小 型 微 型 计 算 机 系 统
Journal of Chinese Computer Systems
DOI:10.20009/j.cnki.21-1106/TP.2023-0404
2025 年 1 月 第 1 期
Vol.46 No.1 2025

工业物联网中的个性化联邦学习算法的研究

刘 洋[1],吴 旭[1,2],刘承坤[1]

[1](广西大学 计算机与电子信息学院,南宁 530004)
[2](海南师范大学 信息科学技术学院,海口 571158)
E-mail:ly9811@foxmail.com

摘 要: 为了在不直接共享原始数据的前提下构建联合模型,联邦学习应运而生.然而,在复杂的工业物联网环境中,联邦学习的应用面临两大挑战:1)工业物联网设备之间彼此异构,掉队和离线设备的存在极大拖慢了联邦学习的训练速;2)不同数据所有者拥有的数据彼此异构,客户端的本地模型差异较大,简单对本地模型进行平均无法获得适用于所有客户端的高质量模型.为了解决上述挑战,本文设计了一个融合数字孪生的联邦学习架构,实现对设备资源的高效调度.此外,本文提出了一个基于参数解耦和聚类的个性化联邦学习算法,既能满足用户的个性化需求,又能实现同构客户端的深度协作.实验结果验证了提出的个性化联邦学习算法的有效性.

关 键 词: 联邦学习;工业物联网;数字孪生;个性化
中图分类号: TP393 **文献标识码:** A **文 章 编 号:** 1000-1220(2025)01-0209-08

Research on Personalized Federated Learning Algorithm in Industrial Internet of Things

LIU Yang[1],WU Xu[1,2],LIU Chengkun[1]

[1](College of Computer and Electronic Information,Guangxi University,Nanning 530004,China)
[2](School of Information Science and Technology,Hainan Normal University,Haikou 571158,China)

Abstract: In order to build a joint model without directly sharing the original data,federated learning came into being. However,in the complex industrial IoT environment,the application of federated learning faces two major challenges:1) Industrial IoT devices are heterogeneous with each other,and the existence of outdated and offline devices greatly slows down the training speed of federated learning;2) The data owned by different data owners are heterogeneous with each other,and the local models of the clients are quite different. Simply averaging the local models cannot obtain a high-quality model suitable for all clients. In order to solve the above challenges,this paper designs a federated learning architecture that integrates digital twins to achieve efficient scheduling of device resources. In addition,this paper proposes a personalized federated learning algorithm based on parameter decoupling and clustering,which can not only meet the individual needs of users,but also realize the deep collaboration of isomorphic clients. Experimental results verify the effectiveness of the proposed personalized federated learning algorithm.

Keywords: federated learning;industrial internet of things;digital twin;personalization

0 引 言

工业物联网贯穿了工业生产制造的全过程,通过监控生产设备状态、能源消耗、物流供应链等一系列数据,为工业数字化的实现奠定了基础[1,2].随着近几年工业物联网技术在制造业中的不断应用,连接到工业物联网中的设备的数量日益增长,产生的数据呈指数增长[3].联邦学习的出现为工业物联网中数据的安全共享和利用提供了一个可行的解决方案.作为一种分布式的机器学习框架,联邦学习可以在数据共享和隐私保护之间实现一个良好的平衡[4].通过联邦学习,多个用户可以协作训练一个全局模型.用户之间的信息交换通过不同用户的本地模型的全局聚合实现.在联邦学习的整个训练过程中,每个用户的数据始终保留在本地,这极大的保护了用户的数据隐私.

然而,异构的工业物联网环境为联邦学习的部署和应用带来了极大的挑战,严重的影响了联邦学习的性能.工业物联网的异构性主要包含两个方面:设备的异构性和数据的异构性.

1)设备的异构性是指不同的工业物联网设备有不同的计算、通信和存储能力.一些计算资源有限的设备由于需要花费更多的时间来进行本地模型的训练,可能会成为掉队者.此外,由于能源限制或者故障,一些工业物联网设备可能会中途退出联邦学习.掉队或者中途退出的客户端可能会严重降低联邦学习全局模型的收敛速度和性能[5].对于设备异构性问题,Xie 等人[7]提出异步联邦优化算法 FedAsync,参数服务器对每个来自客户端的本地更新根据其陈旧性进行加权聚合.

收稿日期:2023-08-22 收修改稿日期:2023-09-18 基金项目:国家自然科学基金项目(62462029,62062006)资助;广西自然科学基金项目(2018JJA170028,2019JJA170045)资助;国家重点研发计划课题项目(2018YFB1404404)资助. 作者简介:刘 洋,男,1998 年生,硕士研究生,CCF 会员,研究方向为联邦学习、信任管理、工业物联网;吴 旭(通信作者),女,1978 年生,博士,博士后,副教授,CCF 高级会员,研究方向为信息安全、边缘计算、软件工程、区块链;刘承坤,男,1999 年生,硕士研究生,研究方向为物联网、区块链、信息安全.

Wu 等人[5]提出利用边缘计算来缓解设备计算资源不足的问题. Sun 等人[8]使用聚类对客户端分组,自适应的调整不同组客户端的聚合频率. Lu 等人[9]提出一个数字孪生边缘网络,通过动态分配通信信道来提高通信效率. Zhang 等人[10]使用小型基站的计算资源来辅助工业物联网设备完成本地训练. 然而,异步模型更新[7]会损害模型性能,边缘计算[5]可能有隐私泄露的风险,文献[8-10]无法同时解决计算和通信资源的限制.

2)数据异构性是指属于不同用户的工业物联网设备采集到的数据是非独立同分布的. 本文主要考虑两种常见的数据异构:1)标签分布偏差,即不同设备的本地数据集的标签分布比例不同;2)数量偏差,即不同的客户端拥有的数据的数量不同. 在这种非独立同分布数据下执行联邦学习,会发生客户端漂移现象[6],不同客户端的本地更新会朝着各自的本地最优的方向更新,从而导致聚合后的全局模型偏离全局最优. 为了解决数据异质性问题,近年来许多个性化联邦学习算法被提出,一些工作通过训练一个具有强大泛化能力的全局模型来缓解客户端漂移,例如文献[11]将数据类别均衡的公共数据集分发给各个客户端以减少数据异构性,文献[12]选择具有最小类别不平衡的客户端参与联邦学习. 此外,迁移学习[13]、元学习[14]和正则化[15]的方法也被广泛采用,但上述方法只学习一个全局模型,无法满足所有客户端的个性化需求. 还有一些工作通过学习多个模型来解决设备异构问题,例如:FedPer[16]通过对模型进行分解,仅共享部分参数实现个性化. 文献[17]利用层次聚类识别客户端的相似性,然而并不是所有客户端都会找到合适的分组.

