移动边缘计算中通信高效的联邦学习模型剪枝算法

胡海峰[1]，张熙[1]，赵海涛[2]，吴建盛[3]
（1. 南京邮电大学通信与信息工程学院，江苏 南京 210003；2. 南京邮电大学物联网学院，江苏 南京 210003；
3. 南京邮电大学计算机学院，江苏 南京 210023）

摘　要：移动边缘计算中，边缘端服务器和移动终端利用联邦学习分布式架构构建深度模型，使终端之间无须共享数据就可以协作训练，然而深度模型训练需要在服务器和多个客户终端之间进行多轮通信传输，需要消耗大量的通信资源和训练开销。针对这个问题，提出了一种通信高效的联邦学习模型剪枝（CEMP-FL, communication-efficient model pruning for federated learning）架构，服务器运行单次层平衡网络剪枝（SBNP, single-shot layer balance network pruning）算法，通过粗剪枝和精细剪枝的组合，并结合非结构化稀疏参数压缩，显著减少了通信过程中传输的深度模型参数量，并有效地减少了终端侧训练样本分布差异带来的剪枝偏差。同时，使用网络剪枝的层平衡策略（LBP, layer balance policy），确保了深度模型层之间的参数量平衡，在稀疏度很大的情况下有效地推迟了深度模型坍塌。最后，基于两种基准数据集讨论了 CEMP-FL 在无线场景中的性能，实验表明，提出的 CEMP-FL 在保证性能的前提下取得了最优的通信成本压缩比，实现了联邦学习分布式训练架构下的高效通信。

关键词：联邦学习；剪枝算法；通信效率；层平衡
中图分类号：TP301
文献标志码：A
doi：10.11959/j.issn.2096−3750.2024.00392

Communication-efficient model pruning for federated learning in mobile edge computing

HU Haifeng[1], ZHANG Xi[1], ZHAO Haitao[2], WU Jiansheng[3]
1. School of Communications and Information Engineering, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
2. School of Internet of Things, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
3. School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210023, China

Abstract: In the mobile edge computing scenario, the distributed architecture of federated learning allows the edge server and mobile terminals to cooperatively train the deep model, without necessitating sharing of local data across the mobile terminals. While the training process generally consists of multiple rounds between the server and several clients, which can incur high communication costs and training overhead. To address this issue, a communication-efficient model pruning for federated learning (CEMP-FL) framework, which employed the single-shot layer balance network pruning (SBNP) algorithm, combined with unstructured sparse weight compression, was proposed to significantly reduce the size of the global model, and to effectively alleviate the biased pruning due to training samples discrepancy between clients. Meanwhile, layer balance policy (LBP) was adopted to ensure a balance of the model parameters between layers, which could effectively circumvent the problem of layer-collapse in the case of high sparsity. Finally, the performance of CEMP-FL in wireless scenarios was discussed on two benchmark datasets. The experimental results show that the proposed

收稿日期：2023−09−15；**修回日期：**2024−06−07
通信作者：赵海涛，zhaoht@njupt.edu.cn
基金项目：国家自然科学基金项目（No. 62071242，No. 61571233，No. 61901229，No. 61872198，No. 62371245）
Foundation Items: The National Natural Science Foundation of China (No. 62071242, No. 61571233, No. 61901229, No. 61872198, No. 62371245)

CEMP-FL method achieves the highest compression ratio of communication costs while maintaining performance, and provides efficient communication in the distributed architecture of federated learning.

Key words: federated learning, pruning algorithm, communication-efficiency, layer balance

0　引言

移动边缘计算将计算任务从云端转移到无线网络边缘端，并利用无线接入网络提供高带宽、低时延的业务场景。考虑移动终端自身存储和计算能力不断地提高，联邦学习分布式学习架构中，终端之间无须共享数据，通过和边缘服务器交互协作以训练深度模型，从而可以在服务器侧快速部署深度模型，以提供各种面向人工智能的应用服务。

近年来，联邦学习得到了学术界和工业界的广泛研究[1-3]，但联邦学习在应用过程中尚存在一些挑战[1]。移动边缘计算场景中，联邦学习的通信效率问题显得尤为突出，移动终端和边缘服务器通过无线信道交互模型的参数，以得到训练服务器侧的深度模型。而客户端数据分布的差异性导致服务器端和客户端需要较多的通信轮次达到模型的收敛，频繁进行模型参数交互带来很大的通信开销，通信开销进一步增加无线网络的能量消耗和模型的训练时延，因此，如何在联邦学习交互过程中提高通信效率、降低通信开销显得尤为重要。

移动边缘计算场景中，提高联邦学习通信效率是指在保证深度模型性能的前提下减少训练过程中模型参数交互的大小[4]。早期采用的方法是客户端上传模型前使用有损压缩进行模型编码，在服务器端进行相应的解码并聚合模型，这种方法增加了计算开销，并且有损压缩导致训练过程中模型参数传递产生误差。另一种方法是联邦集成学习[5]，使用公共数据集预训练分类器，并利用集成学习提升模型的收敛速度，以达到减少通信开销的目的。然而，实际的分布式数据场景中，在服务器端使用大规模公共数据集进行预训练很难实现。近年来，由于剪枝算法[6-7]可以对深度模型的冗余结构进行删减，并不会带来性能的明显降低，在联邦学习中引入剪枝算法成为提升通信效率的有效途径。但现有的剪枝算法对网络结构非常敏感，因此，每次进行一定比例的剪枝后，必须重新用训练数据集进行训练，即采用剪枝、重新训练和再剪枝、再重新训练的循环策略。这种方法如果直接用到联邦学习架构中，每次剪枝都需要模型重新训练，必然会带来额外的多轮无线通信，所以，必须对剪枝算法在联邦学习架构下进行改进以提高通信效率。

针对联邦学习通信效率、客户端训练数据分布差异以及深度网络剪枝算法的特点和不足，本文提出了通信高效的联邦学习模型剪枝（CEMP-FL, communication-efficient model pruning for federated learning）架构，设计了单次层平衡网络剪枝（SBNP, single-shot layer balance network pruning）算法，利用小批量训练样本，以单次方式对全局深度模型进行剪枝。首先，服务器在首个通信轮次中运行SBNP算法进行粗剪枝，剪枝稀疏度尽量接近目标稀疏度，尽量减少模型交互过程中的通信开销，提高联邦学习的通信效率；其次，在随后的通信轮次中，服务器端运行SBNP算法进行精细剪枝，逐步增加模型稀疏度，以便利用客户端本地训练数据集的分布信息，减少终端训练样本分布差异造成的剪枝偏差，提高深度模型性能，加快模型收敛，实现通信和模型性能的联合优化。值得注意的是，在每次粗剪枝和精细剪枝后，服务器端或客户端在获得非结构化剪枝的高稀疏度模型参数时，利用参数矩阵压缩方法进行实际的参数矩阵维度压缩，减少待传输的模型参数大小，确保服务器端和客户端交互模型时能显著降低通信开销。最后，SBNP算法中执行层平衡策略（LBP, layer balance policy）确保了深度模型层之间参数量的平衡，模型稀疏度很大的情况下，有效地推迟了深度模型层坍塌的发生，在实际场景中有利于进一步压缩模型，优化联邦学习的通信效率。本文的创新点总结如下。

1) 提出了通信高效的联邦学习模型剪枝架构，在联邦学习多轮次通信中，通过粗剪枝和精细剪枝的组合，结合非结构化稀疏参数压缩，显著减少了通信过程中传输的深度模型参数量，同时有效地减少了终端侧训练样本分布差异造成的剪枝偏差，实现了通信和模型性能的联合优化。

