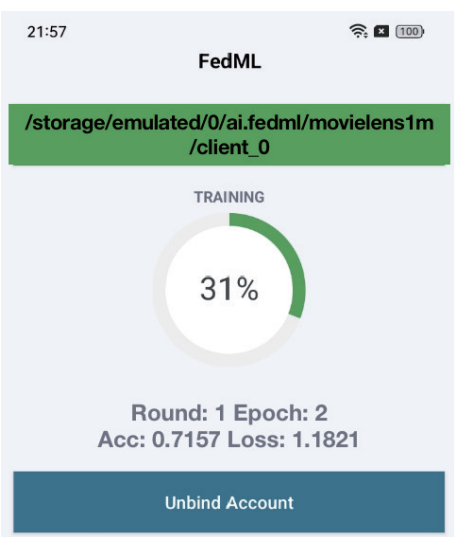


图 8 端侧设备训练状态

Fig. 8 End-side device training status

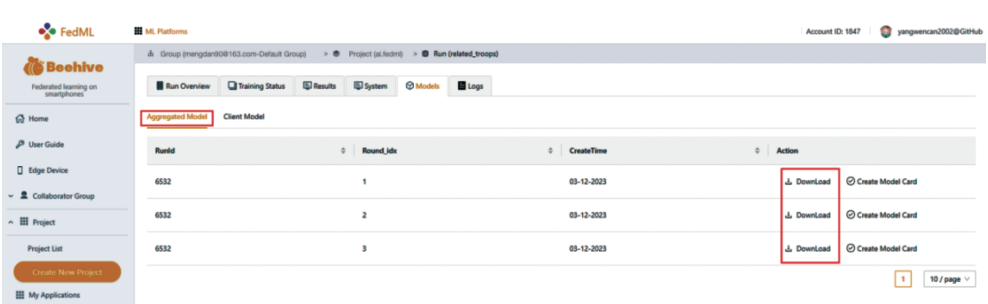


图 9 云服务器模型聚合

Fig. 9 Cloud sever model aggregation

5 结 论

本文以端云协同训练 FedMNN 框架为例, 详细介绍了端云协同训练的流程以及算法的实现步骤.

同时, 通过对比 FedMNN 与 FLTFlite 在基准任务上不同参数设置下的训练功耗, 在同等情况下, FedMNN 在每轮通信过程中的平均功耗较 FLTFlite 低 32% ∼ 51%. 此外, 通过在 DSSM 与 Deep and Wide 模型上进行的实验也进一步验证了 FedMNN 的优势. 在这两种场景下, FedMNN 能够利用端侧设备上的用户数据进行模型训练, 实现个性化的服务. 这种方法可以保护用户数据安全与隐私, 避免数据泄露和潜在攻击的风险.

[参　考　文　献]

[1] HAFFAR R, SANCHEZ D, DOMINGO-FERRER J. Explaining predictions and attacks in federated learning via random forests [J]. Applied Intelligence, 2023, 53(1): 169-185.

[2] ZHAO B, MOPURI K R, BILEN H. IDGL: Improved deep leakage from gradients [EB/OL]. (2020-01-08)[2023-05-18]. https://arxiv.org/pdf/2001.02610.pdf.

[3] WEI K, LI J, DING M, et al. Federated learning with differential privacy: Algorithms and performance analysis [J]. IEEE Transactions on Information Forensics and Security, 2020, 15: 3454-3469.

[4] FANG H K, QIAN Q. Privacy preserving machine learning with homomorphic encryption and federated learning [J]. Future Internet, 2021, 13(4): 94.

[5] LI T, SAHU A K, TALWALKAR A, et al. Federated learning: Challenges, methods, and future directions [J]. IEEE Signal Processing Magazine, 2020, 37(3): 50-60.

[6] KAIROUZ P, MCMAHAN H B, AVENT B, et al. Advances and open problems in federated learning [J]. Foundations and Trends® in Machine Learning, 2021, 14(1/2): 1-210.

[7] KARIMIREDDY S P, KALE S, MOHRI M, et al. SCAFFOLD: Stochastic controlled averaging for federated learning [C]// Proceedings of the 37th International Conference on Machine Learning. 2020: 5132-5143.

[8] HE C Y, LI S Z, SO J, et al. FedML: A research library and benchmark for federated machine learning [EB/OL]. (2020-11-08)[2023-05-18]. https://arxiv.org/pdf/2007.13518.pdf.

[9] MATHUR A, BEUTEL D J, DE GUSMÃO P P B, et al. On-device federated learning with flower [EB/OL]. (2021-04-07)[2023-05-18]. https://arxiv.org/pdf/2104.03042.pdf.

[10] JIANG X T, WANG H, CHEN Y L, et al. MNN: A universal and efficient inference engine[EB/OL]. (2020-02-27)[2023-06-01]. https://arxiv.org/pdf/2002.12418.pdf.

[11] ZHOU P, WANG K H, GUO L K, et al. A privacy-preserving distributed contextual federated online learning framework with big data support in social recommender systems [J]. IEEE Transactions on Knowledge and Data Engineering, 2019, 33(3): 824-838.

[12] YANG Q, LIU Y, CHEN T J, et al. Federated machine learning: Concept and applications [J]. ACM Transactions on Intelligent Systems and Technology, 2019, 10(2): 12.

[13] CHEN Q, YAO L, WU Y L, et al. PyHENet: A generic framework for privacy-preserving DL inference based on fully homomorphic encryption [C]// Proceeding of the 4th International Conference on Data Intelligence and Security. 2022: 127-133.

[14] LIU Y, FAN T, CHEN T J, et al. FATE: An industrial grade platform for collaborative learning with data protection [J]. Journal of Machine Learning Research, 2021, 22: 226.

[15] 马艳军, 于佃海, 吴甜, 等. 飞桨: 源于产业实践的开源深度学习平台 [J]. 数据与计算发展前沿, 2019, 1(1): 105-115.

[16] MCMAHAN B, MOORE E, RAMAGE D, et al. Communication-efficient learning of deep networks from decentralized data [C]// Proceedings of the 20th International Conference on Artificial intelligence and statistics. 2017: 1273-1282.

[17] LI Q B, WEN Z Y, HE B S. Practical federated gradient boosting decision trees [C]// Proceedings of the AAAI Conference on Artificial Intelligence. 2020: 4642-4649.

[18] 林伟伟, 石方, 曾岚, 等. 联邦学习开源框架综述 [J]. 计算机研究与发展, 2023, 60(7): 1551-1580.

[19] WU D P, SUN M Y, ZHANG P, et al. Personalized secure demand-oriented data service toward edge-cloud collaborative IoT [J]. IEEE Internet of Things Journal, 2022, 10(1): 378-390.

[20] WANG T, MEI Y X, JIA W J, et al. Edge-based differential privacy computing for sensor-cloud systems [J]. Journal of Parallel and Distributed Computing, 2020, 136: 75-85.

[21] YAN Y K, NIU C Y, GU R J, et al. On-device learning for model personalization with large-scale cloud-coordinated domain adaption [C]// Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2022: 2180-2190.

[22] YAO J C, WANG F, DING X C, et al. Device-cloud collaborative recommendation via meta controller [C]// Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2022: 4353-4362.

[23] YAO J C, WANG F, JIA K Y, et al. Device-cloud collaborative learning for recommendation [C]// Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2021: 3865-3874.

(责任编辑: 陈丽贞)