为了同时解决工业物联网的设备和数据异构性对联邦学习的影响带来的问题,本文提出了一个融合数字孪生的 3 层联邦学习架构. 通过工业物联网设备和虚拟数字孪生模型之间的实时交互,实现对计算资源的高效利用. 此外,本文设计了一个基于参数解耦和聚类的个性化联邦学习算法,以解决数据异构性带来的负面影响. 算法将模型分为基础层和个性化层,使用层次聚类算法根据客户端的相似性对用户进行分组. 在整个训练过程中,基础层的参数由所有客户端进行聚合更新,个性化层的参数由同组客户端协作训练. 实现了在数据异构环境下个性化和用户协作的平衡.

本文的主要贡献如下:

1)提出了一个融合数字孪生的 3 层联邦学习架构,使用数字孪生技术实现了为设备构建数字孪生模型,对工业物联网设备的实时可用资源进行监控,动态调度所有可用计算资源完成训练,大大提高联邦学习的稳定性. 此外,本文提出的 3 层架构可以有效降低云服务器的通信负载.

2)在数字孪生联邦学习架构的基础上,提出一个基于参数解耦和聚类的个性化联邦学习算法,实现了个性化和用户协作的平衡. 模型被分解为基础层和个性化层. 基础层的参数由所有客户端训练,而个性化层的参数由同组用户协作训练,有效提高算法在工业物联网数据异构环境下的鲁棒性,为每个参与者提供个性化的模型.

3)本文实验分析了不同的超参数以及不同的数据异构水平对算法性能的影响. 与主流的个性化联邦学习算法的比对实验表明,本文提出的算法在数据高度异构的环境下表现出更好的有效性和稳定性.

1 系统模型

1.1 架构设计

如图 1 所示,为了解决将联邦学习应用于工业物联网环境所面临的设备异构问题,本节设计了一个融合数字孪生的联邦学习架构. 架构主要分为 3 层,分别为:设备层、本地服务器层和云服务器层.

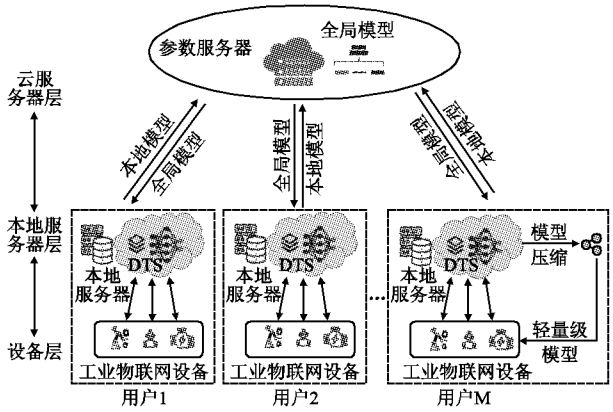


图 1 系统架构

Fig. 1 System architecture

设备层包含大量的工业物联网设备,例如,智能传感器,机器人,执行器等. 这些设备负责收集工业生产中产生的大量数据,接收用户的指令,完成生产任务. 它们会将采集到的所有数据传输到用户的本地服务器,并利用可用的计算资源辅助本地服务器完成计算任务. 它们可能有不同的计算能力,但本文假设所有的工业物联网设备都有足够的通信能力将收集到的数据发送到用户的本地服务器.

本地服务器位于用户本地,由具有充足计算资源和通信资源的设备组成. 每个工业用户都对自己的本地服务器有完全的控制权限. 本地服务器接收工业物联网设备采集到的数据以及工业物联网设备当前可用计算资源的实时状态信息. 用户可以使用数字孪生建模技术[18]基于接受到的数据在本地服务器为每个工业物联网设备构建数字孪生模型,实时获取工业物联网设备的当前状态(例如可用的计算资源). 属于相同用户的所有工业物联网设备的数字孪生模型在本地服务器融合为一个完整的数字孪生系统(Digital Twin System, DTS). 基于构建的数字孪生系统,工业物联网设备的可用计算资源可以被实时监控. 在联邦学习的每个轮次,用户可以调度本地服务器和工业物联网设备的可用计算资源并行协作完成本地模型的更新.

云服务器层由可信第三方提供,由具有充足计算资源的云服务器集群组成. 在联邦学习中,云服务器负责初始化全局模型参数并发送给本地服务器,与本地服务器通信,接收更新后的本地模型参数,聚合得到新的全局模型. 云服务器主要负责聚合模型参数,因此也被称为参数服务器.

此外,针对具有离线工作需求的工业物联网设备,用户可以使用模型压缩技术例如知识蒸馏[20]对本地服务器上的本地模型进行压缩,获得可以运行在工业物联网设备上的轻量

级模型,从而满足异构工业物联网设备的不同需求.

1.2　数字孪生驱动的计算资源调度

在本文提出的架构中,每个用户的工业物联网设备都会将采集到的数据发送到本地服务器. 本地服务器可以使用数字孪生建模技术为每个设备构建一个可视化的虚拟模型. 文献[18]将数字孪生模型分为多个维度,包含几何模型、物理模型、行为模型、规则模型. 几何模型主要描述设备的形状和内部结构. 物理模型为基于集合模型的物理性质分析和预测提供了基础. 行为模型反映了设备周期性和随机性的行为. 规则模型揭示了物理实体的演变趋势和工作模式,可用于设备的故障检测和预测性维护. 用户可以基于自己的需求构建不同维度的数字孪生模型. 多维数字孪生模型的融合构成了一个完整的数字孪生,可以通过与设备之间的连接监控设备的实时状态,辅助用户制定决策. 具体的模型构建技术在文献[18]有详细介绍.

本文假设工业物联网中一共有 M 位用户,每个用户都拥有大量的设备,用户 i 拥有的设备可以表示为 $D_i=\{d_1^i, d_2^i,\dots,d_{|D_i|}^i\}$,其中 $|D_i|$ 代表用户 i 拥有的工业物联网设备的总数. 同一用户的所有设备均连接到同一个本地服务器,对于设备 d_j,其相对应的数字孪生 DT_j 通过数字孪生建模技术在本地服务器生成. 在时刻 t,设备 d_j 的数字孪生可以表示为:

$$DT_j(t)=\{Model_j(t),DATA_j(t),Comp_j(t)\} \tag{1}$$

其中 $Model_j(t)$,$DATA_j(t)$,$Comp_j(t)$ 分别表示设备 d_j 在 t 时刻的数字孪生模型,采集的历史数据和可用的计算资源. 在本地服务器,由不同设备采集的数据经过处理融合为用户的本地数据集,并随着时间不断扩充.

在联邦学习过程中,每个工业物联网设备的数字孪生模型都可以映射物理设备的实时状态信息. 用户可以通过数字孪生模型实时监控工业物联网设备的可用计算资源. 在本地模型的本地训练时,本地服务器会根据可用计算资源将本地模型训练任务分发给本地服务器以及所有的工业物联网设备协作并行完成训练. 基于负载均衡的原则,将训练任务表示为 $Task$,那么,分配给设备 i 的计算任务可以表示为:

$$Task_i=\frac{Comp_i(t)}{\sum_{i=1}^{M}Comp_i(t)}Task \tag{2}$$

在深度学习的模型训练中,训练所需的计算资源是与训练数据量成正比的. 因此,本文采用本地训练数据集包含的数据规模作为训练任务 $Task$ 的量化指标,根据各个工业物联网设备拥有的计算资源对训练数据进行重新分发,例如,一个设备的可用计算资源占总量 10%,则本地服务器将训练数据的 10% 分配给该设备进行训练.