2) 提出了单次层平衡网络剪枝算法，设计了网络剪枝的层平衡策略，利用小批量训练样本，在考虑层间参数相对平衡的情况下，以单次方式对全局深度

模型进行深度剪枝，在稀疏度很大的情况下，有效地推迟了深度模型层坍塌的发生，有利于进一步压缩模型，提高通信效率，并在理论上给出了证明。

1　相关工作

1.1　联邦学习

联邦学习是一种机器学习范式，可以在一个中心服务器的协调下让多个客户端相互合作，实现数据分散在客户端的情况下也可以得到一个完整的机器学习模型。文献[1]首次提出了联邦学习的概念，并且提出了模型聚合的方法Fedavg。然后，OTfusion方法[8]被提出用于模型聚合优化，通过基于层的模型融合方法实现联邦学习聚合更新。文献[9]将联邦学习聚合分解为局部和全局两个步骤，并提出基于相似度引导的模型聚合方法。同时，Ditto算法[10]用于解决个性化联邦学习场景，实现了联邦学习中的公平性和鲁棒性。根据本地数据分布是底层分布混合这一假设，Marfoq等[11]提出了新的联邦多任务学习算法EM-like，更好地为个性化模型服务。近年来，联邦学习因其独特的分布式架构，很多工作对其在各方面进行了优化，针对后门攻击，在联邦学习架构中提出了一种新的威胁评估框架DBA[12]。文献[13]通过梯度对训练数据进行逆向，提出了新的攻击算法和防御手段。利用生成模型从公共数据集提取先验信息，GGL算法[14]用于改进由隐私防御产生的梯度退化。文献[15]针对目前联邦学习客户端选择策略存在有偏问题，提出了改进算法以实现更快更平滑地聚合收敛。为了实现了更高的学习效率、收敛速度和性能，DivFL算法[16]被提出以改进聚合和分发策略。总之，联邦学习的各种性能优化受到学术界和工业界的广泛关注，但在无线通信环境下，联邦学习的通信高效问题很少被关注，需要进一步研究。

1.2　深度模型剪枝

深度模型剪枝是一种通过去除神经网络冗余参数，实现模型压缩的技术，一般来说，分为训练前剪枝、训练中剪枝和训练后剪枝3类。训练前剪枝是在初始化时对给定的网络进行一次修剪，以减少待训练的模型大小。代表性工作包括根据连接重要性，提出的训练前剪枝算法SNIP[6]和Synflow算法[17]，以实现不依赖训练数据达到目标稀疏度。在初始化时，Edge-Popup算法[18]被提出以实现找到具有很高精度的子模型。文献[19]基于神经切线核分解提出了新的算法，以实现更高的效率和模型准确度。文献[20]提出了一个内存友好的可拓展框架，以加载有效的初始化模型。训练中剪枝是在网络训练过程中通过迭代训练调整神经网络连接。文献[7]提出了一种渐进剪枝算法，将剪枝率从初始稀疏度开始，逐渐增加到目标最终稀疏度。文献[21]在迭代训练中引入额外变量和先验经验进行参数选择。文献[22]在较小规模的设置中利用不稳定分析，从初始化训练开始平衡性能和剪枝深度。训练后剪枝通常在预训练网络的基础上对不重要的权值进行裁剪，早期的剪枝策略一般都是训练后剪枝。近年来，文献[23]通过结合组合搜索方法和坐标下降方法的有效性，提出了新的基于块分解的剪枝算法。针对现有方法使用预定义的修剪策略，LFPC算法[24]被提出以实现不同功能层自适应选择合适的剪枝标准。显然，在联邦学习场景中训练前剪枝更有利于减少通信开销，提高通信效率。

1.3　联邦学习剪枝

联邦学习分布式架构支持将工作负载从服务器分配到资源有限的边缘设备，但目前的深度网络模型对于边缘设备来说，推理和训练所需的计算和存储资源开销很大，特别在无线通信场景中，在带宽受限的无线网络上进行深度模型的多轮交互更新所需的通信开销过大，严重影响了无线场景下联邦学习的应用性能。而深度模型剪枝算法可以通过减少模型参数，进而减少模型多轮交互的通信量。文献[25]提出了一种稀疏增强隐私的联邦学习架构，通过随机剪枝造成的梯度波动来增强各终端侧的数据隐私。文献[26]分别在服务器端和客户端进行全局迭代幅度剪枝实验，提出了对应的联邦学习全局稀疏化和局部稀疏化算法。ZeroFL方法[27]在上行通信之前应用本地稀疏化，并提出3种局部稀疏化策略，在提高了稀疏训练性能的同时降低了通信成本。文献[28]将权重冻结在初始随机值上，并学习如何稀疏随机网络以获得最佳性能。另外，训练前剪枝算法SNIP[6]也常用于联邦学习架构下的一次性剪枝策略，通过训练前剪枝来减少无线场景下深度模型多轮交互的通信量。现有的联邦学习剪枝策略在无线场景下如何有效地提高通信效率考虑不足，例如，如何在保证模型性能的前提下，起始通信轮次就尽可能地减少模型大小？如何有效地避免在深度稀疏

化时模型坍塌现象的发生？因此，无线场景下通信有效的联邦学习剪枝技术值得深入研究。

2　通信高效的联邦学习模型剪枝架构

通信高效的联邦学习模型剪枝架构如图1所示，通信高效的联邦学习模型剪枝（CEMP-FL）架构由一个中心服务器和N个边缘设备组成的联邦学习系统，服务器只存储小批量训练样本用于一次性网络剪枝，边缘设备客户端以分布式方式存储训练数据集，用于本地的深度模型训练而无须传输。

CEMP-FL训练过程由多个通信轮次组成，共包括以下7个阶段，其中，这7个阶段与图1中带序号的流程对应一致。

1) 服务器运行单次层平衡网络剪枝（SBNP）算法进行粗剪枝，即利用小批量训练样本，在考虑层间参数相对平衡的情况下，以单次方式对全局深度模型进行初步深度剪枝。值得注意的是，粗剪枝只在首轮通信中执行，目的是使得深度模型剪枝稀疏度尽量接近目标稀疏度，最大可能地减少随后通信轮次的参数传输量。

2) 随机选择一部分客户端，将轻量化后的全局模型分发给指定的客户端。

3) 客户端接收到服务器端剪枝后的深度模型参数后，利用本地存储的训练数据集进行模型训练，以更新模型参数。

4) 客户端将更新后的全局模型参数上传服务器。

5) 服务器收集到所有客户端更新的深度模型参数后，利用联邦学习汇聚方法形成全局的深度模型。

6) 判断模型剪枝后的稀疏度是否达到目标稀疏度，如果没有达到，则服务器继续运行SBNP算法进行精细剪枝，即利用小批量训练样本，在避免层坍塌的情况下，以单次方式对深度模型进行微细剪枝，以递进的方式逐步逼近模型的目标稀疏度。

7) 判断是否达到目标稀疏度且通信轮次是否超过预定值，如果不满足，返回2)继续下发剪枝后的模型参数，否则直接输出达成目标稀疏度的且训练收敛的深度模型。

第1)步中，服务器只在首轮通信中运行SBNP算法进行粗剪枝，剪枝比例相对较大，剪枝稀疏度尽量接近目标稀疏度，以确保轻量化模型下发到各终端，尽量减少模型交互过程中的通信开销，提高联邦学习的通信效率。与此相反，每轮通信中，只要没有达到目标稀疏度，在第6)步中运行SBNP算法进行精细剪枝，每次剪枝比例相对较小，以利用

