基于相似度加速的自适应聚类联邦学习

朱素霞[1,2]，顾玢珂[1,2]，孙广路[1,2]

（1. 哈尔滨理工大学计算机科学与技术学院，黑龙江 哈尔滨 150080；
2. 黑龙江省智能信息处理及应用重点实验室，黑龙江 哈尔滨 150080）

摘 要：为了解决联邦学习过程中数据异质性导致模型性能下降的问题，考虑对联邦模型个性化，提出了一种新的基于相似度加速的自适应聚类联邦学习（ACFL）算法，基于客户端本地更新的几何特性和客户端联邦时的正向反馈实现自适应加速聚类，将客户端划分到不同任务簇，同簇中数据分布相似的客户端协同实现聚类联邦学习（CFL），从而提升模型性能。该算法不需要先验确定类簇数量和迭代划分客户端，在避免现有基于聚类的联邦算法计算成本过高、收敛速度慢等问题的同时保证了模型性能。在常用数据集上使用深度卷积神经网络验证了 ACFL 的有效性。结果表明，所提算法性能与聚类联邦学习算法相当，优于传统的迭代联邦聚类算法（IFCA），且具有更快的收敛速度。

关键词：联邦学习；个性化；聚类；几何特性；正向反馈

中图分类号：TP301

文献标志码：A

DOI: 10.11959/j.issn.1000−436x.2024069

Adaptive clustering federated learning via similarity acceleration

ZHU Suxia[1,2], GU Binke[1,2], SUN Guanglu[1,2]

1. School of Computer Science and Technology, Harbin University of Science and Technology, Harbin 150080, China
2. Heilongjiang Key Laboratory of Intelligent Information Processing and Application, Harbin 150080, China

Abstract: In order to solve the problem of model performance degradation caused by data heterogeneity in the federated learning process, it is necessary to consider personalizing in the federated model. A new adaptively clustering federated learning (ACFL) algorithm via similarity acceleration was proposed, achieving adaptive acceleration clustering based on geometric properties of local updates and the positive feedback mechanism during clients federated training. By dividing clients into different task clusters, clients with similar data distribution in the same cluster was cooperated to improve the performance of federated model. It did not need to determine the number of clusters in advance and iteratively divide the clients, so as to avoid the problems of high computational cost and slow convergence speed in the existing clustering federation methods while ensuring the performance of models. The effectiveness of ACFL was verified by using deep convolutional neural networks on commonly used datasets. The results show that the performance of ACFL is comparable to the clustered federated learning (CFL) algorithm, it is better than the traditional iterative federated cluster algorithm (IFCA), and has faster convergence speed.

Keywords: federated learning, personalization, clustering, geometric characteristic, positive feedback

收稿日期：2023−10−09；**修回日期：**2024−01−17

通信作者：孙广路，sunguanglu@hrbust.edu.cn

基金项目：黑龙江省自然科学基金资助项目（No.LH2021F032）；黑龙江省重点研发计划基金资助项目（No.2022ZX01A34）

Foundation Items: The Natural Science Foundation of Heilongjiang Province (No.LH2021F032), The Key Research and Development Program of Heilongjiang Province (No.2022ZX01A34)

0 引言

在自然语言处理、图像识别、推荐系统等数据密集型应用中，如果对数据进行集中处理计算，就需要耗费大量的计算能力，因此将应用进行模块分解的分布式计算相比单系统更具优势。随着技术的发展，诸如手机、计算机等终端设备产生了海量的数据信息，如何在保护隐私的前提下充分利用客户端的数据是下一代分布式机器学习重要的研究方向。谷歌在 2016 年提出了一种新的分布式学习方法——联邦学习（FL, federated learning）[1-3]。参与联邦学习的客户端在进行本地训练后将多个客户端本地模型发送到服务器进行联邦，而不需要将任何客户端数据上传给服务器或者第三方客户端[4]。然而，由于参与客户端的高度分散性，联邦学习过程中遇到了数据异质性挑战，造成模型精度的严重下降[5-6]。数据异质性问题主要是指参与联邦的客户端在网络中生成和收集的数据虽然都是独立分布的，但并不服从同一采样方法导致的数据非独立同分布（non-IID）问题[7]。

尽管联邦学习可以在隐私保护的前提下进行分布式模型训练[8]，获得一个全局模型，但它可能并不是每个客户端的最佳解决方案，因为客户端数据实际上是 non-IID 的，不同用户可能有不同的使用倾向。合理地利用数据异构性可以在广告投放、推荐系统和医疗领域[9]创造巨大的价值，例如不同人群喜欢不同类别的新闻，广告提供商需要精准分析这类人群特征以推送不同的题材；年龄、地域不同的患者的一些检测的异常值也会发生改变，而不能仅凭借单一标准诊断。因此，单一全局模型并不能很好地适应不同用户独特的数据分布。

当前，已有研究者对数据异质性问题进行探索，并提出了若干方法来应对该挑战。其中一条技术路线就是个性化联邦学习[10-11]，理论上每个客户端都应该拥有个性化模型而不是全局共享的模型来适应独特的本地数据。通常联邦学习有多种个性化方案，包括微调[12-15]、多任务学习[16-18]、知识蒸馏[19-21]等。聚类联邦学习（CFL, clustered federated learning）[22]是值得考虑的技术路线，它是客户端本地训练和经典联邦学习全局训练的折中方法，在联邦训练过程中利用数据分布的相似性将客户端划分到不同的类簇，同一个类簇内的客户端共享一个簇内联邦模型，该方法可以有效缓解客户端本地训练数据量不足的问题，也可以有效降低 non-IID 对联邦模型的影响。然而，现有聚类联邦学习工作仍存在不足，例如现有聚类联邦研究中经典的迭代联邦聚类算法（IFCA）[23]提出了迭代的 K 聚类算法，每次通信时客户端都要基于损失最小值原则确认类簇身份，导致本地计算量大幅增加，此外，该算法的性能上限受到了初始类簇身份随机分配的影响。随后，Sattler 等[24]提出了迭代聚类联邦学习算法，根据客户端本地更新的余弦相似度对其进行二分，直到确认所有客户端身份。虽然 CFL 性能比 IFCA 有提升，但仍面对计算量剧烈攀升的挑战。

为了解决经典聚类联邦算法造成的计算量激增和性能受限的问题，本文研究了一种自适应的聚类联邦学习（ACFL, adaptive clustering federated learning）算法。ACFL 试图利用正向激励机制来确认客户端类簇身份，使用本地更新的相似度来实现聚类加速，进一步降低计算成本。本文主要贡献如下。

1) 引入客户端聚类的新算法。利用不同客户端之间的正向激励进行聚类，在较低计算成本的情况下，获得了性能较好的聚类联邦模型。

2) 实现客户端的加速聚类。与 CFL 算法直接利用本地更新余弦相似度进行二分聚类不同，本文利用客户端本地更新的余弦相似度来评估其数据分布，实现 ACFL 聚类过程有序化，从而达到客户端加速聚类的目的，进一步降低计算成本。