也就是说,对于每个用户的本地模型训练任务都是由多个设备并行协作完成的,所有的可用计算资源可以被充分调动. 常用的深度学习框架如 Pytorch 和 Tensorflow 均支持多机并行训练神经网络模型,本文的方案有切实的可行性. 此外,本地服务器的稳定性远高于工业物联网设备,且企业用户所拥有的所有工业物联网设备同时故障的概率是极小的. 相比于传统方案直接使用工业物联网设备作为联邦学习的客户端,本方案的可靠性大大增加,即使部分设备发生掉线或故障,其余未故障设备也可以保证顺利完成本地训练. 此外,在提出的三层架构中,云服务器只需与本地服务器进行通信,这极大的提高了联邦学习的可拓展性,缓解了云服务器的通信压力.

2　基于参数解耦和聚类的个性化联邦学习

在工业物联网环境下,不同用户的本地数据是异构的,单一全局模型无法满足所有用户的数据分布. 为了实现个性化联邦学习,本文设计了一个新颖的基于参数解耦和聚类的个性化联邦学习算法.

在联邦学习中,每个客户端都会训练一个本地模型,$\boldsymbol{\theta}_k$ 代表客户端 k 的本地模型,该客户端的目标函数可以表示为:

$$F_k(\boldsymbol{\theta}_k)=\frac{1}{|DS_k|}\sum_{i\in DS_k}l(x_i,y_i,\boldsymbol{\theta}_k) \tag{3}$$

其中 $l(\cdot)$ 是在训练样本 (x_i,y_i) 上的损失函数. x_i 是训练数据的特征,y_i 是标签,DS_k 是客户端 k 的本地训练数据集,$|DS_k|$ 是客户端 k 的本地训练数据集的样本数量. 每个客户端的目标都是最小化本地损失函数,可以表示为:

$$minF_k(\boldsymbol{\theta}_k) \tag{4}$$

客户端使用小批量随机梯度下降最小化损失函数,模型参数的更新公式为:

$$\boldsymbol{\theta}_k^{t+1}=\boldsymbol{\theta}_k^t-\eta\nabla F_k(\boldsymbol{\theta}_k^t,\xi_k^t) \tag{5}$$

其中 $\eta>0$ 是学习率,$\nabla F_k(\boldsymbol{\theta}_k^t,\xi_k^t)$ 是第 t 轮损失函数在随机抽取的本地训练数据子集 ξ_k^t 上的梯度.

在联邦平均算法 FedAvg 中,本地训练完成后,客户端将参数发送到本地服务器聚合,全局模型的更新公式可以表示为:

$$\boldsymbol{\theta}^t=\frac{1}{\sum_{i=1}^{M}|DS_i|}\sum_{j=1}^{M}|DS_j|\cdot\boldsymbol{\theta}_j^t \tag{6}$$

其中 $|DS_i|$ 是客户端 i 的本地数据集的大小.

由于不同客户端的本地数据集在标签类别分布和大小上互不相同,不同客户端的训练目标不同,直接对所有客户端的本地模型参数进行平均可能会导致聚合模型偏离全局最优. 考虑到上述情况,本文提出了一个基于参数解耦和聚类的个性化联邦学习算法.

整个个性化联邦学习算法分为预训练阶段和分组训练阶段. 预训练阶段模型参数被分解为基础层和个性化层,个性化层的参数由客户端单独训练,不参与全局聚合. 预训练阶段结束后,客户端将给个性化层参数上传到参数服务器. 由于预训练阶段个性化层始终在本地训练,个性化层的参数分布很大程度上反映了用户的训练数据的分布. 在分组训练阶段,基于预训练阶段得到的个性化层的参数,参数服务器采用层次聚类算法识别不同客户端之间的相似性,将数据分布相近的客户端分为一组. 接下来,同组的同构客户端协作训练个性化层的参数,在保证客户端协作的同时降低异构数据分布带来的负面影响. 基础层始终由所有客户端协作训练,保证模型的泛化能力. 算法的两个阶段的具体实现细节如下所示.

2.1　基于参数解耦的预训练阶段

由于工业物联网环境下的数据异质性,联邦平均算法获得的单一全局模型无法在所有的客户端均表现良好,这要求算法为每个用户提供个性化模型来满足不同用户的个性化需

求. 在联邦学习中,神经网络的浅层主要用于提取一些通用特征,而神经网络接近输出的深层则包含更多的个性化信息. 因此,如图 2 所示,本文首先将神经网络参数解耦,神经网络接近输入的层被划分为基础层 $\boldsymbol{\theta}^F$,神经网络靠近输出的层被划分为个性化层 $\boldsymbol{\theta}^P$. 在预训练阶段,每个客户端使用随机梯度下降更新本地模型的参数,将基础层的参数发送至参数服务器,而每个客户端本地模型的个性化层的参数保留在本地,不参与全局聚合,参数服务器通过公式(7)聚合收到的基础层参数,获得新的全局基础层参数.

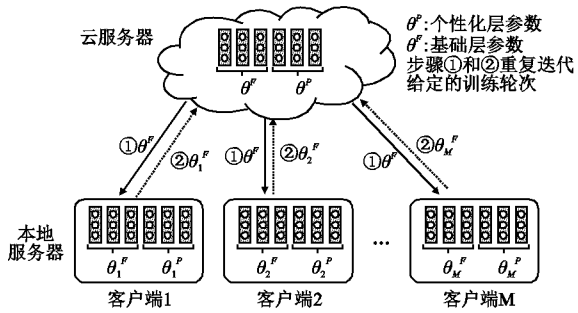


图 2 基于参数解耦的预训练

Fig. 2 Pre-training based on parameter decoupling

$$\boldsymbol{\theta}^{F_{t+1}} = \frac{1}{\sum_{i=1}^{M} |DS_i|} \sum_{j=1}^{M} |DS_j| \cdot \boldsymbol{\theta}_j^{F_{t+1}} \tag{7}$$

其中 $\boldsymbol{\theta}_j^{F_{t+1}}$ 是在 $t+1$ 轮客户端 j 的基础层的参数,M 是客户端的数量,$|DS_j|$ 是客户端 j 的本地训练数据集的大小. 算法 1 描述了预训练阶段的详细细节.

算法 1. 基于参数解耦的预训练.