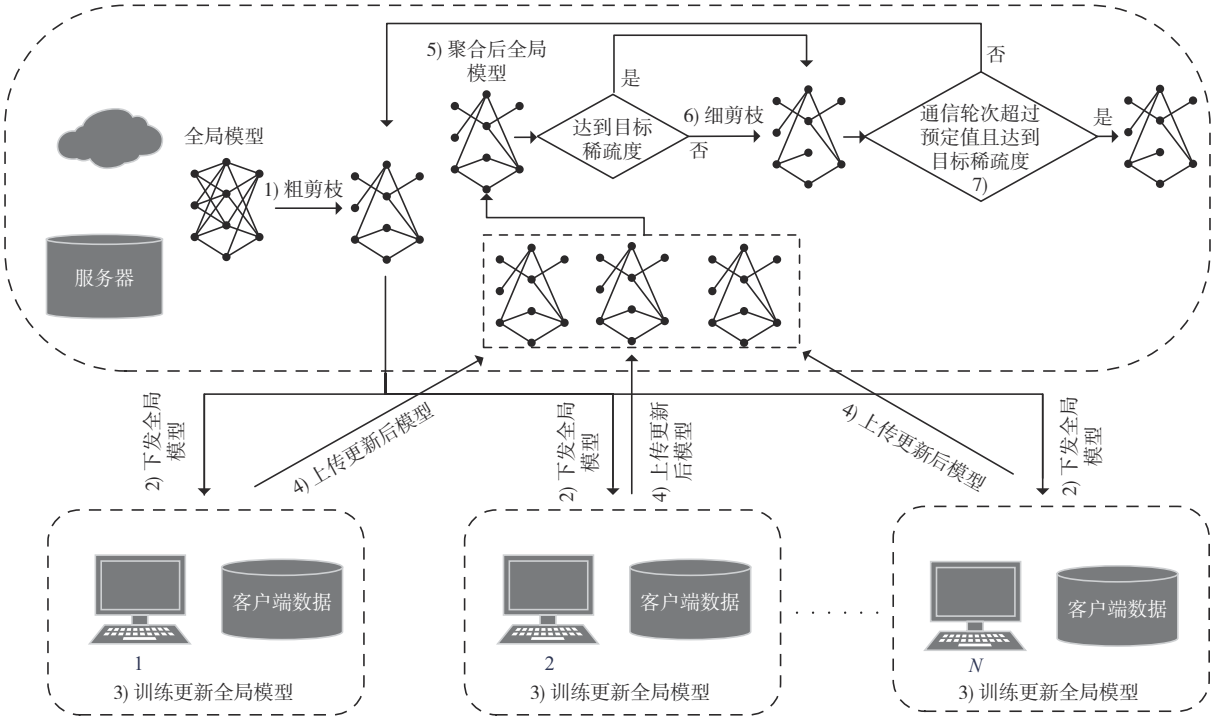


图1　通信高效的联邦学习模型剪枝架构

客户端本地存储的训练数据集分布信息，减少终端训练样本分布差异所带来的剪枝偏差。总之，CEMP-FL 在多个通信轮次中，通过首轮粗剪枝和每轮精细剪枝的组合，可以显著减少通信过程中传输的深度模型参数量，同时有效地减少了终端侧训练样本分布差异所带来的剪枝偏差，实现了通信和模型训练性能的联合优化。并且，CEMP-FL 运行 SBNP 算法，确保了深度模型层之间参数量的平衡，在稀疏度很大的情况下，有效地避免了深度模型坍塌，有利于在实际场景中实现通信有效的联邦学习应用。

3　单次层平衡网络剪枝（SBNP）算法

本文提出的 SBNP 算法包括两个模块：单次网络剪枝算法和网络剪枝的层平衡策略（LBP），其中，单次网络剪枝算法输出的连接敏感度输入 LBP 中，最终输出连接指示变量对应的剪枝策略。

3.1　单次网络剪枝方法

单次网络剪枝使用数据相关的方式来衡量深度网络连接的重要性，即标准打分值仅与训练样本相关，对网络参数不敏感，这样，单次网络剪枝只需要利用训练数据库中的小批量样本，在正式训练之前进行一次网络剪枝，并在稀疏化的深度网络中进行正常的模型训练，避免了剪枝和重新训练递归循环带来的巨大开销，所以单次网络剪枝策略特别适合应用在面向无线通信和联邦学习的场景中。

单次网络剪枝首先从数据集挑选出小批量的数据 $D_{\mathrm{S}}=\left\{\left(x_i,y_i\right)\right\}_{i=1}^{n}$，以数据相关方式确定各个连接的重要性，如式(1)所示

$$\begin{aligned}\min_{\boldsymbol{m},\boldsymbol{\theta}} L\left(\boldsymbol{m}\odot\boldsymbol{\theta};D_{\mathrm{S}}\right)=\min_{\boldsymbol{m},\boldsymbol{\theta}}\frac{1}{n}\sum_{i=1}^{n}\ell\left(\boldsymbol{m}\odot\boldsymbol{\theta};\left(x_i,y_i\right)\right)\\ \text{s.t.}\quad \boldsymbol{\theta}\in\mathbf{R}^k\\ \boldsymbol{m}\in\{0,1\}^k,\left\|\boldsymbol{m}\right\|_0\leqslant K\end{aligned}\tag{1}$$

其中，$L(\cdot)$是标准的损失函数，$\boldsymbol{m}\in\{0,1\}^k$是连接指示变量，取值“0”或“1”分别表示对应参数剪去或保留，k是深度神经网络总的参数量，$\boldsymbol{\theta}$是深度网络的参数集合，K代表目标非零参数量，$\ell(\cdot)$是交叉熵损失函数，$\|\cdot\|_0$表示 0 范数表达式，$\odot$是哈达码积。从式(1)可以看出，$m_j=1$表明对应的第j参数保留在深度网络中，而$m_j=0$表明对应的参数应该被剪枝删去。因此，删除第j参数对损失函数的影响可以表达如下

$$\Delta L_j\left(\boldsymbol{\theta};D_{\mathrm{S}}\right)=L\left(\mathbf{1}\odot\boldsymbol{\theta};D_{\mathrm{S}}\right)-L\left(\left(\mathbf{1}-\boldsymbol{e}_j\right)\odot\boldsymbol{\theta};D_{\mathrm{S}}\right)\tag{2}$$

其中，ΔL_j用来衡量连接j对损失函数的重要程度，$\boldsymbol{e}_j\in\{0,1\}^k$是独热码，对应的第$j$元素是1，$\mathbf{1}$是维度为$k$的全1向量。为了利用深度学习反向传播机制有效地计算出 ΔL_j，需要通过损失函数对连接变量的梯度近似表达式(2)。但由于连接指示变量$\boldsymbol{m}$是2进制的，通过放宽$\boldsymbol{m}$的二元约束，可以由L对m_j的导数$g_j\left(\boldsymbol{\theta};D_{\mathrm{S}}\right)$近似得到$\Delta L_j$，表示如下

$$\begin{aligned}\Delta L_j\left(\boldsymbol{\theta};D_{\mathrm{S}}\right)\approx g_j\left(\boldsymbol{\theta};D_{\mathrm{S}}\right)=\frac{\partial L\left(\boldsymbol{m}\odot\boldsymbol{\theta};D_{\mathrm{S}}\right)}{\partial m_j}\Big|_{\boldsymbol{m}=\mathbf{1}}=\\ \lim_{\delta\to 0}\frac{L\left(\boldsymbol{m}\odot\boldsymbol{\theta};D_{\mathrm{S}}\right)-L\left(\left(\boldsymbol{m}-\delta\boldsymbol{e}_j\right)\odot\boldsymbol{\theta};D_{\mathrm{S}}\right)}{\delta}\Big|_{\boldsymbol{m}=\mathbf{1}}\end{aligned}\tag{3}$$

