研究与开发

FedGMH：基于全局多头部的标签干扰消除方法研究

孙圳峰，倪郑威
（浙江工商大学信息与电子工程学院，浙江 杭州 310018）

摘　要：个性化联邦学习因其在应对数据异质性和隐私保护方面的优势而备受关注。现有算法专注于平衡全局信息和个性化信息之间的矛盾，忽视了全局信息中的不同标签信息带来的干扰，尤其在维护单一全局头部的算法中，容易出现标签间特征冲突导致的收敛困难。为此，提出一种新的算法——全局多头部联邦学习（federated learning with global multi-head，FedGMH）算法，该算法在服务器创建多个全局头部，每个头部专门处理一种标签信息，而客户端下载与本地标签相关的全局头部，从而避免无关标签信息的干扰。此外，FedGMH引入参数级聚合机制：评估头部参数重要性，并将关键参数更新为全局多头部的加权参数，以加快收敛速度并且提高准确率。在3个视觉数据集上的大量实验表明，FedGMH优于现有的先进算法。

关键词：个性化联邦学习；数据异质性；标签信息；全局多头部；头部参数
中图分类号：TN918；TP309
文献标志码：A
doi: 10.11959/j.issn.1000−0801.2026005

FedGMH: research on label interference elimination via global multi-head

Sun Zhenfeng, Ni Zhengwei
College of Information and Electronic Engineering, Zhejiang Gongshang University, Hangzhou 310018, China

Abstract: Personalized federated learning has garnered significant attention due to its advantages in addressing data heterogeneity and privacy protection. However, existing algorithms predominantly focus on balancing the contradiction between global and personalized information, often overlooking the interference caused by distinct label information within global features. Particularly in algorithms maintaining a single global head, conflicts between label-specific features can lead to convergence challenges. To address this, a novel algorithm—federated learning with global multi-head (FedGMH) was proposed. The proposed algorithm creates multiple global heads on the server, each dedicated to processing one category of label information. Clients selectively download global heads relevant to their local labels, thereby avoiding interference from unrelated label information. Furthermore, FedGMH incorporates a parameter-level aggregation mechanism: it assesses the importance of head parameters and updates critical parameters

收稿日期：2025−03−09；修回日期：2025−06−17
通信作者：倪郑威，zhengwei.ni@zjgsu.edu.cn
基金项目：桐乡市通用人工智能研究院项目（No.TAGI2-B-2024-0017）
Foundation Item: Tongxiang General Artificial Intelligence Research Institute Project (No.TAGI2-B-2024-0017)

to a weighted ensemble of the global multi-head, accelerating convergence and improving accuracy. Extensive experiments on three visual datasets demonstrate that FedGMH outperforms state-of-the-art baseline algorithms.

Key words: personalized federated learning, data heterogeneity, label information, global multi-head, head parameter

0 引言

联邦学习是一种新兴的分布式机器学习范式，因其具有保护隐私的特点而被广泛应用。它允许每个客户端在本地使用私有数据集进行模型训练，而无须将数据上传至中央服务器。每个客户端在本地训练模型后，仅需将模型参数或模型输出结果发送至服务器。服务器随后对这些本地模型进行聚合，生成全局模型。通过这种方式，联邦学习能够在不泄露原始数据的前提下，实现联合训练，保证数据安全性。

联邦学习的优势在于它可以在不集中数据的前提下进行有效的模型训练。这种方法有效地避免了数据泄露的风险，并且大大降低了数据传输的需求，从而降低了带宽占用和传输成本。这对于那些对数据隐私要求严格或数据量庞大的应用场景来说，具有重要意义。如在医疗健康领域[1-3]、金融服务行业[4-6]以及智慧城市建设[7-8]中，联邦学习技术的应用实例广泛存在。联邦学习以其独特的隐私保护和数据去中心化处理能力，在多个行业中展现出巨大潜力，成为推动技术创新和保障数据安全的重要工具。随着技术的不断成熟和应用场景的拓展，联邦学习有望成为未来数据协作和人工智能发展的基石。

目前，可根据是否为客户端提供私有模型将联邦学习分为传统联邦学习（traditional federated learning，TFL）以及个性化联邦学习（personalized federated learning，PFL），TFL专注于训练一个单一全局模型以适用于所有的客户端，而PFL将为每一个客户端个体或每一组客户端群体提供私有的模型。虽然TFL适用于许多场景，但其面临两个基本挑战：在数据高度异质的情况下收敛性差；缺乏针对特定情况的个性化解决方案。与TFL相比，针对PFL的研究旨在解决这两个挑战[9]。

PFL的核心目标是为每个客户端训练符合其本地数据分布的个性化模型。实现这一目标的一个核心挑战在于如何有效平衡和融合全局信息与个性化信息。特别是在现实场景中，客户端间的数据异质性不仅体现在样本数量上，更关键的是体现在标签分布的显著差异上。常见的挑战是，每个客户端通常只持有全局标签类别的一个小子集。在这种高度标签异质的场景下，传统的全局模型聚合方法面临严峻问题：不同客户端上传的、代表不同标签类别的特征在全局空间中相互干扰甚至冲突，而大量无关标签的噪声信息会严重阻碍模型对本地相关标签特征的有效学习。这些由标签差异导致的干扰和冲突直接造成模型收敛速度显著变慢、性能波动增大，尤其在标签类别众多且客户端持有类别差异大的复杂场景下问题更为突出。

以往大部分PFL算法（如基于个性化层的联邦学习（federated learning with personalization layers，FedPer）[10]、联邦表示学习（federated representation learning，FedRep）[11]等）会将模型解耦成两个部分，分别为头部（一般为模型的最后一层）和特征提取器（除了头部以外的模型参数），并通过其中一部分来获取个性化信息，通过另一部分来获取全局信息，以此来适应非独立同分布数据。然而，这些基于参数解耦的方法在处理客户端间标签分布高度异质的场景时存在显著缺陷。它们通常维护一个单一的全局头部，该头部需要处理所有客户端上传的、混合着无关标签的信息。当客户端本地数据仅包含全局标签类

别的一个小子集时，这种单一头部设计会导致两个主要问题：不同标签的特征信息在共享的表示空间中相互干扰甚至冲突；无关标签的噪声信息会阻碍模型有效学习本地相关标签的特征信息。这些干扰和冲突直接导致模型收敛速度变慢，性能波动增大，尤其是在标签类别众多且客户端持有类别差异大的复杂场景下。与以往研究不同，为了克服单一全局头部在处理标签异质性时的根本局限，寻求一种能够更精细地区分和融合不同标签全局信息的新方法，是本文研究的出发点。

准确模拟客户端之间的数据异构性，是评估PFL算法性能与泛化能力的前提和关键。然而，现有文献在构建实验场景时所采用的数据划分策略，与真实世界存在明显的差距[12-14]。具体而言，主要存在局限如下。

（1）基于数量的类别不平衡策略[12-13]：该策略将每种类别的样本随机且平均地分配给拥有该类别的客户端，导致所有拥有相同类别数量的客户端之间的样本数量一致，这与现实场景中客户端之间样本数量不平衡严重不符。

（2）基于分布的类别不平衡策略[14]：该策略使得每个客户端根据狄利克雷分布（Dirichlet distribution）获得各类数据样本的比例，导致客户端均拥有整个数据集内的大量标签类别，而实际情况下，客户端一般只拥有数据集内的少量标签类别。

针对上述挑战，本文引入了拓展的狄利克雷分布[15]来模拟现实的异构场景。该策略充分结合了上述两种常见策略的优点，具体表示为$\mathrm{ExDir}(C,\alpha)$：首先随机分配C个不同的标签类别给每个客户端，获得每个类别c的先验分布q_c；然后对于每个类别c，从p_c~$\mathrm{Dir}(\alpha q_c)$中抽取p_c，最后将类别c的样本按照$p_{c,m}$的比例分配给客户端m，最终实现更贴近现实、聚焦于“每个客户端仅拥有少量标签类别”这一典型异构场景的模拟效果。