3) 进行详尽实验来验证所提算法。构建具有不同异质程度的客户端数据集，在深度神经网络上进行广泛的实验，结果证明了 ACFL 的有效性和稳定性。

1 相关工作

1.1 个性化联邦学习

大量研究倾向于使用联邦模型个性化方案解决数据异质性挑战，个性化联邦包括两大技术路线：全局模型个性化和学习模型个性化。全局模型个性化的主要方法包括正则化本地损失[6,25-27]和基于模型优化的元学习[28-30]等。正则化本地损失易于操作，是对 FedAvg 算法的微调，但学习到的仍是单一全局模型。基于模型优化的元学习通过多任务方式训练一个初始化模型，新任务仅需要很少的训练步骤就可以使模型适应本地数据集，快速地收获一个高效的个性化模型，而该方法取得较好效果的

前提是设备间数据分布是相似的，从而全局模型可以扮演一个很好的初始化角色。学习模型个性化的方法有知识蒸馏、多任务学习、聚类等。知识蒸馏赋能联邦学习实现客户端模型架构高度个性化[19-21]。多任务学习把每个客户端看作一个任务，基于客户端异质性数据去学习不同客户端间的关系，最终为每个客户端学习一个个性化模型。尽管上述方法能够促进联邦学习的个性化，但它们的显著缺点就是过于依赖单设备，导致个性化模型泛化能力较差。实际联邦学习过程中，客户端及数据分布之间往往有明显不同的特征划分，训练一个共享的全局模型并不是最优的。受此启发，考虑令多个数据分布相似客户端协作，通过更丰富的数据实现个性化联邦，避免单客户端个性化造成的模型过拟合，更好地解决数据异质性挑战。因此，聚类联邦学习是一条很有前景的技术路线，它克服了单客户端个性化过程中数据量有限的缺点，实现了多客户端联邦的个性化，也就是说聚类联邦模型具有更好的泛化能力。

1.2　聚类联邦学习

部分聚类联邦学习运用 K-Means 聚类算法，聚类前需要耗费大量额外的计算成本确定聚类数量。Ghosh 等[31]提出了基于 K-Means 的聚类联邦算法，初步训练的客户端参数运行 K-Means 聚类算法，然后类内客户端执行联邦训练。Liu 等[32]提出可以根据不同客户端之间稀疏向量的相似度将它们划分到 K 个类簇中。K-Means 聚类算法的另一个不足之处在于一旦在聚类时错误地对客户端聚类身份进行了划分，后续联邦过程无法对其身份进行纠正，从而导致联邦模型性能受限。Briggs 等[13]提出了通用联邦学习层次聚类，在每轮联邦模型训练期间对客户端子集进行训练。随后，Ghosh 等[23]提出了改进的 K-Means 聚类联邦算法 IFCA，每轮迭代时根据客户端损失函数最小化实现客户端聚类身份动态分配，该算法的不足之处在于本地客户端的计算量很大，每轮都需要将类簇模型发送到本地，然后本地客户端对所有类簇模型进行训练并将其划分到损失值最小的模型对应的类簇。此外，IFCA 仍存在性能受限的问题，这和它要求客户端身份初始化分配有关。因此为了提高聚类联邦模型性能，研究者提出了更先进的自适应迭代聚类联邦，例如 Sattler 等[24]提出了二分迭代 CFL，在进行聚类时不需要指定聚类数，每轮迭代时根据客户端相似度进行二分聚类，聚类簇个数随着轮次迭代而自适应变化。CFL 虽然有了更好的性能保证，但是依旧需要付出巨大的计算成本。

2　问题描述

本节将对聚类联邦学习进行形式化描述，并讨论如何在未知数据分布信息的情况下对不同数据异质性的客户端做出评估。

2.1　联邦聚类学习形式化描述

联邦学习作为一种分布式学习方法，允许多个客户端在服务器协调下基于它们的异质性数据共同训练一个联邦模型，训练过程中客户端的数据只在本地运用，并不会泄露给第三方，因此有极强的隐私性。在第 t 次通信进行联邦训练时，客户端首先会与服务器交互，同步下载服务器存储的全局模型 θ_t。所有客户端基于本地数据进行多批次随机梯度下降迭代训练，实现本地权重更新。

$$\Delta\theta_k^{t+1}=\mathrm{SGD}(D_i,D_t)-\theta_t,k=1,\cdots,N \tag{1}$$

最后，所有客户端上传它们本地模型的更新到服务器，服务器对其加权聚合以进行全局模型的更新[33]。

$$\theta^{t+1}=\theta^t+\sum_{k=1}^{N}\frac{|D_k|}{|D|}\Delta\theta_k^{t+1} \tag{2}$$

联邦学习最初旨在训练一个单一的全局模型去拟合所有客户端的数据，即存在最优全局模型 θ^* 使所有客户端基于本地数据 D_k 的损失值最小，形式化如下

$$F_k(\theta^*)\leqslant\min F_k(\theta) \tag{3}$$

其中，$F_k(\theta)$ 是客户端 k 基于其本地数据 D_k 的预测损失值。然而，实际联邦应用中训练得到的全局模型很难满足式(3)，这主要是因为客户端之间的数据分布通常有较大的差异[34]，即数据异质性导致式(3)很难满足。如果按照式(3)进行联邦训练为所有客户端生成统一的全局模型，该模型并不能很好地拟合所有客户端的本地数据，为了实现更好的联邦模型性能，可以考虑使用聚类联邦学习来代替传统的联邦学习方法，本文假定所有 N 个客户端可以根据它们的本地数据分布划分到 M 个类簇，$C=\{c_1,\cdots,c_M\}$，其中 $\bigcup_{m=1}^{M}c_m=\{1,\cdots,N\}$，每个类簇内客户端数据分布相似，簇内客户端可以满足式(3)，从而可得到聚类联邦学习形式化计算式为

$$F_{c_i}^k(\theta^*) \leqslant \min F_{c_i}^k(\theta) \tag{4}$$

其中，$F_{c_i}^k(\theta)$ 是类簇 c_i 内客户端 k 基于其本地数据 D_k 在类簇联邦模型上的损失值。

2.2 基于余弦相似度的客户端评估

方便起见，假定全部 N 个客户端的数据是随机采样自若干数据分布的其中一个，即

$$D_k \sim \alpha_k(x,y) \tag{5}$$

在某轮联邦训练时，每个客户端都对应一个本地模型更新 $\Delta\theta_k$，显然联邦模型当前轮次的更新为

$$\Delta\theta = \sum_{k=1}^{N} \frac{|D_k|}{|D|}\Delta\theta_k = s_1\Delta\theta_1 + \cdots + s_N\Delta\theta_N \tag{6}$$