输入: 本地训练批次大小 B、客户端数量 M、本地迭代轮次 E、预训练阶段全局迭代次数 $R1$、学习率 η、个性化层的划分位置 P

输出: 全局共享的基础层模型参数 $\boldsymbol{\theta}^{F_{R1}}$,所有客户端个性化层的模型参数 $\{\boldsymbol{\theta}_1^{P_{R1}}, \boldsymbol{\theta}_2^{P_{R1}}, \cdots, \boldsymbol{\theta}_M^{P_{R1}}\}$

ServerExecute:

1. 初始化全局模型参数 $\boldsymbol{\theta}$
2. 将全局模型参数分解为基础层 $\boldsymbol{\theta}^{F_0}$ 和个性化层 $\boldsymbol{\theta}^{P_0}$,分发给所有客户端
3. FOR 通信轮次 $t=0,1,\cdots,R1\text{-}1$ DO
4. 云服务器将 $\boldsymbol{\theta}^{F_t}$ 发送至所有客户端
5. FOR 每个客户端 $k \in U$ DO
6. $\boldsymbol{\theta}_k^{F_{t+1}} \leftarrow ClientExecute(k, \boldsymbol{\theta}^{F_t}, \boldsymbol{\theta}_k^{P_t})$
7. END FOR
8. 使用公式(7)聚合本地更新,得到新的全局基础层参数 $\boldsymbol{\theta}^{F_{t+1}}$
9. END FOR
10. 服务器接收每个客户端的个性化层参数
11. RETURN $\boldsymbol{\theta}^{F_{R1}}, \{\boldsymbol{\theta}_1^{P_{R1}}, \boldsymbol{\theta}_2^{P_{R1}}, \cdots, \boldsymbol{\theta}_M^{P_{R1}}\}$ **ClientExecute**$(k, \boldsymbol{\theta}^{F_t}, \boldsymbol{\theta}_k^{P_t})$:
12. 将 $\boldsymbol{\theta}^{F_t}$ 和 $\boldsymbol{\theta}_k^{P_t}$ 连接为完整本地模型 $\boldsymbol{\theta}_k^t$
13. FOR 本地迭代次数 $i=0,1,\cdots,E\text{-}1$ DO
14. FOR 数据 $Batch \in DS_k$ DO
15. 使用公式(5)更新 $\boldsymbol{\theta}_k^t$
16. END FOR
17. END FOR
18. RETURN $\boldsymbol{\theta}_k^{F_{t+1}}$ to the server

在预训练阶段完成后,客户端可以得到公共的基础层参数 $\boldsymbol{\theta}^F$ 和依赖于本地数据分布的特定个性化层参数 $\boldsymbol{\theta}^P$. 这种基于参数解耦的分层更新方式使得不同客户端既可以通过基础层参数实现知识共享,又可以利用个性化层参数满足不同用户的个性化需求. 但是,上述过程无法实现神经网络的深层参数,也就是个性化层参数的协作训练. 考虑到当本地训练数据严重不足时,客户端可能无法训练出一个满足要求的个性化层,算法在第 2 阶段将参数解耦和聚类相结合,利用层次聚类捕捉客户端之间数据分布的相似性,实现深层参数的协作更新. 在下一节将详细介绍算法的第 2 阶段.

2.2 基于凝聚层次聚类的分组训练

在预训练阶段,基础层的参数不参与全局聚合,经过若干轮迭代,每个客户端得到的个性化层参数很大程度上反映了客户端本地数据分布. 也就是说,数据分布类似的客户端得到的个性化层的参数也具有一定的相似性. 为了在避免数据异构带来的影响的同时进一步加强客户端之间的协作,本节使用层次聚类算法对客户端进行分组. 数据分布相同的客户端被分为一组,同组客户端共享相同的全局个性化层.

在执行聚类算法之前,所有的客户端将第 1 阶段获得的个性化层参数上传到云服务器,这些参数被展平为一维向量. 使用凝聚层次聚类算法,云服务器可以在不知道聚类中心数量的前提下对客户端进行分组. 凝聚层次聚类算法最初将所有的 M 客户端各自作为一组,在层次聚类的每一步,云服务器会计算所有组之间的成对距离,将距离最近的两个组合并,不同组之间的距离采用平均距离度量计算,也就是说,通过计算两个组内所有参数之间的距离的平均值来确定两个组之间的距离. 此外,本文设置了一个距离阈值 TH,当距离超过距离阈值 TH 或所有客户端聚为一组时,凝聚层次聚类算法会终止,此时分组完成.

本文假设任意两个客户端的个性化层的参数向量表示为 $\boldsymbol{v}^i$ 和 $\boldsymbol{v}^j$,这两个向量之间的余弦距离可以表示为:

$$dist(\boldsymbol{v}^I, \boldsymbol{v}^j) = 1 - \left(\frac{\boldsymbol{v}^i}{\| \boldsymbol{v}^I \|}, \frac{\boldsymbol{v}^j}{\| \boldsymbol{v}^j \|} \right) \tag{8}$$

对于大规模神经网络,参数的数量极其庞大,即使只考虑个性化层参数,其转化来的向量 $\boldsymbol{v}^i$ 和 $\boldsymbol{v}^j$ 仍然处于相当高的维度. 在高维空间,存在维度灾难的问题,随着维度的增加,向量之间的计算量呈指数倍增长. 传统的距离度量方式如余弦距离和欧式距离会失效,具体来说,随着维度增加,任意两个高维向量之间的最大距离和最小距离的差值趋于零,无法根据余弦距离确定两个向量之间的相似性. 考虑到神经网络的最后一层最接近输出,最能反映客户端的标签分布,本文只选择个性化层的最后一层的参数进行聚类,这种选择可以有效缓解高维灾难问题.

图 3 是分组训练过程的示意图. 首先,参数服务器利用层次聚类对客户端进行分组,分组 i 的客户端集合表示为 C_i. 假设一共 M 个客户端被分为 N 组,在云服务器端,每组客户端均保持一个单独的全局个性化层参数. 同组客户端协作更新同一个全局个性化层,而全局模型的基础层的参数通过聚合所有客户端的基础层参数完成更新. 例如,在图 3 中,假设 $client_{1,1}, client_{1,2}, \cdots, client_{1,m}$ 均为组 C_1 中的客户端. 首先服务器将全局基础层参数 $\boldsymbol{\theta}^F$ 和个性化层参数 $\boldsymbol{\theta}^{P1}$ 发送给 C_1 中客

户端,然后,客户端在本地数据集上执行给定次数的本地训练,分别将更新完成的参数($\boldsymbol{\theta}_1^F,\boldsymbol{\theta}_2^F,\cdots,\boldsymbol{\theta}_m^F$)和($\boldsymbol{\theta}_1^{P1},\boldsymbol{\theta}_2^{P1},\cdots,\boldsymbol{\theta}_m^{P1}$)发送至云服务器聚合. 云服务器根据接收到的参数,使用公式(7)聚合基础层的参数,使用公式(9)聚合得到不同分组的全局个性化层的参数.