基于对标签异质性挑战和现有方法局限性的深刻认识，本文提出全局多头部联邦学习（federated learning with global multi-head，FedGMH）算法。其核心创新在于：在服务器创建多个全局头部，每个头部专门负责学习和提炼一种特定的全局信息。客户端仅下载与其本地标签相关的全局头部，用于个性化模型的构建。这种方法从根本上避免了无关标签信息的干扰。以往的PFL算法在提取特征向量后，通常将其作为一个头部[16]或两个头部[17-18]的训练样本，以此来训练头部并获取全局信息和个性化信息。然而，这种方法缺乏对不同标签样本之间信息的区分能力。FedGMH的核心创新点正是解决上述标签异质性带来的挑战。本文将通过客户端本地的特征提取器将每个样本的特征向量提取出来，并且将其按照标签进行区分，计算出其对应标签的平均特征向量，上传至服务器，由服务器的全局多头部按照对应的标签进行训练，以得到相应标签的全局头部。这种专注于标签的全局头部设计，确保了每个头部只学习和提炼一种特定标签的全局信息，从而有效避免了不同标签信息间的相互干扰。客户端通过聚合本地标签对应的全局头部和本地头部，即可获取标签全局信息和个性化信息，该方式可降低非本地标签信息对客户端的干扰，加快客户端的收敛速度并提高准确率。

总结本文的贡献如下。

（1）针对现有PFL算法，如联邦全局预测头（federated global prediction header，FedGH）[16]算法，由单一全局头部导致标签信息混杂的核心问题，FedGMH创新性地在服务器构建与全局标签类别总数一致的全局头部。每个头部仅接收相应标签的平均特征向量并进行训练。该架构首次实现了全局信息在标签维度的解耦，显著降低了无关标签噪声干扰。

（2）本文提出了一种参数级个性化头部聚合

机制。该机制应用基于谨慎协同优化的个性化联邦学习（federated cautiously aggressive collaboration，FedCAC）[19]的参数重要性评估思路（即利用参数在本地训练中的变化幅度和最终值来评估其关键程度），在客户端聚合个性化头部时，优先将识别出的关键参数位置更新为下载的全局多头部参数的加权组合（权重由本地标签样本比例决定），而非关键参数则保留本地头部参数。该设计旨在优先融合关键的全局信息，加速模型收敛并提升准确率。

1 相关工作

TFL专注于为全部客户端训练一个单一全局模型，其中主流的算法是联邦平均（federated averaging，FedAvg）算法[12]，它通过将客户端上传至服务器的模型进行平均，得到全局模型。联邦近端优化（federated proximal optimization，FedProx）算法[20]基于FedAvg通过正则化项使其更新的参数保持接近全局模型，以此来提高联邦学习的稳定性。然而，在现实场景下，联邦学习的客户端往往是具有非独立同分布数据的，因此很难通过一个单一全局模型来适应这种情景。

而PFL旨在解决这种非独立同分布的情况，其中有以下几种主要的方法。基于正则化的PFL：Ditto算法[21]、基于Moreau包络的个性化联邦学习（personalized federated learning with moreau envelopes，pFedMe）[22]，基于元学习的PFL：联邦元学习（federated meta-learning，FedMeta）[23]、个性化联邦平均算法（personalized federated averaging，Per-FedAvg）[24]，基于参数解耦的PFL：FedPer[10]、FedRep[11]、局部全局联邦平均（local global federated averaging，LG-FedAvg）算法[25]，基于聚类的PFL：自适应个性化联邦学习（adaptive personalized federated learning，APFL）算法[26]、联邦自适应局部聚合（federated learning with adaptive local aggregation，Fedala）算法[27]、联邦一阶模型优化（federated first order model optimization，FedFomo）算法[28]、FedCAC[19]、联邦注意力消息传递（federated attentive message passing，FedAMP）算法[29]。其中FedCAC的核心创新在于提出了一种评估模型参数对本地数据关键程度的机制，并据此选择其他客户端进行聚合。本文提出的FedGMH算法在个性化头部聚合环节，应用了FedCAC这类参数重要性评估机制来识别头部中的关键参数位置。FedGMH使用参数评估机制的应用场景和目标与FedCAC不同，FedGMH主要用于头部参数聚合，而非客户端选择。

其中，FedPer[10]和FedRep[11]通过聚合特征提取器参数，将头部参数保留在本地得到个性化模型，两者只在本地训练上有细微差别，FedPer在本地训练时同时更新特征提取器和头部（即整个模型参数同时更新），而FedRep则是先更新头部再更新特征提取器；LG-FedAvg[25]通过聚合头部参数，将特征提取器保留在本地得到个性化模型。除了这几个较为简单的参数解耦算法，近年来还提出了许多较为复杂的参数解耦算法：联邦条件策略（federated conditional policy，FedCP）[18]由个性化特征提取器、全局头部、个性化头部以及一个条件策略网络构成个性化模型，该算法将每个样本特征中的全局信息和个性化信息分离并通过全局头部和个性化头部进行处理；全局与个性化特征学习（global and personalized feature learning，GPFL）[30]从客户端的角度，在每个客户端上同时学习全局和个性化特征信息；FedGH[16]由一个全局头部以及本地特征提取器构成个性化模型，该算法由客户端通过特征提取器将每个样本的特征向量提取出来，然后计算出平均特征向量并将其上传至服务器的全局头部中供全局头部训练；联邦鲁棒解耦（federated robust decoupling，FEDRoD）[17]中每个客户端的个性化模型都由两个头部和一个特征提取器构成，并

且使用两种损失函数，平衡了通用性和个性化需求。以上算法都是按照头部和特征提取器将模型解耦，除此之外，还有其他的解耦方式，如联邦批量归一化（federated batch normalization，FedBN）[31]保持客户端的批量归一化层在本地更新，而不上传到服务器进行聚合。还有循环蒸馏引导的通道解耦联邦个性化算法（cyclic distillation-guided channel decoupling for model personalization in federated learning，CD2-pFed）[32]与前面介绍的逐层解耦方法不同，该算法提出通道解耦，在个性化时动态解耦通道维度的参数。

其中，FedGH[16]与本文所提算法最为相似，不过FedGH只在服务器维护一个全局头部，在面对复杂的场景时，容易导致性能波动，从而无法收敛，而FedGMH分别维护各种标签的全局头部，并且在客户端上聚合全局头部得到个性化头部，适应多种场景并且提高了收敛速度。

2　FedGMH

2.1　初步准备

本文首先设置客户端总数为N个。PFL全局轮次总数设置为T，在第t轮全局轮次中（$t\in\{1,2,\cdots,T\}$），服务器按照参与率ρ随机选择客户端加入训练（$0<\rho\leqslant 1$），即每轮中有$\rho N=M$个被选定的客户端，其中M个客户端分别持有数据集$\mathcal{D}_1,\mathcal{D}_2,\cdots,\mathcal{D}_m,\cdots,\mathcal{D}_M$，并且它们的本地模型为$\boldsymbol{\omega}_1^t,\boldsymbol{\omega}_2^t,\cdots,\boldsymbol{\omega}_m^t,\cdots,\boldsymbol{\omega}_M^t$。使用$\mathcal{D}^t$来表示第$t$轮$M$个客户端的数据集之和：

$$\mathcal{D}^t\triangleq\bigcup_{m=1}^{M}\mathcal{D}_m \tag{1}$$

对于客户端m来说，定义其数据集$\mathcal{D}_m$具体为：

$$\mathcal{D}_m=\left\{\left(\boldsymbol{x}_i,y_i\right)\right\}_{i=1}^{|\mathcal{D}_m|},m\in\{1,2,\cdots,M\} \tag{2}$$

其中，$\boldsymbol{x}_i$为客户端的数据样本，y_i为对应的标签类别。

令$L_m(\boldsymbol{\omega}_m^t;\mathcal{D}_m)$为客户端$m$的损失函数，那么PFL的训练目标可以表示为：

$$\min_{\boldsymbol{\omega}_1^t,\boldsymbol{\omega}_2^t,\cdots,\boldsymbol{\omega}_m^t,\cdots,\boldsymbol{\omega}_M^t}\sum_{m=1}^{M}L_m\left(\boldsymbol{\omega}_m^t;\mathcal{D}_m\right) \tag{3}$$

本文主要符号及含义见表1。

表1　主要符号及含义

符号	含义
N	客户端总数
T	全局训练轮次总数
ρ	参与率
η	学习率
$\mathcal{D}$	数据集
$\boldsymbol{\omega}$	模型
$\boldsymbol{f}$	特征向量
$\boldsymbol{h}$	头部
$\boldsymbol{\phi}$	特征提取器
$\boldsymbol{\kappa}$	头部参数关键程度向量
$\boldsymbol{\psi}$	头部关键参数向量