其中，$s_k = \dfrac{|D_k|}{|D|}$，$D = \bigcup_{k=1,2,\cdots,N} D_k$。在经典联邦学习假设下，联邦学习优化过程如式(1)和式(2)所示，最终联邦学习会收敛到一个稳定最优解 θ^*。这时，当前轮次联邦模型的本地更新接近于 0，即 $\Delta\theta \to 0$。这种情况下会产生如下 2 种可能：第一种可能是所有客户端本地更新都趋于 0，即 $\Delta\theta_1 = \Delta\theta_2 = \cdots = \Delta\theta_N = 0$，意味着所有客户端同时实现了风险函数最小化，也就是说不同客户端之间的数据分布是相同的；第二种可能是客户端的本地更新并不是全部趋于 0，这是训练过程中数据异质性导致的，更加符合联邦学习实际应用场景。此时，考虑通过相似度函数来评估不同客户端的数据分布相似性，从而实现客户端的区分，这里使用式(7)计算任意 2 个客户端本地更新之间的余弦相似度

$$\mathrm{sim}_{i,j} = \frac{<\Delta\theta_i, \Delta\theta_j>}{\|\Delta\theta_i\| \|\Delta\theta_j\|} \tag{7}$$

为了更详细地说明联邦优化过程和余弦相似度的有效性，本文对 4 个客户端的联邦优化路径进行可视化，如图 1 所示。这 4 个客户端采样自 2 个不同的聚类结构。训练过程中，簇内客户端的更新方向并不会发生剧烈变化，类簇之间的更新方向偏差越来越大，余弦相似度快速减小。联邦学习最终收敛到一个稳定最优解，这时类簇 1 和类簇 2 的本地更新方向相反，对应的类簇更新之间的余弦相似度达到最小值−1。

3 研究动机

本文考虑将不同的客户端划分到不同的类簇中，首先探讨了现有经典的联邦聚类算法 IFCA、CFL 的潜在缺点，并以此为动机在第 4 节提出了一个自适应聚类算法 ACFL，旨在获得收敛速度较快的高性能聚类联邦模型。

IFCA 需要在执行前指定 K 个聚类数量，如图 2 所示，假设所有客户端属于 2 个不同的类簇，即 K=2。每个类簇共享一组模型参数，每轮通信时需要迭代计算客户端在不同类簇中的损失值，根据损失最小化原则确认其聚类身份 j。然后不同类簇内客户端利用梯度下降法将本地模型 θ_j 更新为 $\overline{\theta}_j$，并将各个客户端本地更新上传到服务器聚合更新 2 个不同类簇的联邦模型 θ_1, θ_2。CFL 算法是一种自适应聚类算法，不需要指定聚类簇数量，最初通过经典的联邦学习训练，使所有客户端 $\{1,\cdots,M\}$ 收敛得到最优联邦模型 θ^*。在下一次聚类划分时，根据本地更新余弦相似性，将客户端划分到 2 个不同类簇 c_0, c_1 中。每个类簇内的客户端通过类内联邦学习又分别收敛得到联邦模型 θ_0^*, θ_1^*。重复此过程，直到客户端不再满足聚类条件。其核心思想在图 2 中进行展示。值得注意的是，IFCA 和 CFL 作为迭代算法，需要大量的计算消耗来完成客户端聚类身份的

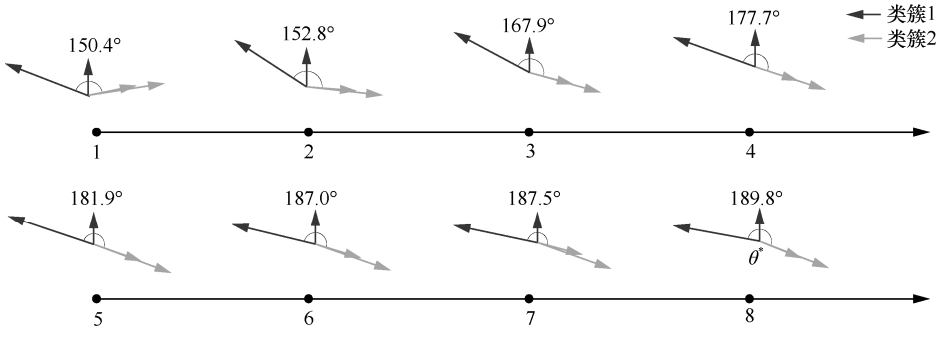


图 1 客户端的联邦优化路径

确认。此外，IFCA 也是一种 K-Means 聚类算法，需要在算法执行前先验设定类簇数 K，探寻最佳类簇数量需要高昂的计算开销，更重要的是，与自适应联邦聚类算法相比，K-Means 聚类算法对聚类联邦模型的性能上限有很大的影响。

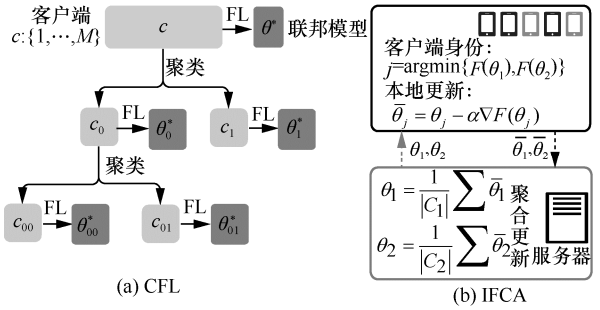


图 2　聚类联邦算法 CFL 和 IFCA 核心思想

基于上述聚类联邦学习中存在的问题，本文提出了改进的基于相似度加速的 ACFL 算法，实现在更少计算成本下快速收敛的高性能聚类联邦模型。该算法的灵感来源是联邦学习过程中不同客户端相互作用，共同决定了模型的性能。本文通过一种新颖的正向激励机制来解决客户端联邦学习过程中的聚类问题，该机制鼓励相似客户端的协作，自适应地实现类簇成员身份的确认。ACFL 算法的核心思想就是正向激励机制，在确认客户端聚类身份的过程中，待聚类客户端依次和类簇成员组合联邦，如果它们的联邦模型性能足够好，那么这个待聚类客户端就属于该类簇。此外，不同客户端的数据分布存在异质性，数据分布越相似的客户端，其联邦训练的模型性能往往效果越好，因此可以考虑使用余弦相似性函数衡量客户端数据分布的相似度，聚类时相似度大的客户端优先与类簇成员组合联邦，当连续多次组合联邦后模型性能不满足正向激励时，停止组合联邦，通过部分待聚类客户端和类簇成员组合联邦，就确认了所有类簇成员，而不需要让所有待聚类客户端进行组合联邦，从而实现聚类加速。