其中，δ是乘性因子，用来扰动权重并测量损失的变化。由式(3)可知，$\left\{g_j\left(\boldsymbol{\theta};D_{\mathrm{S}}\right)\right\}_{j=1}^{k}$可以在深度神经网络一次前向和反向传播中，利用梯度推导计算得到。这样我们可以利用$g_j\left(\boldsymbol{\theta};D_{\mathrm{S}}\right)$来衡量参数$j$对损失函数的影响，如果$g_j\left(\boldsymbol{\theta};D_{\mathrm{S}}\right)$值较大，说明参数$j$对损失函数的影响较大，剪枝过程中最好保留；反之，就对参数j进行剪枝。将式(3)归一化得到式(4)

$$v_j=\frac{\left|g_j\left(\boldsymbol{\theta};D_{\mathrm{S}}\right)\right|}{\sum_{i=1}^{k}\left|g_i\left(\boldsymbol{\theta};D_{\mathrm{S}}\right)\right|}\tag{4}$$

其中，v_j是归一化的梯度值，用来衡量参数j的重要性，通常定义为参数j的连接敏感度。这样网络剪枝问题就可以转换成：在一次小批量样本训练情况下，通过式(3)和式(4)获取连接敏感度$\left\{v_j\right\}_{j=1}^{k}$后，对$\left\{v_j\right\}_{j=1}^{k}$进行降序排列，并根据目标非零参数量，采用保留 top$-K$个对应参数的剪枝策略，即如果v_j不在 top$-K$中，则$m_j=0$表明第j参数必须被剪去，且非零参数量满足$\|\boldsymbol{m}\|_0=K$。

3.2　网络剪枝的层平衡策略（LBP）

由式(3)和式(4)可以看出，单次网络剪枝策略实际上是基于梯度的剪枝方法，连接敏感度v_j与连接j所在层的可训练参数量大小成反比，因而，修剪时倾向于修剪参数量大的层，对于剪枝和重新训练的循环策略，随着某层可训练参数被剪去，对应的平均连接敏感度会提高，在下一次循环剪枝时，该层参数被剪枝的概率会减少。但由于一次性剪枝仅在训练前剪枝一次，连接敏感度v_j不会随着剪枝

过程而动态调整，因此会造成参数量大的层被过分修剪甚至出现层坍塌的现象。剪枝过程中的层坍塌现象是指在按照某种剪枝算法，剪除了深度模型单层的所有权重参数，而深度网络其他位置还存在可剪枝的参数。层坍塌因训练的不可持续性造成模型性能的急剧下降，因此如何避免剪枝过程中过早地出现层坍塌成为衡量剪枝算法的重要指标之一。

为了在单次网络剪枝中改善深度模型层坍塌问题，我们设计了单次剪枝的层平衡策略，设整个深度神经网络共有l层，首先定义了层稀疏比例r_i为

$$r_i = 1 - \frac{\left\| \boldsymbol{m}_i \right\|_0}{\left\| \boldsymbol{\theta}_i \right\|_0}, i \in \{1, \cdots, l\} \tag{5}$$

其中，$\boldsymbol{m}_i$和$\boldsymbol{\theta}_i$分别表示深度神经网络第i层的连接指示变量和未剪枝前的参数集合，且满足$\left\| \boldsymbol{m}_i \right\|_0 \leqslant \left\| \boldsymbol{\theta}_i \right\|_0$，很显然，$r_i \in [0, 1]$表示第$i$层中剪去的参数量所占的比例。为了避免层坍塌，我们希望第i层参数剪枝的概率和层稀疏比例r_i成反向关系，为此，我们利用反余弦函数来定义第i层参数剪枝概率p_i为

$$p_i = \frac{2}{\pi} \arccos r_i, i \in \{1, \cdots, l\} \tag{6}$$

其中，剪枝算法起始时，第i层参数很少被剪枝，r_i接近0，第i层参数剪枝概率p_i接近1；随着被剪枝的参数占比越来越大，r_i接近1，第i层参数剪枝概率p_i接近0。这样，可以有效地保证当第i层的参数所剩不多时，剪枝概率越来越小，从而有效地避免了层坍塌的出现。

为了进一步解释提出的LBP算法的功能，下面证明LBP可以有效地推迟深度模型层坍塌发生。

首先，定义压缩率$\rho = k/u$，其中，k是剪枝前深度神经网络总的参数量，u是剪枝后深度神经网络的参数量，ρ越大说明剪枝后的网络稀疏度越大。

其次，定义临界压缩率$\rho_{\text{cr}} = k/(k - k_1)$，其中，$k_1$是单次网络剪枝算法在不发生层坍塌时最大的参数剪除数量；定义层平衡策略下的压缩率$\bar{\rho}_{\text{cr}} = k/(k - \bar{k}_1)$，其中，$\bar{k}_1$是考虑层平衡策略下的单次剪枝算法不发生层坍塌时参数剪除数量。

剪枝算法的最大压缩率为$\rho_{\max} = k/l$，其中，l是深度神经网络的层数，$\rho_{\max}$是不发生层坍塌时的最大压缩比，每层只保留一个参数，也是所有剪枝算法不发生层坍塌时压缩比的理论上限值，即$\rho_{\text{cr}} \leqslant \rho_{\max}$，$\bar{\rho}_{\text{cr}} \leqslant \rho_{\max}$。为了评价LBP的有效性，需要证明$\rho_{\text{cr}} \leqslant \bar{\rho}_{\text{cr}} \leqslant \rho_{\max}$，即不发生层坍塌时，LBP可以剪除更多的参数量，达到更大的模型稀疏度，证明过程如下。

1) 对于全局模型参数$\boldsymbol{\theta} \in \mathbf{R}^k$，根据式(3)和式(4)，计算连接灵敏度$\{v_j\}_{j=1}^{k}$，其中，$v_j$为参数$j$的连接敏感度。

2) 对$\{v_j\}_{j=1}^{k}$进行升序排列，得到$T = \{\tau_1^{l_1}, \cdots, \tau_i^{l_i}, \cdots, \tau_k^{l_k}\}$，其中,上标$l_i$代表第$i$个元素对应参数所在层的序号，即对模型性能影响不大的参数排列在前，将优先被剪除。

3) 设临界压缩率$\rho_{\text{cr}} = k/(k - k_1)$，即$\tau_1^{l_1}$，$\tau_2^{l_2}$，$\cdots$，$\tau_{k_1}^{l_{k_1}}$对应的共$k_1$个参数被剪除，模型不会发生层坍塌，而第$k_1 + 1$个元素$\tau_{k_1+1}^{l_{k_1+1}}$对应的参数是敏感参数，即$l_{k_1+1}$层的最后一个参数，当$\tau_{k_1+1}^{l_{k_1+1}}$对应的参数被剪除后，层坍塌将会发生，假设敏感参数不连续且随机分散在$T = \{\tau_1^{l_1}, \cdots, \tau_i^{l_i}, \cdots, \tau_k^{l_k}\}$中。