2.2　FedGMH的工作流程

FedGMH的工作流程如图1所示。图1左侧展示了FedGMH客户端处理特征向量以及服务器训练多个全局头部的过程。该系统处理狗（dog）、猫（cat）和鸟（bird）的图片信息，其中$\boldsymbol{f}_1^{\text{dog}}$表示客户端1中标签为dog的特征向量，$\bar{\boldsymbol{f}}_1^{\text{dog}}$表示客户端1中标签为dog的平均特征向量。具体来说，在客户端中，其将每一样本的特征向量提取出来（步骤①），紧接着将特征向量按标签分类并计算其对应平均值（步骤②），最后将平均特征向量以及对应标签发送至服务器（步骤③）。在服务器中，服务器使用标签一致的平均特征向量训练对应的全局头部（步骤④）。

每一轮全局轮次均会发生图1所示流程。具体而言，在第t轮全局轮次中，客户端m的本地模型初始为$\boldsymbol{\omega}_m^{t,0}=\boldsymbol{h}_m^{t,0}\circ\boldsymbol{\phi}_m^{t,0}$，在本地数据集$\mathcal{D}_m$上，经过$E$轮本地轮次训练后，得到模型为$\boldsymbol{\omega}_m^{t,E}=\boldsymbol{h}_m^{t,E}\circ\boldsymbol{\phi}_m^{t,E}$，其中，$\boldsymbol{h}$是头部，$\boldsymbol{\phi}$是特征提取器，$\circ$表示模型拼接。在FedGMH中，客户端$m$需要在更新后的本地特征提取器$\boldsymbol{\phi}_m^{t,E}$上提取出样本$\boldsymbol{x}_i$

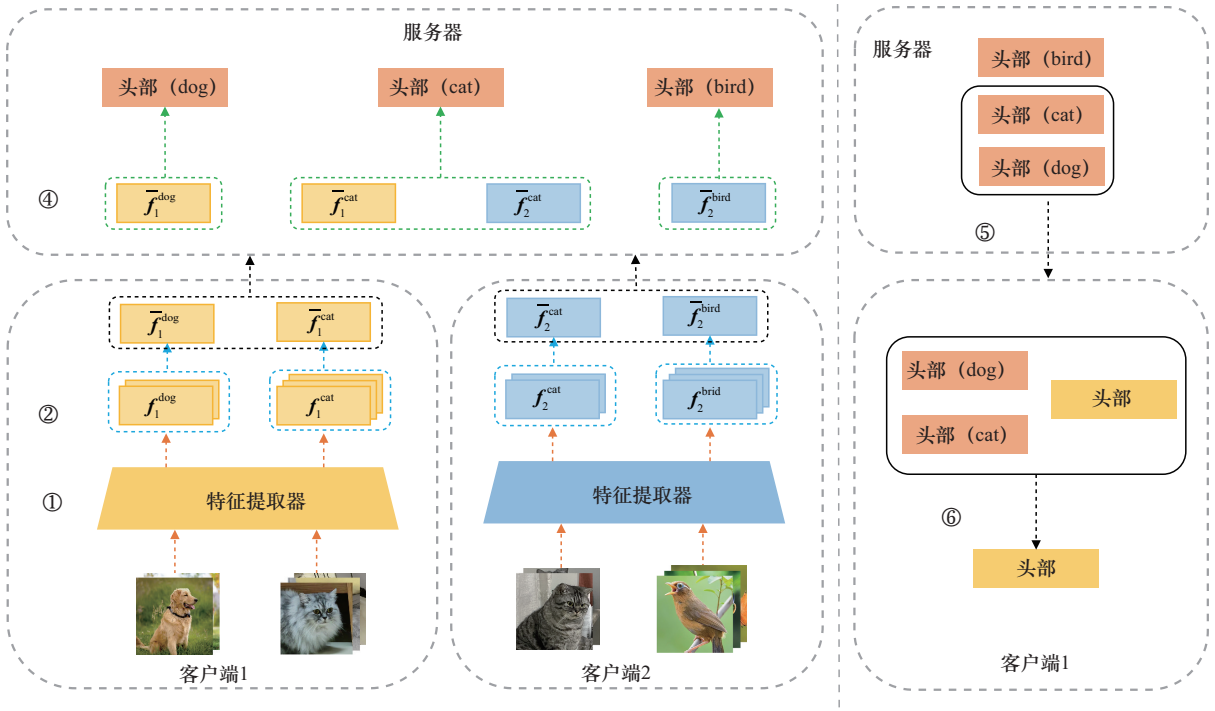

图 1　FedGMH 的工作流程

的特征向量（步骤①）：

$$\boldsymbol{f}_{m,i}^{t}=\boldsymbol{\phi}_{m}^{t,E}(\boldsymbol{x}_i) \tag{4}$$

考虑直接将特征向量$\boldsymbol{f}_{m,i}^{t}$上传至服务器可能会有隐私泄露隐患，因此将特征向量按标签分类并进行平均处理。计算同一标签类别s的样本平均特征向量$\bar{\boldsymbol{f}}_{m}^{t,s}$，具体操作为客户端$m$对于本地数据集$\left\{\left(\boldsymbol{x}_i,y_i\right)\right\}_{i=1}^{|\mathcal{D}_m|}$中所有标签类别为$s$，计算其平均特征向量（步骤②）：

$$\bar{\boldsymbol{f}}_{m}^{t,s}=\frac{1}{\left|\mathcal{D}_{m}^{s}\right|}\sum\nolimits_{i\in\mathcal{D}_{m}^{s}}\boldsymbol{f}_{m,i}^{t}=\frac{1}{\left|\mathcal{D}_{m}^{s}\right|}\sum\nolimits_{i\in\mathcal{D}_{m}^{s}}\boldsymbol{\phi}_{m}^{t,E}(\boldsymbol{x}_i) \tag{5}$$

其中，$\mathcal{D}_{m}^{s}$表示客户端m中所有标签为s的样本组成的数据集。

最后，客户端m为其每一种本地标签上传平均特征向量$\bar{\boldsymbol{f}}_{m}^{t,s}$以及对应的标签$s$到服务器（步骤③）。在 FedGMH 中，服务器同时维护多个全局头部$\{\boldsymbol{h}_1,\boldsymbol{h}_2,\cdots,\boldsymbol{h}_s,\cdots,\boldsymbol{h}_S\}$，其中$S$是全局标签总数。服务器的全局头部$\boldsymbol{h}_s$会收集所有标签$s$的平均特征向量进行训练。具体来说，在第$t$轮全局轮次中，服务器接收被选定的$M$个客户端的平均特征向量$\bar{\boldsymbol{f}}_{m}^{t,s}$。服务器使用所有平均特征向量训练对应标签的全局头部，得到新的全局头部（步骤④）：

$$\boldsymbol{h}_{s}^{t}\leftarrow\boldsymbol{h}_{s}^{t-1}-\eta_{\mathrm{h}}\nabla l\left(\boldsymbol{h}_{s}^{t-1};\bar{\boldsymbol{f}}_{m}^{t,s},s\right) \tag{6}$$

其中，η_{h}是全局头部的学习率，$l(\,;\,)$是损失函数。

在 FedGMH 客户端中，每个客户端都具有一个个性化头部，该头部由本地头部以及多个全局头部构成。如图 1 右侧所示，客户端 1 是标签为“cat”和“dog”的数据，因此客户端 1 从服务器下载处理“cat”和“dog”的全局头部参数。在 FedGMH 中，客户端从服务器下载本地标签对应全局头部（步骤⑤），然后根据这些全局头部和本地头部聚合新个性化头部（步骤⑥）。

客户端在聚合头部参数时，考虑了参数级别的影响，FedGMH 通过增加头部关键参数向量来评估每个参数是否为关键参数。假设头部$\boldsymbol{h}$的参数集合是$\{\theta_1,\theta_2,\cdots,\theta_l,\cdots,\theta_L\}$，即头部总共有$L$个

参数。在第 t 轮全局轮次时，客户端 m 在本地训练前的头部参数是 $\{\theta_{m,1}^{t,0},\theta_{m,2}^{t,0},\cdots,\theta_{m,l}^{t,0},\cdots,\theta_{m,L}^{t,0}\}$，客户端 m 在本地数据集 $\mathcal{D}_m$ 上，训练 E 个本地轮次得到新的头部参数 $\{\theta_{m,1}^{t,E},\theta_{m,2}^{t,E},\cdots,\theta_{m,l}^{t,E},\cdots,\theta_{m,L}^{t,E}\}$，因此，头部参数 $\theta_{m,l}^{t}$ 的变化程度为：

$$\left|\Delta\theta_{m,l}^{t}\right|=\left|\theta_{m,l}^{t,E}-\theta_{m,l}^{t,0}\right| \tag{7}$$