4　算法设计

考虑将不同的客户端划分到不同的类簇中，本节对基于相似度加速的 ACFL 算法进行详细讨论。ACFL 是一种后处理的一次性聚类算法，其核心思想是在经典联邦训练收敛后，通过不断搜索与当前聚类内客户端有正向激励作用的待聚类客户端来确认当前聚类内所有客户端身份，基于客户端相似度排序实现聚类的总体线性有序化，并实现聚类加速。图 3 展示了基于相似度加速的 ACFL 框架。

ACFL 算法流程如下：1) 全部客户端进行经典联邦学习训练至收敛；2) 服务器利用本地客户端将当前轮次的本地更新向量进行相似度计算并依据相似度对客户端排序；3) 根据正向激励原则搜寻当前类簇所有客户端成员，完成簇内客户端联邦模型训练；4) 通过步骤 2)和步骤 3)的迭代执行，确认所有客户端的聚类身份并完成多个个性化簇内联邦模型的训练。

4.1　基于相似度的客户端有序化

如图 4 所示，在聚类操作之前首先根据数据分布相似性大小对客户端进行有序化操作，从而相似性越大的客户端空间距离越近。

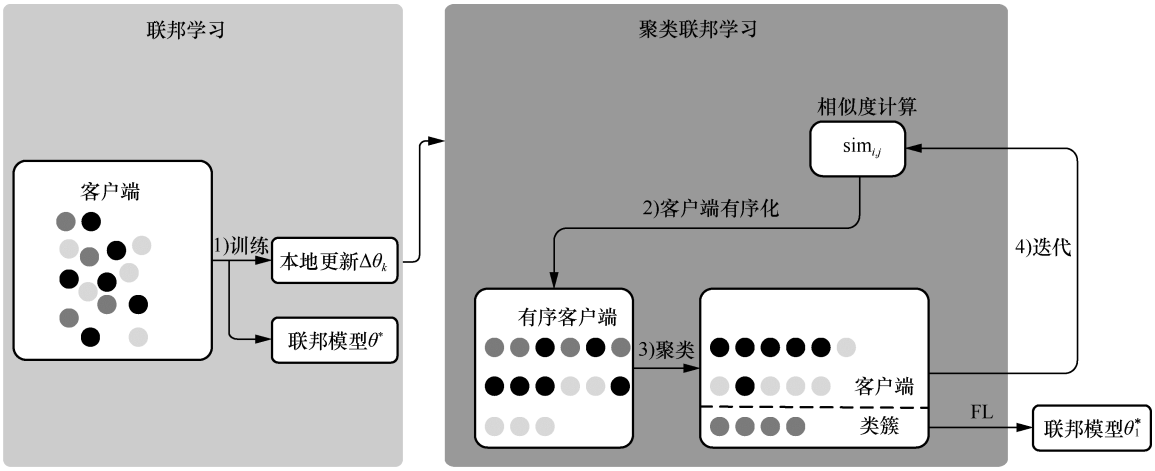


图 3　基于相似度加速的 ACFL 框架

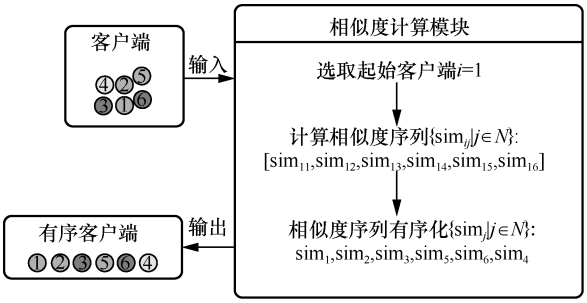


图 4 基于相似度的客户端有序化过程

这里使用 3.2 节中的余弦相似度衡量不同客户端之间的相似性，即式(7)。任意选定一个起始客户端 i，利用联邦学习收敛时不同客户端本地模型更新 $\Delta\theta_k$ 值分别计算其与其他客户端的余弦相似度数值，获得相似度序列 $\{\mathrm{sim}_{i,j} \mid j \in N\}$：$\mathrm{sim}_{11}$，$\mathrm{sim}_{12},\cdots,\mathrm{sim}_{1N}$，接着对相似度序列从大到小进行排序，得到有序的相似度序列 $\{\mathrm{sim}_j \mid j \in N\}$，使与客户端 i 相似度越大的客户端空间位置越靠前，以此映射生成基于相似度的有序客户端序列。

需要注意的是，相似度计算模块的输入是经过经典联邦学习预训练的客户端，目的是通过最后一轮经典联邦学习客户端的本地模型更新参数 $\Delta\theta_k$ 计算不同客户端间的余弦相似度，从而衡量客户端数据分布的相似度。基于相似度的客户端有序化是 ACFL 算法后续实现聚类加速的关键环节，生成的基于相似度的有序客户端序列反映了客户端之间的数据分布相似性的大小，客户端离起始客户端越近说明两者的数据分布越相似，越有可能属于同一个类簇。

4.2 基于正向激励的客户端聚类算法

联邦学习作为分布式机器学习方法，要求每轮通信时不同客户端基于本地数据训练本地模型，将模型参数上传并通过服务器聚合生成全局模型来拟合不同客户端的数据分布，也就是说全局模型的性能受到不同客户端共同作用的约束。对于客户端 $1,\cdots,N$，它们本地训练后的性能分别表示为 $\mathrm{eval}(1),\cdots,\mathrm{eval}(N)$，对应的客户端 $1,\cdots,N$ 通过联邦训练后获得的聚合模型在客户端本地测试集上的准确率分别为 $\mathrm{eval}_{\mathrm{glo}}(1),\mathrm{eval}_{\mathrm{glo}}(2),\cdots,\mathrm{eval}_{\mathrm{glo}}(N)$。如果满足式(8)，说明该组客户端存在正向激励。

$$\sum_{i=1}^{N}\left[\mathrm{eval}_{\mathrm{glo}}(i)-\mathrm{eval}(i)\right] \geqslant \beta \tag{8}$$

其中，β 是超参数，衡量一组客户端通过联邦学习和该组客户端仅仅通过本地训练获得的模型性能的差值，超参数 β 的最优解通过实验获得，本文实验环境下 $\beta_{\mathrm{best}} = -0.17$。需要说明的是，$\beta \to 0^-$，从而避免基于有序客户端序列聚类过程中过度追求联邦性能造成聚类效果下降的问题。当不同客户端满足正向激励机制时，通过联邦学习获得的联邦模型对多方客户端是有益的，与客户端仅通过训练得到的本地模型相比，既保证了它面对不同客户端数据分布时的性能，同时泛化能力也更强。

算法 1 给出了基于正向激励的客户端聚类算法。该算法执行后最终会返回一个包含全体聚类成员的类簇，也就是说通过该算法可以实现同一类簇客户端的聚类操作，确认部分待聚类客户端的类簇身份，从而降低待聚类客户端的数量，重复执行该算法即可确认所有待聚类客户端的类簇身份。