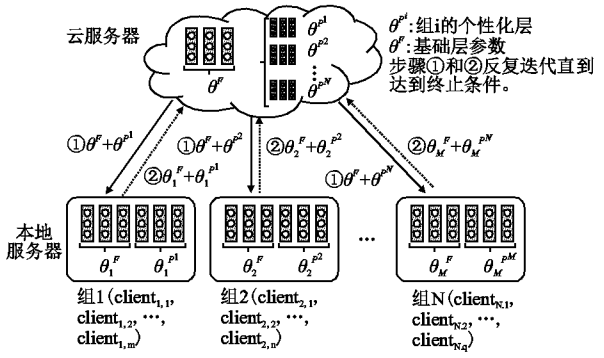


图 3　基于聚类的分组训练

Fig. 3　Group training based on clustering

$$\boldsymbol{\theta}^{P_t^i+1}=\frac{1}{\sum_{j\in C_i}|DS_j|}\sum_{j\in C_i}|DS_j|\cdot\boldsymbol{\theta}_j^{P_t^i+1} \tag{9}$$

其中 $\boldsymbol{\theta}_j^{P_t^i+1}$ 是第 t 轮中组 C_i 中的客户端 j 的个性化层参数. C_i 是组 i 中的客户端索引集合. 这个过程持续迭代直到达到终止条件. 训练完成后,每个客户端都可以得到一个个性化模型. 算法 2 是基于聚类的分组训练算法的细节.

算法 2. 基于层次聚类的分组训练

输入:全局共享的基础层模型参数 $\boldsymbol{\theta}^{FR1}$,各客户端的个性化层的模型参数$\{\boldsymbol{\theta}_1^{PR1},\boldsymbol{\theta}_2^{PR1},\cdots,\boldsymbol{\theta}_M^{PR1}\}$,距离阈值 TH,全局通信轮次 $R2$

输出:个性化模型$\{\boldsymbol{\theta}_1,\boldsymbol{\theta}_2,\cdots,\boldsymbol{\theta}_M\}$

ServerExecute:

1. $C\leftarrow Clustering(\boldsymbol{\theta}_1^{PR1},\boldsymbol{\theta}_2^{PR1},\cdots,\boldsymbol{\theta}_M^{PR1},TH)$;
   //生成聚类结果
2. 使用公式(9)计算每个组的全局个性化层参数 $\boldsymbol{\theta}^{P_{R1}^i}$;
3. FOR 通信轮次 $t=R1,R1+1,\cdots,R1+R2\text{-}1$ DO
4. 云服务器将每组的个性化层参数 $\boldsymbol{\theta}^{P_t^i}$和基础层参数 $\boldsymbol{\theta}^{F_t}$发送给所有客户端;
5. FOR 对于每组客户端 $C_i\in C$ DO
6. FOR 客户端 $k\in C_i$ DO
7. $\boldsymbol{\theta}_k^{F_t+1},\boldsymbol{\theta}_k^{P_t^i+1}\leftarrow ClientExecute(k,\boldsymbol{\theta}^{F_t},\boldsymbol{\theta}^{P_t^i})$;
8. END FOR
9. END FOR
10. 使用公式(7)聚合基础层参数为 $\boldsymbol{\theta}^{F_t+1}$;
11. 使用公式(9)聚合不同组个性化层参数为$\{\boldsymbol{\theta}^{P_t^1+1},\boldsymbol{\theta}^{P_t^2+1},\cdots,\boldsymbol{\theta}^{P_t^N+1}\}$;
12. END FOR
13. RETURN $\{\boldsymbol{\theta}_1,\boldsymbol{\theta}_2,\cdots,\boldsymbol{\theta}_M\}$;

ClientExecute$(k,\boldsymbol{\theta}^{F_t},\boldsymbol{\theta}^{P_t^i})$:

14. 将 $\boldsymbol{\theta}^{F_t}$和 $\boldsymbol{\theta}_k^{P_t^i}$ 连接为完整本地模型 θ_k^t;
15. FOR 本地迭代次数 $i=0,1,\ldots,E\text{-}1$ DO
16. FOR 数据 $Batch\in DS_k$ DO
17. 使用公式(5)更新 $\boldsymbol{\theta}_k^t$;
18. END FOR
19. END FOR
20. RETURN $\boldsymbol{\theta}_k^{t+1}=(\boldsymbol{\theta}_k^{F_t+1},\boldsymbol{\theta}_k^{P_t^i+1})$;

在设置层次聚类算法的距离阈值时,如果设置一个较大的距离阈值,那么聚类后同组客户端的数据分布差异也就越大,数据异质性带来的负面作用越大,同时,用户之间协作也会增强. 如果设置一个较小的距离阈值,意味着客户端之间的协作会减弱,数据异质性带来的影响也会减弱. 为了平衡用户协作和数据异质性的影响,需要设置一个合适的距离阈值,获得更好的模型性能.

3　仿真实验

3.1　数据集和模型

本文在两个广泛使用的联邦学习数据集 CIFAR-10[19] 和 CIFAR-100[19] 上进行了实验,这两个数据集均为图像分类数据集. 本文设计了一个小的卷积神经网络模型 CNN-CIFAR10,包含两个 5×5 的卷积层(分别为 6 和 12 的通道),在每个卷积层之后均有一个 2×2 的最大池化层,之后是 3 个使用 RELU 激活函数的全连接层,最后是一个 SoftMax 输出层. 此外,本文还使用了两种常见的神经网络模型 ResNet-34[20] 和 MobileNet-V1[21]. 在所有的实验中,客户端均采用随机梯度下降进行本地模型更新,批量大小为 128,学习率为 0.01.

3.2　数据集的划分

本文将所有的客户端分为 5 组,此外,实验用 γ 来表示数据异构的程度. 如 $\gamma=0.8$,表示 20% 的数据会按标签均匀分配给所有的客户端,80% 的数据会根据组进行分配. 例如,对于 CIFAR-10 数据集,一共分为 5 组,标签为 0 和标签为 1 的数据的 80% 被均匀分配给第 1 组内的客户端,标签为 2 和标签为 3 的数据的 80% 被均匀分配给第 2 组内的客户端,以此类推. 而剩余的 20% 的数据则是在所有的客户端之间均匀分配. 对于 CIFAR-100 数据集,标签 0 ~ 19 的数据的 80% 被均匀分配给第 1 组内客户端,剩余 20% 在所有客户端之间均匀分配. γ 可以在一定程度上模拟工业物联网不同设备数据标签异构. 此外,本文用 α 表示每个客户端的训练数据集占本地数据的比例,用于调整不同客户端的本地数据集的大小.

3.3　实验环境和对比方法

本文实验在个人电脑上使用 Pytorch1.12.0 通过单机模拟联邦学习过程进行实验. 实验设备的 CPU 为 AMD R7 5800H,内存为 16G,显卡为 NVIDIA RTX 3030,显存为 6G.