算法 FedCAC[19] 评估模型的每个参数对私有数据的关键程度，并基于此评估挑选其他客户端进行聚合。而 FedGMH 客户端在聚合个性化头部参数时采用此方法。使用 $p_{m,l}^{t}$ 来表示客户端 m 的头部参数 $\theta_{m,l}^{t}$ 的关键程度，$p_{m,l}^{t}$ 的值越大表示头部参数 $\theta_{m,l}^{t}$ 越重要，综合 FedCAC 算法，该值设置为：

$$p_{m,l}^{t}=\left|\Delta\theta_{m,l}^{t}\cdot\theta_{m,l}^{t,E}\right| \tag{8}$$

客户端 m 由此可以得到一个头部参数关键程度向量 $\boldsymbol{\kappa}_m^t=(p_{m,0}^t,p_{m,1}^t,\cdots,p_{m,l}^t,\cdots,p_{m,L}^t)$。在此设置一个超参数 $\beta\in[0,1]$（$B=\beta L$），用于确定头部参数的关键程度比例。将 $\boldsymbol{\kappa}_m^t$ 按从大到小排列，如果 $p_{m,l}^t$ 的数值大小属于前 B 名，则在长度为 L 的头部关键参数向量 $\boldsymbol{\psi}_m^t$ 对应的位置 l 上设置其元素为 true，否则设置为 false。头部关键参数向量 $\boldsymbol{\psi}_m^t$ 的元素与头部参数一一对应，如果其元素值为 true 时，表明客户端 m 在该位置头部参数为关键参数，否则为非关键参数。

综上所述，算法 1 中头部关键参数向量 $\boldsymbol{\psi}_m^t$ 的详细步骤总结如下。

算法 1　计算 $\boldsymbol{\psi}_m^t$ 详细步骤

输入　客户端 m 在本地训练前后的头部参数：$\{\theta_{m,1}^{t,0},\theta_{m,2}^{t,0},\cdots,\theta_{m,l}^{t,0},\cdots,\theta_{m,L}^{t,0}\}$ 和 $\{\theta_{m,1}^{t,E},\theta_{m,2}^{t,E},\cdots,\theta_{m,l}^{t,E},\cdots,\theta_{m,L}^{t,E}\}$；超参数 β；头部参数总数 L

遍历头部参数，根据式（7）计算 $\left|\Delta\theta_{m,l}^{t}\right|$，根据式（8）计算 $p_{m,l}^{t}$；

构建头部参数关键程度向量 $\boldsymbol{\kappa}_m^t=(p_{m,0}^t,p_{m,1}^t,\cdots,p_{m,l}^t,\cdots,p_{m,L}^t)$；

按 $p_{m,l}^t$ 降序排序 $\boldsymbol{\kappa}_m^t$；

计算关键参数数量：$B=\beta L$；

初始化 $\boldsymbol{\psi}_m^t$ 为长度 L 的全为 false 向量；

对于 $\boldsymbol{\kappa}_m^t$ 中前 B 位的 $p_{m,l}^t$，设置 $\boldsymbol{\psi}_m^t[l]=$true；

得到头部关键参数向量 $\boldsymbol{\psi}_m^t$。

在 FedGMH 中，客户端的本地头部参数为关键参数时，将更新为全局多头部参数，非关键参数将更新为本地参数。具体来说，在第 t 轮全局轮次时，客户端 m 首先从服务器接收所有本地标签 s 对应的全局头部组 $\boldsymbol{h}_s^{t-1}$。然后客户端 m 将本地头部 $\boldsymbol{h}_m^{t-1}$ 与全局头部 $\boldsymbol{h}_s^{t-1}$ 按照头部关键参数向量 $\boldsymbol{\psi}_m^{t-1}$ 构造成个性化头部，细节如下：

$$\boldsymbol{h}_m^t=\boldsymbol{h}_m^{t-1}\odot\left(\boldsymbol{\psi}-\boldsymbol{\psi}_m^{t-1}\right)+\sum\nolimits_{s\in S_m}\left(\frac{\left|\mathcal{D}_m^s\right|}{\left|\mathcal{D}_m\right|}\boldsymbol{h}_s^{t-1}\right)\odot\boldsymbol{\psi}_m^{t-1} \tag{9}$$

其中，$\boldsymbol{\psi}$ 是长度为 L 且元素全为 True 的向量，$\odot$ 是哈达玛（Hadamard）积，S_m 是客户端 m 的所有标签类别集合。

以图 1 的客户端 1 为例，图 2 展示了客户端聚合个性化头部的过程，矩阵代表整个头部，矩阵内的小方块代表头部参数，通过不同形状的方块将不同头部进行区分。图 2 左边虚线框内为聚合前所需头部，图 2 右边为聚合后的个性化头部。在客户端 1 中，头部关键参数向量 $\boldsymbol{\psi}_1$ 会标记关键参数的位置，如图 2 所示，本地头部（图 2 中五边形方块组成的矩阵）中深色的方块即关键参数，在构造个性化头部的过程中，客户端 1 下载服务器中的两个全局头部参数，并且将关键参数加权求和（以客户端 1 为例，若其 dog 类型的数据占比 1/3，而 cat 数据占比 2/3，则关键参数加权求和就是 $\frac{1}{3}\times$全局头部 dog 的关键参数 $+\frac{2}{3}\times$全局头部 cat 的关键参数），赋值到本地头部对应的位置中（即图中菱形和矩形构成的方块），而非关键参数保留原数值（即图中浅色的五边形方块），通

过这种方式，客户端获取全局信息以及个性化信息。

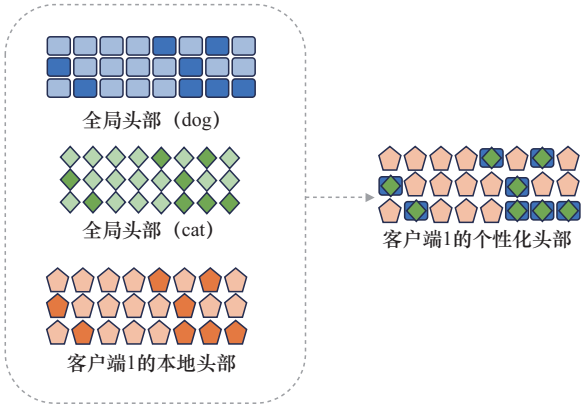

图 2　客户端聚合个性化头部的过程

FedGMH 的详细流程如下。

算法 2　FedGMH 算法

输入　客户端总数 N；参与训练的客户端数 M；全局通信的总轮次 T；全局头部学习率；标签总数 S

初始化　随机初始本地模型 $\{\boldsymbol{\omega}_1^0, \boldsymbol{\omega}_2^0, \cdots, \boldsymbol{\omega}_N^0\}$、全局头部 $\{\boldsymbol{h}_1^0, \boldsymbol{h}_2^0, \cdots, \boldsymbol{h}_s^0, \cdots, \boldsymbol{h}_S^0\}$

在第 t 轮全局通信中（$t \in \{1, 2, \cdots, T\}$），客户端 m 执行（$m \in \{1, 2, \cdots, M\}$）：

接收全局头部 $\{\boldsymbol{h}_1^{t-1}, \boldsymbol{h}_2^{t-1}, \cdots, \boldsymbol{h}_s^{t-1}, \cdots, \boldsymbol{h}_S^{t-1}\}$；

根据式（9）聚合个性化头部 $\boldsymbol{h}_m^t$；

由个性化头部和本地特征提取器组成本地模型 $\boldsymbol{\omega}_m^{t,0} = \boldsymbol{h}_m^t \circ \boldsymbol{\phi}_m^{t-1}$（本轮未被选中的客户端，本地模型使用上一轮的模型，即 $\boldsymbol{\omega}_m^t = \boldsymbol{\omega}_m^{t-1}$）；

模型在本地更新 E 轮得到 $\boldsymbol{\omega}_m^{t,E}$；

计算头部关键参数向量 $\boldsymbol{\psi}_m^t$，详见算法 1；

根据式（5）计算所有本地标签 s 对应的平均特征向量 $\bar{\boldsymbol{f}}_m^{t,s}$；

上传所有本地标签的平均特征向量 $\bar{\boldsymbol{f}}_m^{t,s}$ 以及对应的标签 s 到服务器。

服务器执行：

接收 M 个客户端的本地标签的平均特征向量 $\bar{\boldsymbol{f}}_m^{t,s}$ 以及对应的标签 s；

根据式（6）更新全局头部 $\boldsymbol{h}_s^t$；

广播全局头部集合 $\{\boldsymbol{h}_1^t, \boldsymbol{h}_2^t, \cdots, \boldsymbol{h}_s^t, \cdots, \boldsymbol{h}_S^t\}$；