算法 1 基于正向激励的客户端聚类算法

输入 客户端 k 的相似度序列 sim_k，激励参数 β，聚类终止参数 m

输出 目标类簇 c

1) 初始化目标类簇 $c = \{$客户端$k\}$
2) clients_list←sim_k //将相似度序列 sim_k 映射到有序客户端序列
3) for client in clients_list do://根据正向激励机制聚类
4) if $\mathrm{eval}(c,\mathrm{client}) > \beta$ then //满足正向激励
5) c←将 client 添加到目标类簇
6) m←更新聚类终止参数 m 的值
7) record ←更新日志 record 的值//记录不满足正向激励的待聚类客户端
8) end if
9) if record > m then//目标类簇所有成员身份已经确定，提前终止遍历
10) break
11) end if
12) end for
13) clients_list ←更新待聚类的有序客户端序列

首先，客户端 i 设置为目标类簇 c 中的初始成员，即 $c = \{i\}$。然后，该算法需要利用 4.1 节中相似度计算模块进行客户端有序化，计算得到有序待聚类客户端序列 clients_list，显然与目标类簇 c 中的初始成员客户端 i 数据分布越相似的客户端在序列 clients_list 中的位置越靠前。接着，遍历

clients_list 中的客户端，如果其与目标聚类c的成员客户端组合存在正向激励，则说明该客户端属于目标聚类c，将该客户端加入目标聚类；反之，如果当前遍历的客户端 client 和目标聚类c之间不存在正向激励，说明该客户端并不属于目标聚类c，更新事件日志 record，如果事件日志 record 的记录中不满足正向激励的客户端数量超出了预设的终止聚类参数 m，则提前停止对待聚类客户端序列 clients_list 的遍历，此时意味着目标聚类c的所有成员客户端身份都实现了确认。聚类终止参数 m 的值与未确认聚类身份的客户端数量相关，m 的值随着待聚类客户端数量的减小在一定范围内变化。最后，更新 clients_list，将目标聚类c中的成员客户端从待聚类客户端序列 clients_list 中删除，返回确认了所有类簇成员的目标聚类c。

4.3　聚类加速

4.1 节给出了基于相似度的客户端有序化方法，这是实现客户端聚类有序化的必要步骤，从而实现客户端聚类算法 1 中先让目标类簇 c 的成员和数据分布相似度高的待聚类客户端组合联邦，根据正向激励机制来判断相应的待聚类客户端是否属于目标类簇 c。如果存在正向激励，则将这个待聚类客户端添加到目标聚类；反之，则记录当前不满足正向激励的客户端，继续让下一个待聚类客户端与目标类簇 c 的成员组合联邦。如果记录到的不满足正向激励的待聚类客户端数量超过了预设的终止聚类参数 m，说明剩下的待聚类客户端和目标类簇c的成员客户端之间数据分布差异过大，也就是说剩余的待聚类客户端不可能是目标聚类c的成员，可以提前终止遍历剩余的待聚类客户端。如果没有 4.1 节基于相似度的客户端有序化方法，聚类算法 1 中目标类簇 c 的成员和所有待聚类客户端组合联邦就是随机的，因此必须与所有待聚类客户端进行组合联邦才能确认类簇 c 的全部成员身份，而引入 4.1 节基于相似度的客户端有序化方法之后，只需要优先和目标类簇c的成员数据分布相似度高的部分待聚类客户端组合联邦，当不满足正向激励的客户端数量超过阈值时提前终止聚类即可，达到聚类加速的目的。

4.4　ACFL 算法

算法 2 展示了基于相似度加速的 ACFL 的完整过程，该算法起始于 N 个初始化模型参数为$\theta_i^{(0)}$的客户端。

算法 2　基于相似度加速的 ACFL

输入　客户端集合$\{n_k \mid k \in N\}$，学习率γ，初始化模型参数$\theta_j^{(0)}, j \in [N]$

输出　个性化类内联邦模型$\{\theta_i^* \mid i \in 1,2,\cdots\}$

1) for round $= 0,1,\cdots$ do
2)　　$\theta_k, \Delta\theta_k \leftarrow \text{FederatedLearning}(\theta_j^{(0)}, n_k)$;
3) end for
4) 服务器：
5)　　初始化目标类簇$c_i = \varnothing$；
6)　　$\text{sim}_k \leftarrow$ pairwise_similarity($\Delta\theta_k$)　//相似度序列有序化
7)　　while $|C| < N$　do
8)　　$c_i \leftarrow$ clients_clustering(θ_k, sim_k) //调用聚类算法 1，确认目标类簇成员
9)　　$\text{sim}_k \leftarrow \text{Update}(\Delta\theta_k, \text{sim}_k)$ //更新有序化相似度序列
10)　end while
11) for $c_1, c_2, \cdots, c_i$　do
12)　　$\theta_i^* \leftarrow \text{FederatedLearning}(c_i)$　//类内联邦
13) end for

首先，需要对所有客户端进行若干轮的联邦训练直至收敛，并且客户端模型的本地更新都达到聚类条件，这里的聚类条件指的是所有客户端的本地更新$\{\Delta\theta_i \mid i \in N\}$的平均范数和最大范数都在特定的超参数范围内。

其次，初始化聚类集合，中央服务器随机选择一个聚类起始客户端 k，目的是对该客户端所有同簇客户端进行聚类。计算出客户端 k 和其他客户端之间的相似度序列sim_k，并从大到小对相似度序列进行排序。

再次，进行聚类操作，将聚类起始客户端放入当前聚类集合 c 中，根据客户端 k 的相似度序列，评估当前类簇内客户端与客户端 k 数据分布最相似客户端的联邦性能$\text{eval}(c, \text{clients})$，如果性能符合预先设定的正向激励阈值$\beta$，则称该客户端和类内客户端之间有正向作用，即将该客户端划分到当前聚类中。

最后，依次对客户端 k 相似度序列中的客户端进行聚类评估操作，当连续若干次遍历的客户端都与当前类内客户端不产生正向作用时，认为达到了聚类终止条件，当前簇类内的客户端是当前聚类的全部成员，提前终止对相似度序列sim_k对应客户端

的遍历。删除已经确认聚类身份的客户端，实现待聚类客户端列表的更新。

重复聚类算法，直到所有客户端都明确聚类身份。当所有客户端完成聚类操作之后，再分别对不同簇内客户端进行类内联邦训练，获得多个个性化模型。

4.5 算法的计算复杂度分析

现有的研究工作通常都是提出一种新颖的聚类算法，并不会考虑聚类速度和计算量，这对于联邦学习大规模分布式应用是一个很大的挑战。传统的迭代聚类耗费了大量的计算资源，甚至在每轮通信时都需要执行聚类算法，为了解决这个问题，本文提出一种新的聚类联邦学习算法 ACFL，该算法的计算复杂度为 $O(N)$，其中 N 表示客户端数量。该算法克服了现有聚类联邦的缺点，不需要预先人为探寻效果最佳的聚类簇数量 K，也不需要每次通信都迭代运行聚类算法。在 ACFL 算法中，聚类个数完全是自适应的，可以在单一通信轮次确认所有客户端类簇身份，最终聚类数量取决于客户端之间的数据分布情况。CFL 算法的计算复杂度为 $O(h+N^2+M)$，IFCA 的计算复杂度为 $O(TNM)$，其中，h 代表算法迭代的轮数，M 代表类簇数量且 $h \geqslant M$，T 代表算法迭代的轮数（通信轮次），N 代表客户端数量。因此，ACFL 算法可以实现比 IFCA 和 CFL 更少的算力消耗。