本文将所提出的个性化联邦学习算法与主流的个性化联邦学习算法进行了比较. 其中,FedAvg[4] 是被广泛引用的经典联邦学习算法. FedPer[17] 通过将模型划分为基础层和个性化层来实现个性化,在实验中,FedPer 与本文提出的算法保持相同的模型划分方式. FedProx[16] 通过添加一个近端项来控制本地模型和全局模型之间的偏差,近端项的系数被设置为 0.1. 迁移学习[9](TL)利用迁移学习实现个性化,考虑到联邦学习不共享任何数据的原则,对于 TL,实验首先利用联邦平均算法获得一个全局模型,然后在本地数据集上进行微调. 此外,实验也测试了客户端不参与联邦学习,仅仅在本地数据集

上使用随机梯度下降时的性能作为对照,用 SGD 表示.

3.4 算法性能评估

3.4.1 个性化层划分位置和预训练轮次的影响

实验评估了在不同的个性化层划分位置 P 和预训练轮次 $R1$ 下客户端本地模型在测试集上的准确率的平均值、最大值和最小值,其中,P 代表个性化层的划分位置,例如,$P=2$ 表示最后两个全连接层被划分为个性化层,其余层被划分为基础层. 本实验基于 CNN-CIFAR10 模型在 CIFAR-10 数据集上进行,一共设置了 100 个客户端,$\alpha=0.7$,$\gamma=0.8$,距离阈值 $TH=0.01$,本地迭代次数 $E=4$. 表 1 展示了 100 个全局通信轮次之后的实验结果. 可以看出,预训练轮次 $R1$ 与最终的模型准确率之间没有明显的联系,这是因为只要设置一个合适的距离阈值 TH,无需太多训练轮次,数据分布相同的客户端即可被分为一组. 此外,从表 1 可以看出,随着个性化层包含

表 1 不同个性化层划分位置和预训练轮次的影响

Table 1 Influence of different personalization layer division locations and pre-training rounds

P	准确率(%)	R1 = 1	R1 = 3	R1 = 5	R1 = 7	R1 = 10
1	平均值	66.5	66.6	**67.1**	**67.1**	66.4
	最大值	77.6	80.2	78.2	**80.9**	79.5
	最小值	**56.5**	**56.5**	**56.5**	53.1	54.4
2	平均值	64.5	64.7	64.1	65.3	65.1
	最大值	**80.9**	76.1	78.9	**80.9**	79.5
	最小值	53.7	53.7	53.1	53.7	53.7
3	平均值	63.1	62.3	62.4	62.9	62.3
	最大值	76.8	76.8	76.8	78.9	76.8
	最小值	50.3	48.9	49.6	51.0	48.9

的层数原来越小,本地模型的最终准确率越来越高,当 $P=1$ 时,取得最高的平均准确率 67.1%. 这种情况可能是由于每个客户端分配到的训练数据太少,因此难以完成过于复杂的个性化层的训练,当个性化层包含的参数过多时,客户端的本地模型的准确率随之降低.

3.4.2 距离阈值对分组结果的影响

距离阈值 TH 的设置决定了何时终止聚类,设置一个合适的距离阈值对于正确的客户端分组有重要意义. 本小节探究不同的距离阈值对分组结果的影响,实验基于 CNN-CIFAR10

表 2 不同距离阈值下的分组结果

Table 2 Clustering results under different distance thresholds

TH	聚类结果
0.4	{0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19}
0.35	{0,1,2,3} {4,5,6,7} {8,9,10,11]} {12,13,14,15} {16,17,18,19}
0.3	{0,1,2,3} {4,5,6,7} {8,9,10,11} {12,13,14,15} {16,17,18,19}
0.1	{0,1,2,3} {4,5,6,7} {8,9,10,11} {12,13,14,15} {16,17,18,19}
0.01	{0,1,2,3} {4,5,6,7} {8,9,10,11} {12,13,14,15} {16,17,18,19}
0.001	{12,15} {0,1,2,3} {6,7} {16} {14} {19} {13} {9} {10} {5} {11} {17} {18} {8} {4}

模型在 CIFAR10 数据集上运行. 实验设置了 20 个客户端,每 4 个客户端被分为一组,$\alpha=0.7$,$\gamma=0.8$,$P=1$,$R1=10$,$E=4$. 实验结果如表 2 所示,可以看出,当距离阈值在[0.01, 0.35]之间,所有的客户端均可以被正确分组. 当 TH 被设为 0.4 时,由于所有的客户端的参数的余弦距离均小于 0.4,因此所有的客户端被分为一组,此时算法等价于 FedAvg. 当距离阈值 TH 被设为 0.001 时,由于绝大多数的客户端的参数之间的余弦距离大于 0.001,因此无法完成聚类,只能独自更新个性化层的参数. 如果距离阈值设置过小,所有客户端各为一组,此时算法等价于 FedPer. 关于距离阈值的选取,聚合服务器可以在聚类之前计算所有客户端之间的成对余弦距离,如果存在明显的分界,那么可以选择一个位于分界内的距离阈值. 如果没有明显的分界,建议选取一个相对较小的距离阈值,例如 20% 分位点来降低异构客户端分为一组的概率,缓解工业物联网环境下数据异构带来的负面影响.

3.4.3 不同数据异构水平下与 FedAvg 的性能比较

本节研究算法在不同数据异构水平 γ 下的稳定性. 实验基于 MobileNet-V1 模型在 CIFAR-10 数据集上进行,实验一共设置了 50 个客户端,$\alpha=0.7$,$TH=0.15$,$R1=10$,$R2=60$,$E=4$,$P=1$. 表 3 展示了实验结果,本节选取 5 个不同的 γ 值,即 0.2,0.4,0.6,0.8 和 1. 当数据异构水平为 0.2 时,不同客户端的数据分布相差不同,此时,提出的算法性能略低于 FedAvg. 在其余情况下,提出的算法性能始终优于 FedAvg,这表

表 3 不同数据异构水平下的性能

Table 3 Performance under different levels of data heterogeneity

	准确率(%)	数据异构水平				
		0.2	0.4	0.6	0.8	1
Ours	平均值	55.79	**58.69**	**62.91**	**72.04**	**84.29**
	最大值	64.66	67.01	74.66	82.82	92.00
	最小值	48.66	51.03	51.66	61.61	74.00
FedAvg	平均值	**57.39**	55.94	52.26	41.67	27.91
	最大值	65.33	65.66	64.33	63.63	82.00
	最小值	50.01	46.66	39.33	28.28	0.00

明在不同的数据异构水平下,本文提出的个性化联邦学习算法的有效性和稳定性. 随着数据异构水平的提高,提出的算法的准确率持续提高. 这是因为每个客户端的数据分布更加集中,本地数据集中主导标签的比例越来越高. 由于不同客户端的数据分布的差异,本地模型参数的更新是完全不同的,FedAvg 的准确率越来越差直到无法收敛.