每一个客户端得到个性化模型：$\{\boldsymbol{\omega}_1^T, \boldsymbol{\omega}_2^T, \cdots \boldsymbol{\omega}_N^T\}$。

2.3　讨论与分析

2.3.1　存储需求

客户端需要存储本地模型（包括特征提取器 $\boldsymbol{\phi}_m^t$ 和个性化头部 $\boldsymbol{h}_m^t$），以及一个本地的头部关键参数向量 $\boldsymbol{\psi}_m^t$。其空间复杂度为 $\mathcal{O}(|\boldsymbol{\phi}| + |\boldsymbol{h}| + |\boldsymbol{\psi}|)$，其中 $|\boldsymbol{\phi}|$ 和 $|\boldsymbol{h}|$ 分别表示特征提取器和头部模型的参数量级，$|\boldsymbol{\psi}|$ 表示 $\boldsymbol{\psi}_m^t$ 的存储需求。由于 $\boldsymbol{\psi}_m^t$ 的元素为布尔类型，其大小 $|\boldsymbol{\psi}|$ 仅与头部参数总数 L 成正比（即 $\mathcal{O}(L)$），占用的空间资源相对于模型本身（$\mathcal{O}(|\boldsymbol{\phi}| + |\boldsymbol{h}|)$）通常很小。如在 PyTorch[33] 实现的一个 4 层卷积神经网络（convolutional neural network，CNN）模型[12,34-35] 中（模型大小为 3 431.79 KB），使用 NumPy[36] 存储 $\boldsymbol{\psi}_m^t$ 向量通常只需 5.01 KB。在聚合个性化头部时，客户端需要临时存储其本地标签 S_m 对应的所有全局头部参数。这带来了额外的存储开销 $\mathcal{O}(|S_m| \cdot |\boldsymbol{h}|)$。假设在极端情况下（如 Cifar100[37]），一个客户端拥有 100 种标签类别（$|S_m| = 100$），则该客户端需要 $\mathcal{O}(100 \cdot |\boldsymbol{h}|)$ 的额外空间。虽然这增加了存储负担（如前述 CNN 示例中将增加约 58.54% 的空间），但与其他算法中将模型扩展成两个模型[18]（增加 100% 额外空间）或维护多个特征提取器[38]（增加 99.42% 额外空间）相比，FedGMH 的存储开销更具优势。

2.3.2　通信开销

在客户端到服务器的上行通信中，客户端需要将所有本地标签的平均特征向量 $\bar{\boldsymbol{f}}_m^{t,s}$ 以及对应的标签 s 上传至服务器，一个平均特征向量 $\bar{\boldsymbol{f}}_m^{t,s}$ 的维度等于特征提取器的输出维度 d_{f}（即 $|\bar{\boldsymbol{f}}_m^{t,s}| = d_{\mathrm{f}}$）。

因此，客户端m的上行复杂度为$\mathcal{O}\left(|S_m|\cdot d_{\mathrm{f}}\right)$。以上述CNN模型为例，$d_{\mathrm{f}}=512$，一个平均特征向量需要2 KB（浮点数按4 byte计算）。考虑最极端情况（$|S_m|=100$），该客户需要上传200.39 KB的数据至服务器，其中0.39 KB为标签类别。

在服务器到客户端的下行通信中，服务器将客户端所有的本地标签s对应的全局头部$\boldsymbol{h}_s^t$广播给客户端。因此，客户端m的下行通信复杂度为$\mathcal{O}\left(|S_m|\cdot|\boldsymbol{h}|\right)$。即使在最极端情况下（$|S_m|=100$），客户端只需要下载约2 004 KB的数据。

在FedGMH中，对客户端来说，其上传和下载的通信成本通常低于传输整个模型（复杂度为$\mathcal{O}\left(|\boldsymbol{\omega}|\right)$，传输数据量为3 431.79 KB，如FedAvg[12]），或者特征提取器（复杂度为$\mathcal{O}\left(|\boldsymbol{\phi}|\right)$，传输数据量为3 411.75 KB，如FedPer[10]、FedRep[11]）。

2.3.3　计算开销

FedGMH在客户端引入了如下额外的计算步骤。

（1）向量提取与平均（图1中的步骤①、②）：对每个本地样本$\boldsymbol{x}_i\in\mathcal{D}_m$计算特征向量$\boldsymbol{f}_{m,i}^t$复杂度为$\mathcal{O}\left(|\mathcal{D}_m|\cdot C_{\boldsymbol{\phi}}\right)$，其中$C_{\boldsymbol{\phi}}$是特征提取器前向传播的计算复杂度。随后按标签分组并计算平均特征向量$\bar{\boldsymbol{f}}_m^{t,s}$的复杂度为$\mathcal{O}\left(|\mathcal{D}_m|+|S_m|\cdot d_{\mathrm{f}}\right)$。因此，此部分总复杂度约为$\mathcal{O}\left(|\mathcal{D}_m|\cdot C_{\boldsymbol{\phi}}\right)$。

（2）关键参数评估（算法1）：计算每个头部参数的变化量$\left|\Delta\theta_{m,l}^t\right|$和关键程度$p_{m,l}^t$，然后排序并生成$\boldsymbol{\psi}_m^t$向量的复杂度为$\mathcal{O}(L)$。

（3）个性化头部聚合（图1中步骤⑥）：根据$\boldsymbol{\psi}_m^{t-1}$向量，对关键参数位置进行加权求和，复杂度为$\mathcal{O}\left(|S_m|\cdot L\right)$。

与FedAvg的标准本地训练相比（复杂度为$\mathcal{O}\left(|\mathcal{D}_m|\cdot C_{\boldsymbol{\omega}}\right)$，$C_{\boldsymbol{\omega}}$是整个模型前向+反向传播的计算复杂度），FedGMH增加了$\mathcal{O}\left(|\mathcal{D}_m|\cdot C_{\boldsymbol{\phi}}+L+|S_m|\cdot L\right)$的额外计算开销。$L$和$|S_m|$通常远小于$|\mathcal{D}_m|$，主要额外开销在于特征向量提取（相当于一次额外的前向传播），这一部分消耗本地训练一个周期中的一半计算成本，通常本地训练周期设置是大于一个轮次的，以避免频繁与服务器通信。在提高收敛速度的情况下，这一部分开销是可以接受的。

2.3.4　隐私保护

在FedGMH中，客户端m的本地头部$\boldsymbol{h}_m^t$和本地特征提取器$\boldsymbol{\phi}_m^t$均保留在本地，通过网络传输的只有平均特征向量$\bar{\boldsymbol{f}}_m^{t,s}$以及对应标签类别$s$和全局头部$\boldsymbol{h}_s^t$，所以如果在传输过程中这些数据被截取，也无法通过这些数据将客户端的完整模型或者原数据样本复原出来。因此，FedGMH具有高水平的隐私保护技术，而且还可以结合现有的隐私保护机制进一步提高该算法的安全性。

2.3.5　动态适应

FedGMH的全局头部数量严格等于全局标签总数，在训练过程中固定不变。这种设计能够适应客户端本地标签的动态变化。当客户端本地出现新标签时，只需要在聚合个性化头部时下载对应的全局头部即可。服务器全局头部已覆盖所有潜在标签，无须动态增减。当客户端删除某种标签时，对于客户端来说，只需要下载其他标签的全局头部即可，不影响本地头部的更新和其他标签数据的学习；对于服务器来说，仅减少了该标签的一部分本地头部的样本，不会严重影响全局头部的学习。

3　实验与分析

3.1　实验设置

本文的实验主要在图像分类任务上评估FedGMH和先前的先进联邦学习算法（包括TFL算法和PFL算法），使用了3个流行的数据集，包括MNIST[39]、Cifar10[37]、Cifar100[37]。对于模型，

本实验使用著名的4层CNN[12,34-35]，并且设置了本地学习率$\eta=0.01$，学习率每轮衰减1%。为了模拟现实数据异构场景，本实验使用了扩展的狄利克雷策略[15]。对于MNIST和Cifar10数据集，将采用ExDir(2,0.5)和ExDir(2, 5)，对于Cifar100数据集将采用ExDir(5,0.5)和ExDir(5, 5)，其中，α越小表示数据异质性越高，即$\alpha=0.5$比$\alpha=5$异质性高。