5 实验

本节给出了基于相似度加速的 ACFL 算法的实验结果，验证了该算法的有效性。ACFL 能够实现性能强劲的聚类联邦训练，即使当客户端数据异质性发生较大改变时，仍能实现比经典联邦学习 FedAvg 更小的性能波动，具有更好的稳定性。

5.1 实验设置

本文的实验均基于 PyTorch 框架[35]实现，将训练样本根据狄利克雷（Dirichlet）分布采样结果分配给 20 个客户端来模拟本地数据的 non-IID 分布形式，训练了 MNIST[36] CNN（convolutional neural network）和 EMNIST[37] CNN 这 2 个模型。实验中训练的模型是一个 6 层 CNN 模型[38]，包含输入层（INPUT, I1）、卷积层（Convolutions, C2）、池化层（Subsampling, S3）、卷积层（Convolutions, C4）、全连接层（F5）和输出层（OUTPUT, O6），这里卷积核大小设置为 5×5，采用 ReLU 激活函数。此外，在每次全局通信迭代时，设置客户端基于本地数据样本进行本地训练的轮数 epoch 为 15，用于本地更新的 batch_size 为 128，学习率为 0.1。本文评估了在不同程度的 non-IID 数据异质性情况下，使用 CNN 模型基于 2 种数据集 MNIST 和 EMNIST 通过联邦学习训练的模型性能。

5.2 non-IID 数据集的生成

本文在 MNIST 和 EMNIST 数据集的基础上创建了适用于聚类操作的联邦学习数据集。为了模拟不同客户端上本地数据潜在的类簇关系，在实验中进行数据划分时，会先考虑按照常规的 non-IID 划分方法为客户端分配数据，再使用旋转操作使数据集获得对应的聚类结构。具体来说，本文试图根据样本标签的分布来对样本进行划分，以模拟实际应用过程中不同客户端的数据集呈现 non-IID 形式。数据划分的目的是让每个客户端上的样本标签分布尽可能差异化，也就是说每个类别标签的样本需要按照不同的比例划分到不同的客户端上。本文通过对 Dirichlet 分布函数进行随机采样获得类别标签分布矩阵，其行向量表示类别 K 在不同客户端上的概率分布。为了模拟不同 non-IID 程度的客户端数据，考虑将 Dirichlet 分布函数的参数向量 α 设置为 1.0、0.8、0.6、0.4，α 反映了异质性程度，其值越小，数据异质性程度越大。接下来将参与联邦训练的客户端四等分，然后分别将它们本地的图像数据集旋转 0°、90°、180°和 270°，生成 4 种具有清晰聚类结构的客户端。

5.3 结果和分析

本节评估数据异质性下基于相似度加速的 ACFL 算法的收敛速度和性能，性能评价包括准确率以及算法异质性波动指数两部分。在模型收敛速度分析部分，为了验证 ACFL 算法收敛速度快的特点，在 CNN 模型上与同样为自适应算法的 CFL 进行对比。为了突出 ACFL 算法面对 non-IID 数据时的准确率优势，分别在不同程度数据异质性下实验对比了 CFL 算法[24]、IFCA[23]以及 FedAvg 算法[1]。此外，为了验证 ACFL 算法应对不同程度 non-IID 数据挑战时的稳定性优势，分别对比了基线算法 CFL、IFCA、FedAvg 的数据异质性波动指数。

1) 模型收敛速度分析

本节在 CNN 模型上实验对比了 ACFL 和基线算法 CFL 在异质性参数时模型的收敛表现，如图 5 所示。

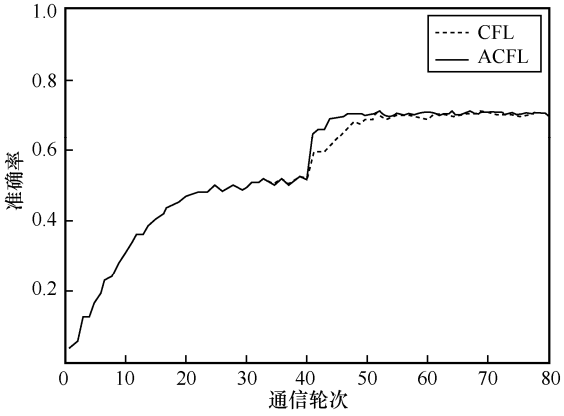


图 5　收敛速度对比

从图 5 可以看到，ACFL 明显加速了聚类联邦模型的收敛速度，在第 40 轮通信时将客户端划分到不同类簇，仅仅经过 5 轮全局通信，聚类联邦模型就达到了收敛；CFL 算法从第 40 轮开始进行了多次迭代的客户端划分操作，经过 20 轮全局通信才能获得收敛的聚类联邦模型。ACFL 算法在收敛速度上更优的原因归结于它不需要多个通信轮次的迭代聚类，而是沿着有序客户端序列基于正向激励机制实现加速聚类联邦学习。

2) 准确率

本节将 ACFL 算法和 3 种基线算法进行了对比，即 CFL、IFCA 和 FedAvg 算法。在实验过程中，设置所有客户端都参与联邦训练。在算法 1 的本地更新步骤中，设置客户端基于本地数据样本进行本地训练的轮数 epoch 为 15，用于本地更新的 batch_size 为 128，学习率为 0.1。对于聚类联邦模型的性能评价，按照如下方式进行：每台参与联邦训练的客户端都会拥有一个测试集，本文让所有类簇内的客户端基于簇内个性化联邦模型给出在其本地测试数据集上的准确率，通过对所有客户端给出的测试集准确率的平均值来衡量不同联邦算法的性能。4 种联邦算法在不同数据异质性程度下基于 MNIST 和 EMNIST 数据集的测试准确率如表 1 所示。