4) 考虑LBP算法的情况下，升序排列的$T = \{\tau_1^{l_1}, \cdots, \tau_i^{l_i}, \cdots, \tau_k^{l_k}\}$中第$i$个元素$\tau_i^{l_i}$对应的参数以层参数剪枝概率$p_{l_i}$被剪除，$p_{l_i}$由式(6)计算，即第$i$层参数剪枝概率和层稀疏比例$r_i$成反余弦函数关系，为了计算方便，设：如果$r_i \to 1$，则$p_{l_i} \to \varepsilon$；否则$p_{l_i} \to 1$，其中，$\varepsilon$是足够小的常量。

5) 考虑LBP算法的情况下，参数剪枝数量$\bar{k}_1$通过计算数学期望得到

$$\begin{aligned} \mathrm{E}(\bar{k}_1) = {} & \varepsilon k_1 + (1-\varepsilon)(k_1 + 1) + \\ & (1-\varepsilon)(k_1 + 2) + \cdots > \varepsilon k_1 + (1-\varepsilon)k_1 = k_1 \end{aligned} \tag{7}$$

其中，如果第$k_1 + 1$个元素$\tau_{k_1+1}^{l_{k_1+1}}$对应的参数以概率ε被剪除，层坍塌发生，则$\bar{k}_1 = k_1$，注意，根据临界压缩率的设定，$\tau_{k_1+1}^{l_{k_1+1}}$对应的参数是l_{k_1+1}层的最后一个参数；如果$\tau_{k_1+1}^{l_{k_1+1}}$对应的参数以概率$1-\varepsilon$未被剪除，则继续以接近1的概率剪除后续元素$\tau_{k_1+2}^{l_{k_1+2}}$，$\tau_{k_1+3}^{l_{k_1+3}}$等对应的参数，则$\bar{k}_1 = k_1 + 1$，$k_1 + 2$等。

因此

$$\bar{\rho}_{\text{cr}} = \frac{k}{k - \mathrm{E}(\bar{k}_1)} > \frac{k}{k - k_1} = \rho_{\text{cr}} \tag{8}$$

由上可知，式(8)证明了层平衡策略不仅以升序方式优先剪除不重要参数，而且通过引入层参数剪枝

概率有效地推迟了对敏感参数的剪除，从而实现了剪除更多的参数量，得到更大的模型稀疏度。从而在移动边缘计算场景中，服务器和终端可以交互更加稀疏的模型，通过稀疏模型的压缩，可以有效地提升通信效率。LBP 算法如算法 1 所示。

算法 1　LBP 算法

输入：连接敏感度 $\{v_j\}_{j=1}^{k}$，目标非零参数量 K，最大循环次数 G。

初始化：层稀疏比例 $r_i = 0$，$i \in \{1, \cdots, l\}$

对 $\{v_j\}_{j=1}^{k}$ 进行升序排列，得到 $T = \{\tau_1^{l_1}, \cdots, \tau_i^{l_i}, \cdots, \tau_k^{l_k}\}$，其中，上标 l_i 代表第 i 个参数所在层序号，令 $k' = k$；　//排序使得重要的参数排列在前面，尽量剪除靠后的参数

根据式(5)和式(6)，计算 r_{l_i} 和 p_{l_i}；

for $j = 1, \cdots, G$ **do**

for $\tau_i^{l_i} = \tau_1^{l_1}, \cdots, \tau_k^{l_k}$ do

以概率 p_{l_i} 设 $\boldsymbol{m}_i = 0$，即以概率 p_{l_i} 对参数剪枝；//参数较少层被剪的概率也减少

if $\boldsymbol{m}_i = 0$

利用式(5)和式(6)，更新 r_{l_i} 和 p_{l_i}；

$k' = k' - 1$；

end if

end for

if $K \geqslant k'$ **break**　//达到预定的剪枝比例就结束剪枝操作；

end for

输出：连接指示变量 $\boldsymbol{m} \in \{0,1\}^k$。

3.3　非结构化剪枝参数矩阵压缩

本文采用的 SBNP 属于非结构化剪枝，相对结构化剪枝，非结构化剪枝在相同性能的情况下，可以达到更大的模型稀疏度，但非结构化剪枝所剪参数分布不规则，不能有效地减少神经网络每层参数矩阵的实际大小，从而不能真正减少所消耗的硬件资源和运算速度。为了解决这一问题，我们采用非结构化剪枝参数矩阵压缩策略[29]，通过压缩稀疏参数矩阵的实际大小，实现对存储和计算资源地有效减少，从而在联邦学习场景下实现真正的通信有效。

非结构化剪枝参数矩阵压缩首先对稀疏参数矩阵进行行或列置换，在置换过程中使用模拟退火算法来获得最佳的置换步骤，置换后的稀疏参数矩阵可以压缩成小而密集的格式，从而实现非结构化剪枝后参数矩阵尺寸的实际减少，以实现参数矩阵最大程度被压缩的目的。稀疏参数矩阵压缩示意图如图 2 所示，根据硬件架构将权重矩阵划分成若干子矩阵部分并置换行和列，当子矩阵中的列向量之间非零元素不重叠时进行列合并，8 × 4 的参数矩阵最终压缩成 8 × 2 的矩阵，压缩后的参数矩阵由稀疏变紧致，从而实现了非结构化剪枝后模型尺寸的实际减少。值得注意的是，压缩过程中置换和合并的位置都有记录，这样压缩后的参数矩阵还可以恢复到原先的稀疏状态。

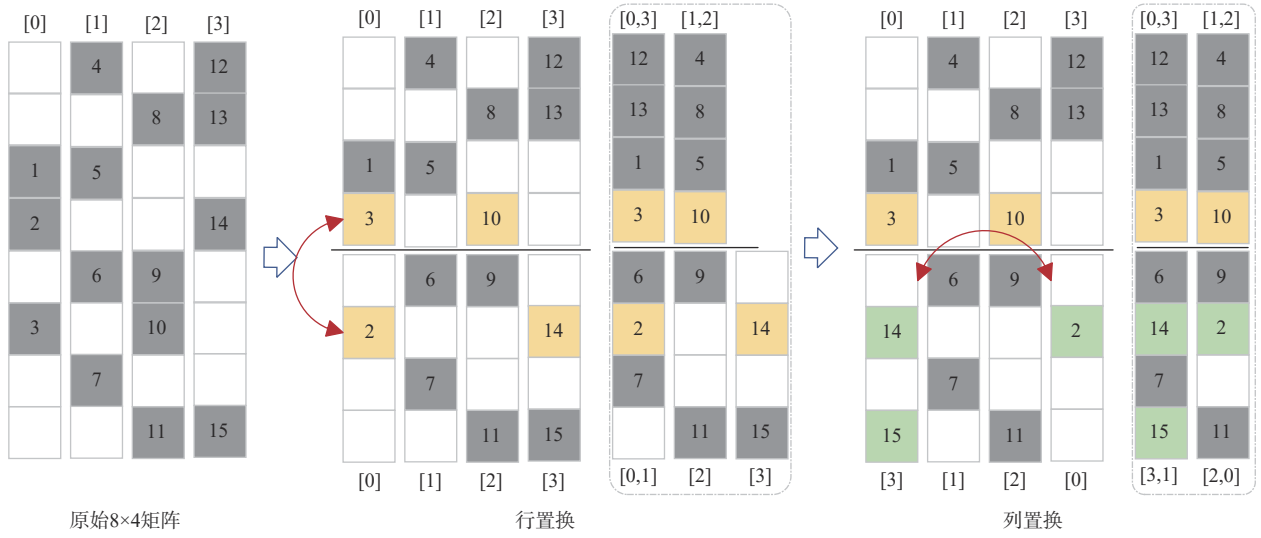


图 2　稀疏参数矩阵压缩示意图

为了表示方便，压缩函数$\Phi(\cdot)$和解压缩函数$\overline{\Phi}(\cdot)$分别表示压缩稀疏参数和解压缩稀疏参数的过程，当服务器端或客户端需要发送剪枝后深度模型时，首先调用压缩函数