在本实验中，默认将数据集的所有数据按扩展的狄利克雷策略分配给100个客户端，然后，将每个客户端上的数据分割成训练数据集（75%）和测试数据集（25%）。本地训练批次大小设置为100，本地的训练轮次设置为2轮。对于MNIST和Cifar10数据集，全局轮次为200轮，而对于Cifar100数据集，全局轮次为300轮。客户端的参与率设置为$\rho=20\%$。在FedGMH算法中，超参数β设置为0.5，η_h设置为1.0。在每轮全局通信结束时，对于PFL算法，每个客户端测量其个性化模型在测试数据集上的准确率，对于TFL算法，每个客户端测量全局模型在测试数据集上的准确率，然后计算所有客户端在本轮的平均准确率。本文在5种不同的随机种子下运行所有实验，并记录每次实验中所有全局轮次下的最佳测试准确率。

本实验中将FedGMH算法与其他TFL算法和PFL算法进行比较。TFL算法有FedAvg[12]和FedProx[20]，PFL算法有Ditto[21]、FedAMP[29]、FedCP[18]、FedFomo[28]、FedGH[16]、FedPer[10]、FedRep[11]、FedRoD[17]和LG-FedAvg[25]。

3.2 准确率实验

实验运行FedGMH和其他联邦学习算法，主实验中图像分类任务的准确率见表2。

根据表2可知，FedGMH算法在所有情况下准确率均为最高，因此该算法优于其他联邦学习算法。在Cifar100数据集下，FedGH算法总是很难收敛，准确率出现波动，这是因为该算法只设置一个全局头部，每轮被选中的客户端需要上传平均特征向量至服务器中供该全局头部训练，而在异质性高（$\alpha=0.5$）的场景下，单一的全局头部难以收敛。这种情况与FedAvg类似，FedAvg将每轮被选中客户端的模型上传至服务器，由服务器聚合这些模型得到单一全局模型，因此在异质性高的场景下，FedAvg难以收敛。而FedGMH算法将单一的全局头部扩展到与标签类别数量一

表2 主实验中图像分类任务的准确率

算法	MNIST		Cifar10		Cifar100	
	ExDir(2,0.5)	ExDir(2, 5)	ExDir(2,0.5)	ExDir(2, 5)	ExDir(5,0.5)	ExDir(5, 5)
FedAvg	88.30%±0.95%	92.28%±0.72%	31.30%±1.05%	35.91%±0.82%	9.73%±0.39%	10.86%±0.42%
FedProx	88.52%±1.21%	92.25%±0.76%	30.73%±1.00%	35.80%±0.75%	9.90%±0.43%	10.87%±0.17%
Ditto	95.96%±0.12%	96.98%±0.18%	91.06%±0.09%	77.06%±0.39%	71.41%±0.51%	55.45%±0.81%
FedAMP	97.18%±0.09%	98.26%±0.10%	92.11%±0.13%	80.33%±0.45%	76.03%±0.40%	61.59%±0.38%
FedCP	98.20%±0.15%	98.43%±0.15%	92.00%±0.13%	81.95%±0.21%	73.85%±0.46%	61.17%±0.31%
FedFomo	98.20%±0.10%	98.40%±0.16%	93.10%±0.10%	79.74%±0.46%	73.70%±0.42%	59.13%±0.36%
FedGH	89.73%±0.96%	98.21%±0.07%	79.26%±1.40%	82.11%±0.33%	58.86%±2.31%	56.13%±0.74%
FedPer	98.42%±0.08%	98.33%±0.22%	92.19%±0.17%	82.05%±0.45%	74.76%±0.50%	61.72%±0.56%
FedRep	97.94%±0.22%	98.24%±0.20%	91.67%±0.31%	80.63%±0.46%	72.14%±0.65%	59.25%±0.62%
FedRoD	98.18%±0.06%	98.45%±0.18%	91.69%±0.17%	81.13%±0.23%	72.45%±0.31%	59.13%±0.54%
LG-FedAvg	97.61%±0.14%	98.27%±0.15%	92.23%±0.15%	80.97%±0.33%	74.30%±0.41%	59.95%±0.51%
FedGMH	**98.66%±0.15%**	**98.75%±0.06%**	**93.92%±0.15%**	**82.50%±0.34%**	**77.47%±0.07%**	**62.99%±0.41%**

样多的全局头部，并且将与标签对应的特征向量上传至相应的全局头部上进行训练，解决了单一全局头部在数据异质性高情况下收敛性差的难题。

FedGMH 与 9 种基准算法在 Cifar100 上的性能如图 3 所示。在图 3 中可以明显看到，FedGH 算法无法收敛，准确率一直波动。而 FedGMH 算法表现最佳。在达到 70% 准确率的目标时，FedGMH 仅需 68 轮，显著快于对比算法中最优的 FedPer（108 轮）和 FedAMP（120 轮）。在更严格的 75% 准确率目标下，FedGMH 仅需 185 轮即可达到，而 FedAMP 和 FedPer 分别需要 240 轮和 297 轮，均超过 200 轮。这些结果充分验证了 FedGMH 优异的快速收敛能力。

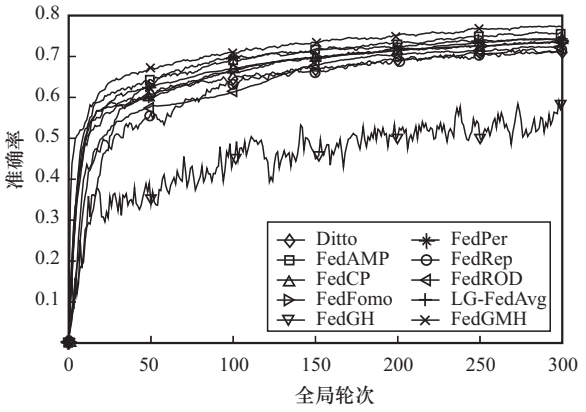

图 3　FedGMH 与 9 种基准算法在 Cifar100 上的性能

3.3　不同的本地轮次

在联邦学习中，更多的本地轮次能减少总的通信轮次，不过这会增加客户端的计算成本[12]。本文设置不同的本地轮次 $E=\{4,6,8,10\}$，并在 ExDir(2,0.5)分布下的 Cifar10 数据集上完成实验，在 Cifar10 上使用不同本地轮次的准确率见表 3，其中本地轮次 $E=2$ 的实验结果见表 2。

由表 3 可知，除了 Ditto 算法，其他联邦学习算法均在更多的本地轮次下得到更高的准确率，即可以在相同准确率下减少全局通信轮次总数。其中 TFL 算法以及 FedGH 的准确率提升最明显，这说明对于维护单一全局模型（或单一全局头部）的算法，在数据异质性高的情况下，增加其本地轮次能有效缓解其收敛差的难题。在不同的本地轮次下，FedGMH 算法的最佳准确率均为最高，因此该算法比其他联邦学习算法优越。

3.4　不同的参与率

本文考虑服务器的调度能力，即服务器每轮能使得多少客户端参与模型训练，设置不同的参与率 $\rho=\{40\%,60\%,80\%,100\%\}$，并在 ExDir(5,0.5) 分布下的 Cifar100 数据集完成本次实验。在 Cifar100 上使用不同参与率的准确率见表 4。TFL 在

表 3　在 Cifar10 上使用不同本地轮次的准确率

算法	不同本地轮次准确率			
	4	6	8	10
FedAvg	34.40%±1.22%	37.34%±0.81%	39.01%±0.63%	40.67%±0.60%
FedProx	34.60%±0.97%	37.48%±1.56%	39.51%±0.44%	40.53%±1.06%
Ditto	91.06%±0.13%	91.01%±0.15%	90.98%±0.12%	91.11%±0.09%
FedAMP	93.06%±0.14%	93.50%±0.14%	93.71%±0.15%	93.83%±0.14%
FedCP	93.20%±0.15%	93.98%±0.20%	94.31%±0.11%	94.71%±0.11%
FedFomo	94.10%±0.08%	94.53%±0.09%	94.78%±0.10%	94.95%±0.13%
FedGH	80.52%±3.47%	82.82%±1.46%	83.06%±1.16%	86.42%±1.19%
FedPer	93.24%±0.19%	93.82%±0.22%	94.16%±0.14%	94.42%±0.20%
FedRep	92.25%±0.27%	93.23%±0.17%	93.54%±0.20%	93.69%±0.25%
FedRoD	92.96%±0.15%	93.94%±0.01%	94.28%±0.03%	94.52%±0.05%
LG-FedAvg	93.07%±0.13%	93.35%±0.12%	93.62%±0.06%	93.81%±0.09%
FedGMH	**94.78%±0.20%**	**95.08%±0.19%**	**95.16%±0.18%**	**95.16%±0.22%**