通过表 1 可以看到，ACFL 的准确率在 MNIST 数据集下 α=0.8 时以及 EMNIST 数据集下 α=0.6 时相对 FedAvg 联邦模型的精度提升最大，分别提高了 0.096 和 0.147。在 MNIST 和 EMNIST 数据集下，ACFL 和 CFL 算法的平均准确率分别为 0.912 5、0.916 7 和 0.685 75、0.693 5，也就是说，ACFL 算法达到了和 CFL 算法几乎相同的准确率。在 MNIST 和 EMNIST 数据集下，ACFL、CFL、IFCA 的平均准确率与 FedAvg 算法相比分别提高了 10.8%、11.3%、4.67%和 28.37%、29.82%、14.52%。聚类联邦算法带来了准确率的大幅度提升，充分证明了聚类联邦学习算法技术路线的优异性能。特别地，ACFL 算法在面对复杂数据集时，对性能的提升效果愈发明显。与 FedAvg 算法相比，ACFL 算法在 MNIST 数据集下联邦模型性能提高了 10.8%，而在更复杂的 EMNIST 数据集下联邦模型性能提高了 28.37%。

总体来讲，聚类联邦学习得到的个性化多联邦模型都比 FedAvg 算法中单一全局联邦模型的性能更优。正如表 1 中本文提出的算法 ACFL，其在 2 个不同数据集下可以实现与 CFL 算法相近的联邦模型精度。4.5 节给出了 ACFL 和 CFL 算法的计算复杂度分别为 $O(N)$ 和 $O(h+N^2+M)$，意味着 ACFL 算法可以用更低的计算成本实现与自适应聚类联邦 CFL 几乎一致的模型性能。ACFL 算法明显比 IFCA 具有更高的模型准确率，也就是说 ACFL 算法相比 IFCA 能获得更好的联邦模型，这是因为在 IFCA 下，不同聚类内客户端的初始化是随机的，所有客户端均匀分配到不同类簇中，很大可能会将不同数据分布的客户端划分到同一个初始聚类，经过 IFCA 聚类算法训练得到的联邦模型仍然包含部分数据分布相差较大的客户端的数据样本信息，它试图拟合部分数据分布相差较大的客户端数据，导致最终获得的簇内联邦模型性能相比自适应聚类联邦表现较差。

表 1　4 种联邦算法在不同数据异质性程度下基于 MNIST 和 EMNIST 数据集的测试准确率

α	MNIST				EMNIST			
	ACFL	CFL	IFCA	FedAvg	ACFL	CFL	IFCA	FedAvg
1.0	0.913	0.914	0.862	0.838	0.685	0.691	0.619	0.533
0.8	0.910	0.916	0.866	0.814	0.683	0.699	0.611	0.536
0.6	0.908	0.919	0.863	0.820	0.678	0.686	0.602	0.524
0.4	0.919	0.918	0.857	0.822	0.697	0.698	0.615	0.544

3) 数据异质性波动指数

为了评估不同联邦算法在不同分布数据下的适应性表现，本文引入了一种名为数据异质性波动指数的衡量标准。数据异质性波动指数是通过计算联邦算法在不同 non-IID 程度下性能变化的比率来衡量不同算法面对异质性数据时性能稳定性的指标。显然除了准确率之外，不同算法的数据异质性波动指数是衡量联邦算法优劣的另一个重要指标，算法的数据异质性波动指数越小，说明其应对数据异质性的能力越强。本文使用式(9)进行数据异质性波动指数的计算。

$$\mathrm{wav}_{i,D}=\frac{\delta_{i,D}}{\sum\delta} \tag{9}$$

其中，$\delta_{i,D}=\mathrm{acc}_{i,\max}-\mathrm{acc}_{i,\min}$ 表示算法 i 基于数据集 D 在不同数据异质性参数集合下的模型准确率波动幅度。

通过表 1 中客户端基于不同算法在 4 种数据异质性程度下的性能表现，可以得到 ACFL 和其他 3 种基线算法的数据异质性波动指数如图 6 所示。从图 6 可以看到，3 种聚类联邦算法的数据异质性波动指数都低于经典联邦学习算法 FedAvg，这说明聚类联邦算法具有更强的稳定性优势。

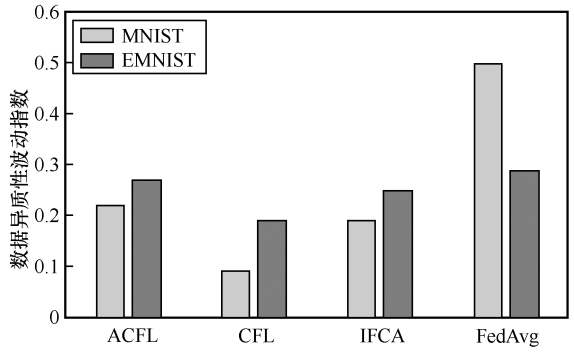


图 6　4 种算法的数据异质性波动指数对比

特别地，在 MNIST 数据集下，ACFL 数据异质性波动指数数值不到 FedAvg 的一半，表现出极其强大的稳定性，这一特性对于应对联邦学习数据异质性挑战是极其重要的。

此外，以 MNIST 数据集为基准，4 种算法在 EMNIST 数据集下数据异质性波动指数的波动幅度也有很大的变化，ACFL 算法的波动幅度是 22%，而 CFL、IFCA、FedAvg 的波动幅度分别为 111%、31.5%、42%，也就是说 ACFL 在不同数据集中的表现并不会出现过大的波动，而 CFL、FedAvg 算法则出现了较明显的波动，这说明 ACFL 算法对数据集的变化有更好的适应性。

6　结束语

本文提出了一种聚类联邦学习算法 ACFL，旨在应对联邦学习数据异质性对模型性能造成的挑战。不同于忽视计算成本和聚类速度的传统聚类联邦学习算法，ACFL 算法基于有序客户端序列根据正向激励机制实现所有客户端的自适应加速聚类，并且具有更低的计算复杂度。在基于 MNIST 和 EMNIST 的 CNN 模型上实验验证了 ACFL 在提高通信效率和模型性能方面的有效性。相比现有的聚类联邦 CFL，ACFL 在 CNN 的模型收敛速度上提高了 25%，能够用更少的计算成本获得和 CFL 几乎相同的准确率。此外，ACFL 在 MNIST 和 EMNIST 数据集下相比传统联邦算法 FedAvg 平均准确率分别提高了 10.8%和 28.37%。

参考文献：

[1] MCMAHAN H B, MOORE E, RAMAGE D, et al. Communication-efficient learning of deep networks from decentralized data[J]. arXiv Preprint, arXiv: 1602.05629, 2016.

[2] KONEČNÝ J, MCMAHAN H B, RAMAGE D, et al. Federated optimization: distributed machine learning for on-device intelligence[J]. arXiv Preprint, arXiv: 1610.02527, 2016.

[3] 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.

[4] 王勇, 李国良, 李开宇. 联邦学习贡献评估综述[J]. 软件学报, 2023, 34(3): 1168-1192.
WANG Y, LI G L, LI K Y. Survey on contribution evaluation for federated learning[J]. Journal of Software, 2023, 34(3): 1168-1192.

[5] TAN Y, LONG G D, LIU L, et al. FedProto: federated prototype learning across heterogeneous clients[J]. Proceedings of the AAAI Conference on Artificial Intelligence, 2022, 36(8): 8432-8440.