$$\overline{\boldsymbol{\theta}} = \Phi(\boldsymbol{\theta}) \tag{9}$$

其中，$\boldsymbol{\theta} \in \mathbf{R}^k$，并且$\|\boldsymbol{m}\|_0 = K$表示总参数量为$k$的稀疏参数非零元素数量为$K$，$\overline{\boldsymbol{\theta}} \in \mathbf{R}^{\overline{\mathcal{K}}}$是压缩后待传输的模型，实际传输的模型参数量为$\overline{\mathcal{K}}$，并且满足$\overline{\mathcal{K}} = (1+\delta)K \ll k$，考虑到不可能完全压缩而存在扩展因子$\delta$，根据文献[29]的实验结果，满足$\delta \in [0.35, 0.50]$，值得注意的是，即使存在扩展因子$\delta$，因为稀疏参数非零元素数量$\mathcal{K}$通常远小于原模型参数量$k$，因而压缩后模型在尺寸上仍远小于原模型大小，可以显著地减少联邦学习架构下模型交互过程中的通信开销。在后面的实验过程中，取最大的扩展因子$\delta = 0.50$以衡量本文所提算法的通信效率。由上可知，正是剪枝算法和压缩稀疏参数的联合使用，使得非结构化剪枝算法在联邦学习架构下能显著地减少通信开销，提高通信效率。

同样，接收端需要调用解压缩函数

$$\boldsymbol{\theta} = \overline{\Phi}\left(\overline{\boldsymbol{\theta}}\right) \tag{10}$$

其中，$\overline{\boldsymbol{\theta}} \in \mathbf{R}^{\overline{\mathcal{K}}}$，$\boldsymbol{\theta} \in \mathbf{R}^k$，且$\|\boldsymbol{m}\|_0 = K$是恢复的稀疏参数，和服务器端模型保持一致，用于联邦学习的训练或者剪枝操作。

4　通信高效的联邦学习模型剪枝（CEMP-FL）算法描述

通信高效的联邦学习模型剪枝（CEMP-FL）利用单次层平衡网络剪枝（SBNP）算法，配合联邦学习多个通信轮次，结合粗剪枝和精细剪枝，实现了通信和模型训练性能的联合优化，并在深度稀疏时有效地避免了层坍塌现象，整个算法的描述如下。

步骤 1　调用SBNP算法，即服务器端利用小批量训练样本D_S，在已知总目标非零参数量K的情况下，确定首轮非零参数量$\hat{\mathcal{K}}$（接近并大于K），根据式(3)和式(4)，计算连接敏感度$\{v_j\}_{j=1}^k$，其中，k是全局模型参数量，调用网络剪枝的层平衡策略算法LBP(·)，获得连接指示变量$\boldsymbol{m} \in \{0,1\}^k$，满足$\|\boldsymbol{m}\|_0 = \hat{\mathcal{K}}$，并进行相应的模型粗剪枝$\boldsymbol{\theta}_t = \boldsymbol{m} \odot \boldsymbol{\theta}_t$，形成轻量化全局模型。

步骤 2　服务器端调用压缩函数$\overline{\boldsymbol{\theta}}_t = \Phi\left(\boldsymbol{\theta}_t\right)$形成压缩的模型参数用于实际传输，随机挑选$N$个客户端，使每个客户端从服务端下载全局模型，客户端接收模型后调用解压缩函数$\boldsymbol{\theta}_{i,t} = \overline{\Phi}\left(\overline{\boldsymbol{\theta}}_t\right)$形成稀疏的模型参数用于本地模型的更新。

步骤 3　客户端利用本地训练数据集，采用监督训练方式优化损失函数，实现对本地模型参数的更新，表示为

$$\boldsymbol{\theta}_{i,t+1} \leftarrow \boldsymbol{\theta}_{i,t} - \eta \nabla L\left(\boldsymbol{\theta}_{i,t}; D_i\right) \tag{11}$$

其中，η为学习率，$L\left(\boldsymbol{\theta}_{i,t}; D_i\right)$为序号为$i$的客户端参数为$\boldsymbol{\theta}_{i,t}$且本地训练集为$D_i$时的损失函数，$\boldsymbol{\theta}_{i,t+1}$代表更新后的模型参数。

步骤 4　每个客户端对更新后的本地模型参数调用压缩函数$\overline{\boldsymbol{\theta}}_{i,t+1} = \Phi\left(\boldsymbol{\theta}_{i,t+1}\right)$进行压缩，并上传到服务器。

步骤5　服务器调用解压缩函数$\boldsymbol{\theta}_{i,t+1} = \overline{\Phi}\left(\overline{\boldsymbol{\theta}}_{i,t+1}\right)$，采用联邦学习汇聚的方法对各个客户端上传的本地模型进行聚合，形成更新后的全局模型参数$\boldsymbol{\theta}_{t+1}$，汇聚如式(12)所示

$$\boldsymbol{\theta}_{t+1} = \sum_{i=1}^{N} \frac{n_i}{\bar{n}} \boldsymbol{\theta}_{i,t+1} \tag{12}$$

其中，$\boldsymbol{\theta}_{i,t+1}$为序号为$i$的客户端更新后的本地模型参数，$N$为参与训练客户端的数量，$n_i$为序号为$i$的客户端本地训练数据集的数量，$\bar{n}$为参与更新的客户端全部训练数据集的数量

步骤 6　当非零参数量满足$\hat{\mathcal{K}} > K$时，调用SBNP算法，即$k = \hat{\mathcal{K}}$为更新后的全局模型参数量，非零参数量更新为$\hat{\mathcal{K}} = \hat{\mathcal{K}} - \Delta$，其中，$\Delta$为递减参数量，根据式(3)和式(4)，计算连接敏感度$\{v_j\}_{j=1}^k$，并调用网络剪枝的层平衡策略算法LBP(·)，获得连接指示变量$\boldsymbol{m} \in \{0,1\}^k$，满足$\|\boldsymbol{m}\|_0 = \hat{\mathcal{K}}$，并进行相应的模型精细剪枝$\boldsymbol{\theta}_{t+1} = \boldsymbol{m} \odot \boldsymbol{\theta}_{t+1}$，形成轻量化的全局模型，减少终端训练样本分布差异所带来的剪枝偏差。当非零参数量满足$\hat{\mathcal{K}} \leqslant K$且通信轮次大于预定值时，算法结束，输出符合稀疏度的收敛全局深度模型；否则，返回步骤2。

CEMP-FL算法（服务器端）如算法2所示，CEMP-FL算法（客户端）如算法3所示。

算法 2　CEMP-FL算法（服务器端）

输入：小批量训练数据集和客户端训练数据集

D_S和D_i，每轮随机抽取的客户端数量N，通信轮次变量和上限$t=1$和T，总目标非零参数量和当前非零参数量K和K_{cur}，递减参数量Δ，全局模型更新后参数大小k。