表 4　在Cifar100上使用不同参与率的准确率

算法	参与率			
	40%	60%	80%	100%
FedAvg	9.47%±0.40%	9.03%±0.37%	8.46%±0.39%	7.82%±0.46%
FedProx	9.52%±0.49%	9.02%±0.44%	8.46%±0.45%	7.81%±0.50%
Ditto	74.90%±0.34%	76.19%±0.29%	76.82%±0.26%	77.08%±0.23%
FedAMP	79.04%±0.12%	80.04%±0.18%	80.43%±0.13%	80.61%±0.14%
FedCP	75.66%±0.58%	76.91%±0.46%	77.44%±0.37%	77.99%±0.22%
FedFomo	76.78%±0.16%	77.68%±0.14%	78.18%±0.16%	78.33%±0.23%
FedGH	66.60%±1.23%	69.17%±0.54%	72.46%±1.12%	76.62%±0.35%
FedPer	75.35%±0.33%	75.10%±0.30%	74.78%±0.31%	74.40%±0.35%
FedRep	73.93%±0.28%	73.93%±0.34%	73.53%±0.34%	73.12%±0.33%
FedRoD	71.84%±0.75%	70.44%±0.98%	68.61%±1.04%	66.90%±0.96%
LG-FedAvg	77.19%±0.26%	78.09%±0.19%	78.46%±0.14%	78.61%±0.11%
FedGMH	**79.73%±0.18%**	**80.56%±0.23%**	**80.74%±0.12%**	**80.71%±0.16%**

更大的参与率下，测试准确率不断下降，这是因为每轮全局训练中更多的非独立同分布数据的加入，导致单一全局模型更难捕捉和适应这种数据多样性，从而导致性能下降。在本实验中，观察到随着参与率的增加，FedRoD的准确率呈现出不断下降的趋势，主要是因为该算法采用两个头部结构，其两个头部分别学习全局信息以及个性化信息，而且两个头部之间具有竞争性，导致其在更多客户端参与下收敛性更差。在所有情况下，FedGMH算法的最佳准确率优于其他联邦学习算法。

3.5　消融研究

在FedGMH中，主要的模块有全局多头部模块以及按关键参数位置聚合头部模块。本文将全局多头部模块取消，设置为一个全局头部，将其命名为FedGMH-onehead（缩写为GMH-o）。

FedGMH根据式（9）在客户端m上可聚合个性化头部得到$\boldsymbol{h}_m^t$。每个客户端需要维护一个头部关键参数向量$\boldsymbol{\psi}_m^t$，并且对于头部关键参数的位置将其更新为全局多头部的聚合参数，将非关键参数的位置更新为本地头部参数。将这种按关键参数位置聚合方式取消，设置为将本地头部与全局多头部的简单聚合，即由式（10）聚合得到个性化头部，并将其命名为FedGMH-avghead（缩写为GMH-a）。

$$\boldsymbol{h}_m^t=\frac{1}{2}\left(\boldsymbol{h}_m^{t-1}+\sum\nolimits_{s\in S_m}\left(\frac{\left|\mathcal{D}_m^s\right|}{\left|\mathcal{D}_m\right|}\boldsymbol{h}_s^{t-1}\right)\right) \tag{10}$$

将FedGMH的两个模块都取消，只保留一个全局头部，并且在客户端聚合时，只有本地头部与全局头部进行聚合。由式（11）聚合得到个性化头部，其中$\boldsymbol{h}^{t-1}$是全局头部。并且将其命名为FedGMH-oneavg（该变形与FedGH不同，FedGH没有聚合全局头部和本地头部的步骤，缩写为GMH-oa）。

$$\boldsymbol{h}_m^t=\frac{1}{2}\left(\boldsymbol{h}_m^{t-1}+\boldsymbol{h}^{t-1}\right) \tag{11}$$

为了展示FedGMH模块的有效性，在ExDir(2,0.5)分布下的MNIST和Cifar10数据集上，以及ExDir(5,0.5)分布下的Cifar100数据集上完成消融研究，其余设置均为默认实验的设置。在3个数据集上进行消融研究的准确率见表5。

在这3个数据集下，GMH-o的准确率总是比GMH-a低，说明全局多头部模块比按关键参数位置聚合头部模块更加重要，全局多头部模块更能有效提高准确率；而GMH-oa的准确率是最低的，比GMH-o和GMH-a都低，说明全局多头部模块以及按关键参数位置聚合头部模块两个模块

表 5　在 3 个数据集上进行消融研究的准确率

算法	数据集		
	MNIST	Cifar10	Cifar100
GMH-o	97.79%±0.05%	92.60%±0.15%	73.13%±0.47%
GMH-a	98.32%±0.15%	93.60%±0.16%	76.33%±0.38%
GMH-oa	96.27%±0.48%	86.24%±1.06%	63.34%±1.38%
FedGMH	**98.66%±0.15%**	**93.92%±0.15%**	**77.47%±0.07%**

之间是相互促进的。

图 4 展示了在 Cifar100 数据集上，3 种变形算法与 FedGMH 的准确率变化情况。

如图 4 所示，GMH-o 由于没有多个全局头部而在一开始难以收敛，直到全局轮次接近 300 轮时才开始收敛，需要大量的全局通信才能收敛。而 GMH-a 没有按关键参数位置进行聚合，所以在收敛速度上比原算法慢一点，并且准确率也稍微低一些。GMH-oa 只有一个全局头部，并且没有按照关键参数位置聚合头部参数，导致其难以收敛。

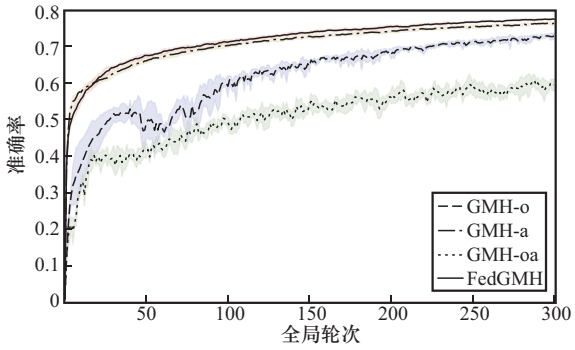

图 4　Cifar100 数据集上 3 种变形算法与 FedGMH 的准确率变化情况

3.6　超参数 β 实验

在 FedGMH 中，超参数 β 控制着客户端在本地聚合个性化头部参数的关键程度比例，该参数值越大，表示全局多头部在本地聚合时占用的位置越多，而本地头部占用的位置越少。该实验在 ExDir(5,0.5)分布下的 Cifar100 数据集上完成，设置超参数 $\beta=\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$。在 Cifar100 上使用不同超参数 β 的准确率见表 6，β 的大小对于最佳准确率来说影响较小。无论 β 的值如何变化，FedGMH 的准确率始终高于 GMH-a 的 76.33%，即按关键参数位置聚合头部模块始终比简单的平均聚合头部更有效地提高准确率（β 在该实验中影响较小的主要原因是 FedGMH 的参数级聚合方式本身具有较强的稳定性，关键参数的评估和更新机制已能较好地平衡全局和本地信息，使得 β 在一定范围内变化时，模型性能不会出现大幅波动）。

表 6　在 Cifar100 上使用不同超参数 β 的准确率

超参数 β	准确率
0.1	77.76%±0.19%
0.2	77.52%±0.39%
0.3	77.60%±0.22%
0.4	77.56%±0.22%
0.5	77.47%±0.07%
0.6	77.48%±0.30%
0.7	77.56%±0.32%
0.8	77.42%±0.27%
0.9	77.62%±0.24%
1.0	77.43%±0.25%