[6] LI Q B, HE B S, SONG D. Model-contrastive federated learning[C]// Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Piscataway: IEEE Press, 2021: 10708-10717.

[7] SMITH V, CHIANG C K, SANJABI M, et al. Federated multi-task learning[J]. arXiv Preprint, arXiv: 1705.10467, 2017.

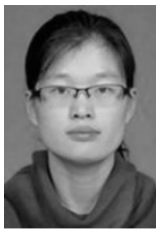
[8] HU H P, WANG D, WU C. Distributed machine learning through heterogeneous edge systems[J]. Proceedings of the AAAI Conference on Artificial Intelligence, 2020, 34(5): 7179-7186.

[9] XU J, GLICKSBERG B S, SU C, et al. Federated learning for healthcare informatics[J]. Journal of Healthcare Informatics Research, 2021, 5(1): 1-19.

[10] TAN A Z, YU H, CUI L Z, et al. Towards personalized federated learning[J]. IEEE Transactions on Neural Networks and Learning Systems, 2023, 34(12): 9587-9603.

[11] CHEN F W, LONG G D, WU Z H, et al. Personalized federated learning with graph[J]. arXiv Preprint, arXiv: 2203.00829, 2022.
[12] LONG G D, XIE M, SHEN T, et al. Multi-center federated learning: clients clustering for better personalization[J]. World Wide Web, 2023, 26(1): 481-500.
[13] BRIGGS C, FAN Z, ANDRAS P. Federated learning with hierarchical clustering of local updates to improve training on non-IID data[C]//Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN). Piscataway: IEEE Press, 2020: 1-9.
[14] SUN B C, FENG J S, SAENKO K. Return of frustratingly easy domain adaptation[C]//Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence. Palo Alto: AAAI Press, 2016: 2058-2065.
[15] VAHIDIAN S, MORAFAH M, CHEN C, et al. Rethinking data heterogeneity in federated learning: introducing a new notion and standard benchmarks[J]. arXiv Preprint, arXiv: 2209.10526, 2022.
[16] CORINZIA L, BUHMANN J M. Variational federated multi-task learning[J]. arXiv Preprint, arXiv: 1906.06268, 2019.
[17] HUANG Y T, CHU L Y, ZHOU Z R, et al. Personalized cross-silo federated learning on non-IID data[J]. Proceedings of the AAAI Conference on Artificial Intelligence, 2021, 35(9): 7865-7873.
[18] SHOHAM N, AVIDOR T, KEREN A, et al. Overcoming forgetting in federated learning on non-IID data[J]. arXiv Preprint, arXiv: 1910.07796, 2019.
[19] LI D L, WANG J P. FedMD: heterogenous federated learning via model distillation[J]. arXiv Preprint, arXiv: 1910.03581, 2019.
[20] LIN T, KONG L J, STICH S U, et al. Ensemble distillation for robust model fusion in federated learning[J]. arXiv Preprint, arXiv: 2006.07242, 2020.
[21] HE C Y, ANNAVARAM M, AVESTIMEHR S. Group knowledge transfer: federated learning of large CNNs at the edge[J]. arXiv Preprint, arXiv: 2007.14513, 2020.
[22] HUANG L, SHEA A L, QIAN H N, et al. Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records[J]. Journal of Biomedical Informatics, 2019, 99: 103291.
[23] GHOSH A, CHUNG J, YIN D, et al. An efficient framework for clustered federated learning[J]. IEEE Transactions on Information Theory, 2022, 68(12): 8076-8091.
[24] SATTLER F, MULLER K R, SAMEK W. Clustered federated learning:model-agnostic distributed multitask optimization under privacy constraints[J]. IEEE Transactions on Neural Networks and Learning Systems, 2021, 32(8): 3710-3722.
[25] LI T, SAHU A K, ZAHEER M, et al. Federated optimization in heterogeneous networks[J]. arXiv Preprint, arXiv: 1812.06127, 2018.
[26] YAO X, SUN L F. Continual local training for better initialization of federated models[C]//Proceedings of the 2020 IEEE International Conference on Image Processing (ICIP). Piscataway: IEEE Press, 2020: 1736-1740.
[27] 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. New York: ACM Press, 2020: 5132-5143.
[28] FALLAH A, MOKHTARI A, OZDAGLAR A. Personalized federated learning with theoretical guarantees: a model-agnostic meta-learning approach[C]//Proceedings of the 34th International Conference on Neural Information Processing Systems. New York: ACM Press, 2020: 3557-3568.
[29] DINH C T, TRAN N H, NGUYEN T D. Personalized federated learning with Moreau envelopes[J]. arXiv Preprint, arXiv: 2006.08848, 2020.
[30] KHODAK M, FLORINA-BALCAN M, TALWALKAR A. Adaptive gradient-based meta-learning methods[J]. arXiv Preprint, arXiv: 1906.02717, 2019.
[31] GHOSH A, HONG J, YIN D, et al. Robust federated learning in a heterogeneous environment[J]. arXiv Preprint, arXiv: 1906.06629, 2019.
[32] LIU B Y, GUO Y, CHEN X Q. PFA: privacy-preserving federated adaptation for effective model personalization[C]//Proceedings of the Web Conference. New York: ACM Press, 2021: 923-934.
[33] LIM W Y B, LUONG N C, HOANG D T, et al. Federated learning in mobile edge networks: a comprehensive survey[J]. IEEE Communications Surveys & Tutorials, 2020, 22(3): 2031-2063.
[34] ZHAO Y, LI M, LAI L Z, et al. Federated learning with non-IID data[J]. arXiv Preprint, arXiv: 1806.00582, 2018.
[35] PASZKE A, GROSS S, MASSA F, et al. PyTorch: an imperative style, high-performance deep learning library[J]. arXiv Preprint, arXiv: 1912.01703, 2019.
[36] DENG L. The MNIST database of handwritten digit images for machine learning research[best of the web[J]. IEEE Signal Processing Magazine, 2012, 29(6): 141-142.
[37] COHEN G, AFSHAR S, TAPSON J, et al. EMNIST: extending MNIST to handwritten letters[C]//Proceedings of the 2017 International Joint Conference on Neural Networks (IJCNN). Piscataway: IEEE Press, 2017: 2921-2926.
[38] ZHOU S R, TAN B. Electrocardiogram soft computing using hybrid deep learning CNN-ELM[J]. Applied Soft Computing, 2020, 86: 105778.

[作者简介]



朱素霞（1978− ），女，黑龙江哈尔滨人，博士，哈尔滨理工大学副教授，主要研究方向为隐私与安全、物联网和并行计算。

顾玢珂（1996− ），男，河南周口人，哈尔滨理工大学硕士生，主要研究方向为联邦学习。

孙广路（1979− ），男，黑龙江哈尔滨人，博士，哈尔滨理工大学教授，主要研究方向为计算机软件、网络计算、信息安全、Agent 技术等。