服务器端导入模型，随机初始化模型参数$\boldsymbol{\theta}_t$；

随机确定参与模型更新的N个客户端集合$\boldsymbol{c}=\{c_i\}_{i=1}^{N}$；

确定首轮非零参数量$\hat{\mathcal{K}}$，$K_{cur}=\hat{\mathcal{K}}$，并根据式(3)和式(4)，计算连接灵敏度$\{v_j\}_{j=1}^{k}$；//连接灵敏度表明参数的重要性，也是剪枝的依据

调用层平衡策略算法LBP(·)，计算连接指示变量$\boldsymbol{m}\in\{0,1\}^k$，且$\|\boldsymbol{m}\|_0=\hat{\mathcal{K}}$；

对模型进行粗剪枝$\boldsymbol{\theta}_t=\boldsymbol{m}\odot\boldsymbol{\theta}_t$，调用压缩函数$\overline{\boldsymbol{\theta}}_t=\Phi(\boldsymbol{\theta}_t)$；//粗剪枝比例相对比较大，然后对稀疏参数调用压缩函数进行维度压缩

while $t<T\|K_{cur}>K$ **do**

for $\forall c_i\in\boldsymbol{c}$ **do**

调用客户端ClientUpdate($\overline{\boldsymbol{\theta}}_t$)，同时调用解压缩函数$\boldsymbol{\theta}_{i,t+1}=\overline{\Phi}(\overline{\boldsymbol{\theta}}_{i,t+1})$，并更新$N$个客户端模型$\{\boldsymbol{\theta}_{i,t+1}\}_{i=1}^{N}$；//调用解压缩函数的目的是使服务器端和客户端深度模型保持一致

end for

根据式(12)，聚合各客户端的上传模型，更新全局模型参数$\boldsymbol{\theta}_{t+1}$；

if $K_{cur}>K$ **do**

$k=K_{cur}$；

$K_{cur}=K_{cur}-\Delta$；//这里是细剪枝，剪枝的幅度相对较小

根据式(3)和式(4)计算连接敏感度$\{v_j\}_{j=1}^{k}$；

调用层平衡策略算法LBP(·)，计算连接指示变量$\boldsymbol{m}\in\{0,1\}^k$，且$\|\boldsymbol{m}\|_0=K_{cur}$；

对模型进行精细剪枝$\boldsymbol{\theta}_{t+1}=\boldsymbol{m}\odot\boldsymbol{\theta}_{t+1}$，调用压缩函数$\overline{\boldsymbol{\theta}}_t=\Phi(\boldsymbol{\theta}_t)$；//精细剪枝比例相对较小，对稀疏参数调用压缩函数进行维度压缩

end if

$t=t+1$；

end while

输出：轻量化的全局模型$\boldsymbol{\theta}_{t+1}$。

算法3　CEMP-FL算法（客户端）

输入：服务器端压缩全局模型参数$\overline{\boldsymbol{\theta}}_t$，客户端本地训练集$D_i$。

调用客户端函数ClientUpdate($\overline{\boldsymbol{\theta}}_t$)；

从服务器端下载全局模型$\overline{\boldsymbol{\theta}}_t$，并调用解压缩函数$\boldsymbol{\theta}_{i,t}=\overline{\Phi}(\overline{\boldsymbol{\theta}}_t)$；//调用解压缩函数的目的是使服务器端和客户端深度模型保持一致

D_i分成多个训练批次B；

for $B\in D_i$ **do**

根据式(11)，更新本地模型$\boldsymbol{\theta}_{i,t+1}$；

end for

调用压缩函数$\overline{\boldsymbol{\theta}}_{i,t+1}=\Phi(\boldsymbol{\theta}_{i,t+1})$；//客户端发送前对模型进行压缩

输出：更新后的本地压缩模型$\overline{\boldsymbol{\theta}}_{i,t+1}$。

5　实验及性能分析

5.1　实验设置和对比算法

1) 实验环境

本文实验的软件环境：Python版本为3.7.1，Pytorch版本为1.0.0，Pandas版本为1.3.5，Protobuf版本为3.6.1；本文实验的硬件环境：服务器版本为Ubuntu16.04.1，GPU型号为Tesla K80，GPU有20块，每块GPU显存为11 441 MB，处理器为英特尔中央处理器Intel(R) Xeon(R) Silver 4210 CPU @ 2.20 GHz。

2) 数据集和深度模型

为了方便算法性能对比，本文在基准数据集MNIST上选用LeNet_5_Caffe和LeNet_300_100模型，在CIFAR-10上选用VGG-D模型。LeNet_5_Caffe模型大小为1.7 MB，架构包括两个卷积层、最大池化层和两个全连接层；LeNet_300_100模型大小为1.0 MB，架构包括3个全连接层。VGG-D模型的大小为58.3 MB，包含19个卷积层和3个全连接层。数据集和深度模型具体参数见表1。

表1　数据集和深度模型具体参数

深度模型	数据集	数据集大小/张	模型大小/MB
LeNet_5_Caffe	MNIST	70 000	1.7
LeNet_300_100	MNIST	70 000	1.0
VGG-D	CIFAR-10	60 000	58.3

本文实验使用Pytorch框架下的随机梯度下降（SGD, stochastic gradient descent）优化器进行反向传播时的参数优化，实验中所用的主要参数为：

学习率设置为0.1，质量衰减设置为5.0×10^{-4}，客户端的batchsize设置为10，MNIST数据集上通信轮次设置为50，CIFAR-10数据集上通信轮次设置为200。

3) 评价指标

联邦学习架构需要服务器和客户端进行多轮通信迭代，对深度模型进行分布式训练，因而从两个角度评估联邦学习轻量化性能，即联邦学习的通信成本和模型的分类精度。

联邦学习的通信成本定义为服务器到客户端多轮通信迭代过程中，交互深度模型所需传输的数据总量，如式(13)所示

$$C = T \times \left|\boldsymbol{\theta}_i\right| + N \times T \times \left|\boldsymbol{\theta}_i\right| \tag{13}$$

其中，C是联邦学习的通信成本，N是参与训练客户端数量，$\left|\boldsymbol{\theta}_i\right|$是第$i$轮通信过程中传输的深度模型参数量，$T$是通信轮次。

深度模型的分类精度采用top-1精度，即取概率向量里最大的一个作为预测结果，如果预测结果正确，则分类正确，否则分类错误，表示为

$$\mathrm{Acc}_{\text{top-1}} = \frac{S_r}{S} \tag{14}$$

其中，$\mathrm{Acc}_{\text{top-1}}$是top-1准确率，$S_r$是正确分类样本总数，$S$是总样本数。

4) 对比算法

为了衡量联邦学习剪枝算法的性能，在相同的实验环境中，本文将所提算法与5种联邦学习算法进行性能对比，其中，第一种联邦学习算法作为基准算法，没有进行剪枝操作，其他4种为联邦学习剪枝算法。

• Fedavg[1]：不涉及剪枝的经典联邦学习算法。在每一轮通信中，随机选择的客户端从服务器下载全局模型，然后使用本地数据训练模型。客户端将训练后的模型上传到服务器，服务器执行聚合操作更新全局模型，重复多轮，最终得到训练好的全局模型。

• FedSparsify-Global [26]：在联邦学习框架中，引入逐步幅度剪枝算法，即每次迭代过程中，计算模型所有层中权重参数并将参数幅度按从小到大的顺序排列，并删除低于阈值的参数，迭代过程中逐步减少模型的参数数量，为了比较剪枝算法效果，对于原论文多数投票的聚合方式，本文仍采用联邦汇聚进行聚合。