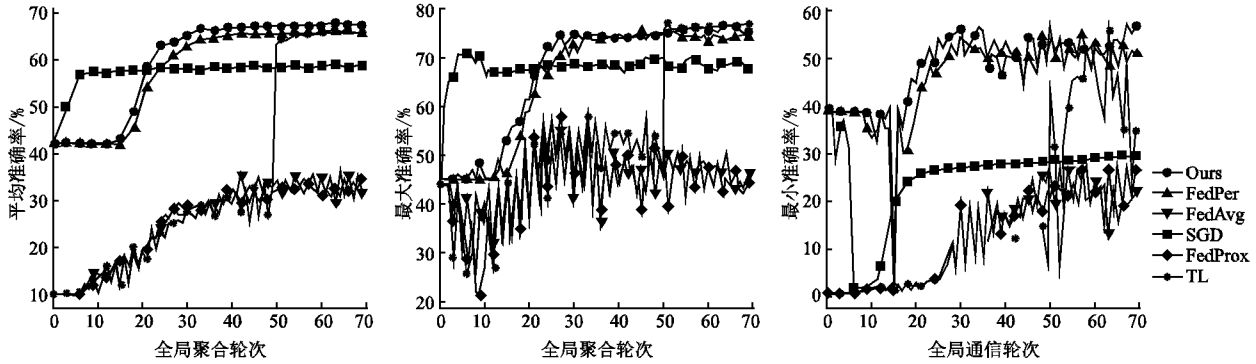
3.4.4 与主流个性化联邦学习算法的性能比较

本节将所提出的算法与主流的个性化联邦学习算法进行比较,分别为 FedAvg、FedPer、FedProx、TL. 实验分别在 CIFAR-10 和 CIFAR-100 数据集上进行. 为了模拟不同客户端数据不平衡的情况,每个客户端的训练数据集的比例从[0, 0.7]之间随机抽取.

图 4 是基于 Mobilenet-V1 模型在 CIFAR-10 数据集上的实验结果,一共设置了 50 个客户端,$\gamma=0.8$,$P=2$,$R1=10$,$R2=60$,$E=4$,$TH=0.1$. 对于迁移学习,实验先使用 FedAvg 运行 50 个全局通信轮次得到全局模型,然后在客户端本地数据集上微调 20 轮. 可以看出 FedAvg 和 FedPer 的性能是最差的,甚至低于随机梯度下降的准确率,因为这两个算法仅保留一个全局模型,无法满足所有客户端的需求,全局模型偏离全

局最优. 本文所提出的算法取得了最高的平均准确率,这也表明了提出的算法的优越性. FedPer 和 TL 的性能略微低于提出的算法,这是因为提出的算法在利用基础层实现个性化的时候仍然保持相似用户之间的协作,每个客户端都可以利用来自同组客户端的知识获得一个更好的个性化模型. 而 FedPer 中每个客户端单独训练自己的个性化层,TL 的每个客户端也是单独训练自己的个性化模型.

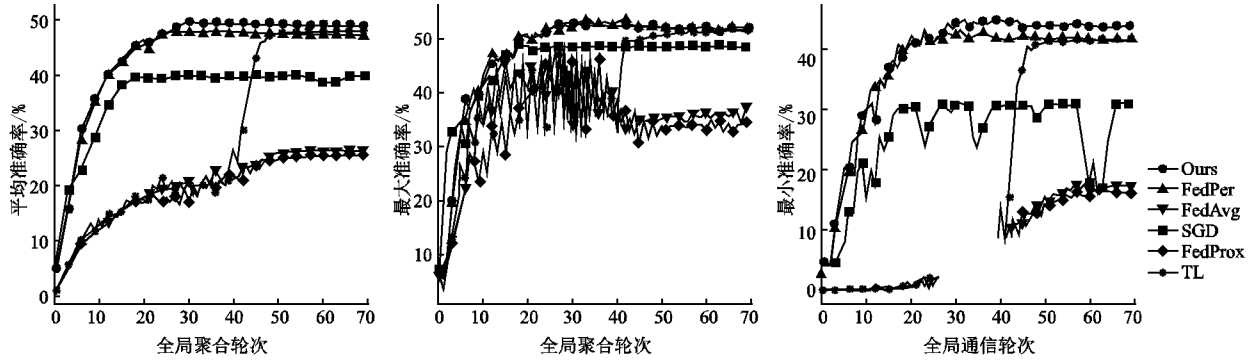
图 5 是基于 Resnet-34 模型在 CIFAR-100 数据集上的实验结果,由于内存限制,仿真实验只设置了 15 个客户端,$\gamma = 1$,$P = 1$,$R1 = 10$,$R2 = 60$,$E = 1$,$TH = 0.01$. 由于 CIFAR-100 数据集中每个标签包含的数据仅为 CIFAR-10 的 1/10,分类任务难度增加,因此,所有的算法的准确率都低于在 CIFAR-10 数据集上的实验. 但本文所提出的算法仍然取得了最高的准确率,这也证明了算法的稳定性. 此外,可能由于某个客户端数据较为充足,导致 FedPer 的客户端最高准确率略高于本文提出的算法.



(a) 客户端本地模型平均测试准确率　(b) 客户端本地模型最大测试准确率　(c) 客户端本地模型最小测试准确率

图 4　在 CIFAR-10 数据集上不同的个性化联邦学习算法的客户端本地模型平均、最大和最小测试准确率曲线

Fig. 4　Average, maximum and minimum test accuracy curves of the client local model for different personalized federated learning algorithms on the CIFAR-10 dataset



(a) 客户端本地模型平均测试准确率　(b) 客户端本地模型最大测试准确率　(c) 客户端本地模型最小测试准确率

图 5　在 CIFAR-100 数据集上不同的个性化联邦学习算法的客户端本地模型平均、最大和最小测试准确率曲线

Fig. 5 Average, maximum and minimum test accuracy curves of the client local model for different personalized federated learning algorithms on the CIFAR-100 dataset

本文提出的算法在两个数据集上均取得了优于其他算法的性能,这也证明了算法的优越性. 由于提出的算法可以识别客户端之间的相似性,不仅仅实现基础层参数的协作,还保持同构客户端之间个性化层的深度协作,数据不足的客户端可以从中受益,获得更高的准确率.