3.7　超参数 η_h 实验

在 FedGMH 中，超参数 η_h 为全局头部学习率。在 ExDir(5,0.5)分布下的 Cifar100 数据集上完成本次实验，设置超参数 $\eta_h=\{0.01, 0.05, 0.1, 0.5, 1.0\}$。在 Cifar100 上使用不同超参数的准确率见表 7，准确率随 η_h 增大而增大，这是因为客户端上传至服务器的特征向量为平均特征向量，因此服务器用来训练全局头部的特征向量总数较少，所以需要较大的学习率来加快全局头部的训练过程。在该场景下，FedGMH 的最低准确率为 76.06%，比基准 FedAMP 的最高准确率 76.03% 还高。

表 7　在 Cifar100 上使用不同超参数 η_h 的准确率

超参数 η_h	准确率
0.01	76.06%±0.30%
0.05	76.27%±0.25%
0.10	76.51%±0.25%
0.50	77.07%±0.30%
1.00	77.47%±0.07%

4 结束语

本文使用拓展的狄利克雷策略来模拟现实的异构场景，在少量标签类别的情况下，所提FedGMH通过服务器训练多个全局头部，每个全局头部专注于对应标签的全局信息。客户端通过聚合本地标签对应的全局头部和本地头部来处理标签全局信息以及个性化信息。在3个流行的数据集上的实验结果表明，FedGMH的准确率均优于现有先进的PFL算法，这说明了FedGMH采用的全局多头部分别处理标签全局信息的有效性。

参考文献：

[1] Islam T U, Ghasemi R, Mohammed N. Privacy-preserving federated learning model for healthcare data[C]//Proceedings of the 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC). Piscataway: IEEE Press, 2022: 281-287.

[2] Sheller M J, Edwards B, Reina G A, et al. Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data[J]. Scientific Reports, 2020, 10: 12598.

[3] Sadilek A, Liu L Y, Nguyen D, et al. Privacy-first health research with federated learning[J]. NPJ Digital Medicine, 2021, 4: 132.

[4] Awosika T, Shukla R M, Pranggono B. Transparency and privacy: the role of explainable AI and federated learning in financial fraud detection[J]. IEEE Access, 2024, 12: 64551-64560.

[5] Yang W S, Zhang Y H, Ye K J, et al. FFD: a federated learning based method for credit card fraud detection[C]//Big Data-BigData 2019. Cham: Springer, 2019: 18-32.

[6] Ferdous Aurna N, Delwar Hossain M, Khan L, et al. FedFusion: adaptive model fusion for addressing feature discrepancies in federated credit card fraud detection[J]. IEEE Access, 2024, 12: 136962-136978.

[7] Liu L, Tian Y X, Chakraborty C, et al. Multilevel federated learning-based intelligent traffic flow forecasting for transportation network management[J]. IEEE Transactions on Network and Service Management, 2023, 20(2): 1446-1458.

[8] Liu T B, Wang Y, Zhou H Y, et al. Distributed short-term traffic flow prediction based on integrating federated learning and TCN[J]. IEEE Access, 2024, 12: 148026-148036.

[9] 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.

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

[11] Collins L, Hassani H, Mokhtari A, et al. Exploiting shared representations for personalized federated learning[PP]. arXiv preprint, 2021, arXiv: 2102.07078.

[12] Mcmahan H B, Moore E, Ramage D, et al. Communication-efficient learning of deep networks from decentralized data[C]//Proceedings of the International Conference on Artificial Intelligence and Statistics, 2016

[13] Li Q B, Diao Y Q, Chen Q, et al. Federated learning on non-IID data silos: an experimental study[C]//Proceedings of the 2022 IEEE 38th International Conference on Data Engineering (ICDE). Piscataway: IEEE Press, 2022: 965-978.

[14] Yurochkin M, Agarwal M, Ghosh S, et al. Bayesian nonparametric federated learning of neural networks[PP]. arXiv preprint,2019, arXiv: 1905.12022.

[15] Li Y P, Lyu X C. Convergence analysis of sequential split learning on heterogeneous data[PP]. arXiv preprint, 2023, arXiv: 2302.01633.

[16] Yi L P, Wang G, Liu X G, et al. FedGH: heterogeneous federated learning with generalized global header[C]//Proceedings of the 31st ACM International Conference on Multimedia. New York: ACM Press, 2023: 8686-8696.

[17] Chen H Y, Chao W L. On bridging generic and personalized federated learning for image classification[PP]. arXiv preprint, 2021, arXiv:2107.00778.

[18] Zhang J Q, Hua Y, Wang H, et al. FedCP: separating feature information for personalized federated learning via conditional policy[C]//Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. New York: ACM Press, 2023: 3249-3261.

[19] Wu X H, Liu X F, Niu J W, et al. Bold but cautious: unlocking the potential of personalized federated learning through cautiously aggressive collaboration[C]//Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision (ICCV). Piscataway: IEEE Press, 2023: 19318-19327.

[20] Li T, Sahu A K, Zaheer M, et al. Federated optimization in heterogeneous networks[PP] arXiv preprint, 2020, arXiv: 1812.06127.

[21] Li T, Hu S Y, Beirami A, et al. Ditto: fair and robust federated learning through personalization[PP]. arXiv preprint, 2020, arXiv:2012.04221.

[22] Dinh C T, Tran N H, Nguyen J D. Personalized federated learning with Moreau envelope[PP]. arXiv preprint, 2020, arXiv: 2006.08848.

[23] Chen F, Luo M, Dong Z H, et al. Federated meta-learning with fast convergence and efficient communication[PP]. arXiv preprint, 2019, arXiv:1802.07876,

[24] Fallah A, Mokhtari A, Ozdaglar A E. Personalized federated learning with theoretical guarantees: a model-agnostic meta-learning approach[C]//Advances in Neural Information Processing Systems. New York: Curran Associates, Inc, 2020: 3557-3568.

[25] Liang P P, Liu T, Liu Z Y, et al. Think locally, act globally: federated learning with local and global representations[PP]. arXiv preprint, 2020, arXiv:2001.01523.

[26] Deng Y Y, Kamani M M, Mahdavi M. Adaptive personalized federated learning[PP]. arXiv preprint, 2020,arXiv: 2003.13461.

[27] Zhang J Q, Hua Y, Wang H, et al. FedALA: adaptive local aggregation for personalized federated learning[J]. Proceedings of the AAAI Conference on Artificial Intelligence, 2023, 37(9): 11237-11244.

[28] Zhang M, Sapra K, Fidler S, et al. Personalized federated learning with first order model optimization[PP]. arXiv preprint, 2020, arXiv: 2012.08565.

[29] 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.

[30] Zhang J Q, Hua Y, Wang H, et al. GPFL: simultaneously learning global and personalized feature information for personalized federated learning[C]//Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision (ICCV). Piscataway: IEEE Press, 2023: 5018-5028.

[31] Li X X, Jiang M R, Zhang X F, et al. FedBN: federated learning on non-IID features via local batch normalization[PP]. arXiv preprint, 2021, arXiv:2102.07623.

[32] Shen Y Q, Zhou Y Y, Yu L Q. CD2-pFed: cyclic distillation-guided channel decoupling for model personalization in federated learning[C]//Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Piscataway: IEEE Press, 2022: 10031-10040.

[33] Paszke A, Gross S, Massa F, et al. PyTorch: an imperative style, high-performance deep learning library[C] //Advances in Neural Information Processing Systems. New York: Curran Associates, Inc, 2019: 8024-8035.

[34] Geiping J, Bauermeister H, Dröge H, et al. Inverting gradients-how easy is it to break privacy in federated learning [C]//Advances in Neural Information Processing Systems. New York: Curran Associates, Inc, 2020: 16937-16947.

[35] Luo M, Chen F, Hu D P, et al. No fear of heterogeneity: classifier calibration for federated learning with non-IID data[C] // Advances in Neural Information Processing Systems. New York: Curran Associates, Inc, 2021: 5972-5984.

[36] Harris C R, Millman K J, Van Der Walt S J, et al. Array programming with NumPy[J]. Nature, 2020, 585(7825): 357-362.

[37] Krizhevsky A, Hinton G. Learning multiple layers of features from tiny images[J]. Handbook of Systemic Autoimmune Diseases, 2009, 1(4).

[38] Gao L, Li Z X, Wu C, et al. FediOS: decoupling orthogonal subspaces for personalization in feature-skew federated learning[PP]. arXiv preprint, 2023, arXiv:2311.18559.

[39] Lecun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[作者简介]

孙圳峰（2000– ），男，浙江工商大学信息与电子工程学院硕士生，主要研究方向为联邦学习。

倪郑威（1989– ），男，博士，浙江工商大学信息与电子工程学院副研究员、硕士生导师，主要研究方向为人工智能、无线网络。