3.4.5　个性化性能比较

为了进一步研究在数据高度异构情况下个性化联邦学习算法的个性化性能,本节在图 6 中以箱型图的形式展示了所有客户端的准确率,实验使用 Mobilenet-V1 模型在 CIFAR-100 数据集上进行,客户端编号为偶数的训练数据集比例 α 设为 0.3,其余客户端的 α 设为 0.6,TH 设为 0.1,其余参数设置与上一个实验相同. 可以看出,由于严重的数据分布异构,FedAvg 和 FedProx 的所有客户端的准确率均低于 SGD. 这表明,在具有高度异构数据的工业物联网场景,单一模型无法满足所有客户端的需求,必须为用户提供个性化的模型. FedPer 和 TL 以及本文提出的算法都通过提供多个模型适应不同的数据分组,客户端的准确率优于 SGD. 在满足个性化的同时,

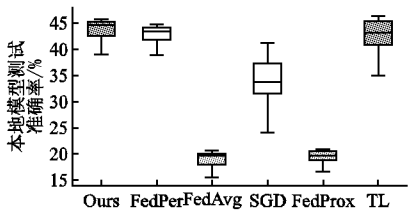


图 6　所有客户端的本地模型测试准确率的箱型图

Fig. 6　Box plot of local model test accuracy for all clients

本文提出的算法使用聚类识别客户端的相似性,实现同构客户端的协作.因此,更多的客户端获得了更高的准确率,这表明本文提出的个性化联邦学习算法具有更好的个性化性能.

3.4.6 动态计算资源调度算法的有效性

本小节实验评估了本文提出的动态计算资源调度算法的有效性,为了便于量化,模拟不同设备计算资源异构的场景,实验设置了 10 个用户,每个用户拥有 10 台工业物联网设备,设备的计算能力从[2,10]范围内随机抽样,本地服务器计算能力为 20.假设计算能力为 1 的设备每秒可以训练 100 条数据.每个设备所拥有的训练数据的取值为{100,200,300,400,500,600,700,800,900,1000}.实验模拟了不同训练数据规模下,采用本文的动态调度方案与不采用本文方案时所有客户端均完成本地训练的单轮迭代时间.从图 7 可以看出,当训练数据规模较小时,所有设备均可以完成本地训练,两个方案的

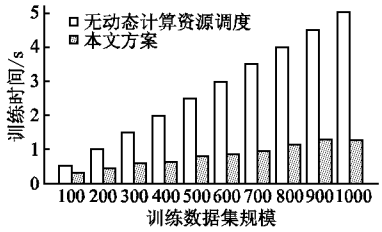


图 7 不同训练数据规模下的单轮训练时间

Fig. 7 Single round training time under different training data scales

训练时间均较低.随着本地训练数据规模的提高,未采用动态资源调度算法的方案出现掉队者,完成训练的设备仍需等待计算能力较弱的设备,其迭代时间等于最慢设备的计算时间.而本文的方案由于可以动态协调多个设备的计算资源,大大降低了完成一轮训练所需的时间,有效缓解了设备异构中掉队者带来的影响.

4 结 论

针对联邦学习在工业物联网应用所面临的设备异构和数据异构问题,本文设计了一个联邦学习数字孪生框架,利用数字孪生动态监控调度可用计算资源完成本地训练,3 层架构缓解了通信压力;同时,本文提出了一个融合参数解耦和聚类的个性化联邦学习算法,参数解耦实现个性化,聚类实现同构客户端的深层协作,在协作的同时为用户提供个性化模型.与主流的个性化联邦学习算法的对比实验,验证了提出的算法的有效性和稳定性.未来的研究方向包括如何应对更加复杂的异构场景,设计自适应的个性化方案.

References:

[1] Sisinni E, Saifullah A, Han S, et al. Industrial internet of things: Challenges, opportunities, and directions[J]. IEEE Transactions on Industrial Informatics, 2018, 14(11): 4724-4734.

[2] Cheng J, Chen W, Tao F, et al. Industrial IoT in 5G environment towards smart manufacturing[J]. Journal of Industrial Information Integration, 2018, 10: 10-19, doi: 10.1016/j.jii.2018.04.001.

[3] Zhang P, Wang C, Jiang C, et al. Deep reinforcement learning assisted federated learning algorithm for data management of IIoT[J]. IEEE Transactions on Industrial Informatics, 2021, 17(12): 8475-8484.

[4] McMahan B, Moore E, Ramage D, et al. Communication-efficient learning of deep networks from decentralized data[C]//Artificial Intelligence and Statistics, 2017: 1273-1282.

[5] Wu Q, He K, Chen X. Personalized federated learning for intelligent IoT applications: a cloud-edge based framework[J]. IEEE Open Journal of the Computer Society, 2020, 1: 35-44, doi: 10.1109/OJCS.2020.2993259.

[6] Karimireddy S P, Kale S, Mohri M, et al. Scaffold: stochastic controlled averaging for federated learning[C]//International Conference on Machine Learning, 2020: 5132-5143.

[7] Xie C, Koyejo S, Gupta I. Asynchronous federated optimization[J]. arXiv preprint arXiv: 1903.03934, 2019.

[8] Sun W, Lei S, Wang L, et al. Adaptive federated learning and digital twin for industrial internet of things[J]. IEEE Transactions on Industrial Informatics, 2020, 17(8): 5605-5614.

[9] Lu Y, Huang X, Zhang K, et al. Communication-efficient federated learning for digital twin edge networks in industrial IoT[J]. IEEE Transactions on Industrial Informatics, 2020, 17(8): 5709-5718.

[10] Zhang J, Liu Y, Qin X, et al. Energy-efficient federated learning framework for digital twin-enabled industrial internet of things[C]//IEEE 32nd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), 2021: 1160-1166.

[11] Zhao Y, Li M, Lai L, et al. Federated learning with non-iid data[J]. arXiv preprint arXiv: 1806.00582, 2018.

[12] Yang M, Wang X, Zhu H, et al. Federated learning with class imbalance reduction[C]//29th European Signal Processing Conference (EUSIPCO), 2021: 2174-2178.

[13] Chen Y, Qin X, Wang J, et al. Fedhealth: a federated transfer learning framework for wearable healthcare[J]. IEEE Intelligent Systems, 2020, 35(4): 83-93.

[14] Jiang Y, Konečný J, Rush K, et al. Improving federated learning personalization via model agnostic meta learning[J]. arXiv preprint arXiv: 1909.12488, 2019.

[15] Li T, Sahu A K, Zaheer M, et al. Federated optimization in heterogeneous networks[C]//Proceedings of Machine Learning and Systems, 2020: 429-450.

[16] Arivazhagan M G, Aggarwal V, Singh A K, et al. Federated learning with personalization layers[J]. arXiv preprint arXiv: 1912.00818, 2019.

[17] Briggs C, Fan Z, Andras P. Federated learning with hierarchical clustering of local updates to improve training on non-IID data[C]//International Joint Conference on Neural Networks (IJCNN), 2020: 1-9.

[18] Tao F, Xiao B, Qi Q, et al. Digital twin modeling[J]. Journal of Manufacturing Systems, 2022, 64: 372-389, doi: 10.1016/j.jmsy.2022.06.015.

[19] Krizhevsky A, Hinton G. Learning multiple layers of features from tiny images[J]. Handbook of Systemic Autoimmune Disease, 2009, 1(4): 1-58.

[20] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016: 770-778.

[21] Howard A G, Zhu M, Chen B, et al. Mobilenets: efficient convolutional neural networks for mobile vision applications[J]. arXiv preprint arXiv: 1704.04861, 2017.