• 一次性剪枝（OMP, one-shot magnitude pruning）[30]：在联邦学习框架中，引入一次性幅度剪枝算法。联邦学习迭代一定轮次后，计算模型所有层中权重参数的幅度，并将它们按从小到大的顺序排列，并一次性删除低于阈值的参数，以减少模型的参数数量。

• Sub-FedAvg[31]：在联邦学习架构中引入结构剪枝和非结构剪枝的混合方式，通过为每个客户端获取小的深度模型子网，在异构数据环境中经过迭代训练以获取满足特定客户端的定制模型。

• Ditto[10]：客户端同时训练服务器传来的全局模型和本地的个性化模型，在客户端局部经验损失和对全局模型的近端项损失函数的基础上，在联邦架构下训练以获取全局模型和满足特定客户端的定制模型。

• 本文所提的通信高效的联邦学习模型剪枝（CEMP-FL）算法：利用单次层平衡网络剪枝（SBNP）算法，在联邦学习多个通信轮次中，结合粗剪枝和精细剪枝，实现了通信和模型训练性能的联合优化。具体包括：粗剪枝在第一个通信轮次就尽可能地减少网络模型参数；随后几个通信轮次，精细剪枝逐步减少模型参数以逼近目标稀疏度，减少因终端训练样本分布差异带来的剪枝偏差，并在深度稀疏时有效地避免层坍塌现象。

实验汇总摘要见表2，其中标明了各实验的目的和衡量指标。

5.2　CEMP-FL算法性能讨论

1) 客户端训练数据集D_i对性能的影响

式(11)中，客户端利用本地训练数据集，采用监督方式优化损失函数，实现对本地模型参数的更新，我们在MNIST数据集和CIFAR-10数据集上讨

表2　　**实验汇总摘要**

实验名称	实验目的	实验衡量指标
客户端训练数据集D_i对性能的影响	验证客户端训练数据集的大小对服务器汇聚模型性能的影响	$\mathrm{Acc}_{\text{top-1}}$
每轮随机抽取的客户端数量N对性能的影响	验证每轮随机抽取客户端数量的大小对服务器汇聚模型性能的影响	$\mathrm{Acc}_{\text{top-1}}$
层平衡策略的消融实验	验证层平衡策略（LBP）算法的抑制模型坍塌性能	top-1准确率
通信有效性对比实验	验证所提CEMP-FL算法的通信有效性	通信成本$\mathrm{Acc}_{\text{top-1}}$

论客户端训练集比例$\bar{D}_i=|D_i|/|D|$（即客户端本地训练集大小占总训练集的比例）对CEMP-FL算法的性能影响，CEMP-FL分类性能随$\bar{D}_i$的变化情况见表3。在MNIST数据集中，客户端训练批次设置为10，参与训练的客户端数量N为10，通信轮次设置为50；在CIFAR-10数据集中，客户端训练批次设置为10，参与训练的客户端数量为10，通信轮次设置为200。

表3　CEMP-FL分类性能随$\bar{D}_i$的变化情况

评价指标	$\bar{D}_i$			
	0.005	0.010	0.020	0.025
$Acc_{top\text{-}1}$（MNIST+LeNet_5_Caffe）	98.16%	98.61%	98.83%	98.89%
$Acc_{top\text{-}1}$(CIFAR10+VGG-D)	87.27%	89.57%	90.09%	90.23%

由表3可知，经过同样的联邦学习通信轮次后，随着$\bar{D}_i$逐渐增大，准确率也逐渐增加，但当$\bar{D}_i$取值大于0.010时，一定的通信轮次后，在两个数据集中$\bar{D}_i$对准确率的影响都不大，因此在接下来的实验中，我们将$\bar{D}_i$设置为0.010。

2) 每轮随机抽取的客户端数量N对性能的影响

式(12)中，联邦学习对N个客户端上传的本地模型进行聚合，形成更新后的全局模型参数$\boldsymbol{\theta}_{t+1}$，接下来本文在MNIST数据集和CIFAR-10数据集上讨论N对性能的影响，CEMP-FL分类性能随N的变化情况见表4。在MNIST数据集中，客户端训练批次设置为10，客户端训练集比例$\bar{D}_i$为0.010，通信轮次设置为50；在CIFAR-10数据集中，客户端训练批次设置为10，客户端训练集比例$\bar{D}_i$为0.010，通信轮次设置为200。

表4　CEMP-FL分类性能随N的变化情况

评价指标	N			
	5	10	20	25
$Acc_{top\text{-}1}$（MNIST+LeNet_5_Caffe）	98.52%	98.61%	98.63%	98.77%
$Acc_{top\text{-}1}$（CIFAR-10+VGG-D)	88.53%	89.57%	89.67%	90.63%

由表4知，在经过同样的联邦学习通信轮次后，不同客户端数量N对准确率没有明显的影响，即客户端数量在一定范围内对联邦学习性能的影响不明显，在接下来的实验中，本文将N设置为10。

3) 层平衡策略的消融实验

本文提出的CEMP-FL算法应用了SBNP算法，确保了深度模型层之间参数量的平衡，在稀疏度很大的情况下，有效地避免了深度模型坍塌，其中SBNP算法的核心是网络剪枝的LBP算法，为了验证CEMP-FL算法在模型深度稀疏化情况下的抑制模型坍塌能力，删除SBNP算法中的LBP模块，命名这种情况下的算法为N-CEMP-FL算法，本文研究了不同模型压缩比下CEMP-FL算法和N-CEMP-FL算法的指标对比，其中，模型压缩比定义为剪枝前后模型参数大小的比例，模型压缩比越大，说明剪枝的程度越大，模型越稀疏，CEMP-FL算法和N-CEMP-FL算法准确率和模型压缩率的关系（LeNet_5_Caffe模型+MNIST数据集）如图3所示。

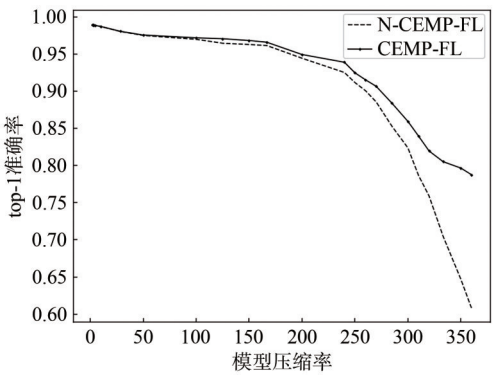


图3　CEMP-FL算法和N-CEMP-FL算法准确率和模型压缩率的关系（LeNet_5_Caffe模型+MNIST数据集）

由图3可以看出，当应用CEMP-FL和N-CEMP-FL两种算法分别在MNIST数据库中对LeNet_5_Caffe模型在联邦学习架构下进行剪枝操作，模型压缩率在0~300之间时，N-CEMP-FL算法和CEMP-FL算法的准确率相差不大，说明此时深度模型没有出现层坍塌情况，随着剪枝操作继续，模型压缩率在300~400之间时，N-CEMP-FL算法出现了比较陡峭的性能下降，说明此时模型出现了层坍塌情况，性能急剧下降；而CEMP-FL算法在相同的压缩率时，维持了相对较好准确率，有效地推迟了性能陡降的发生，表明本文所提的CEMP-FL算法在高压缩率时可以有效地延缓层坍塌的发生。

5.3　算法性能比较和讨论

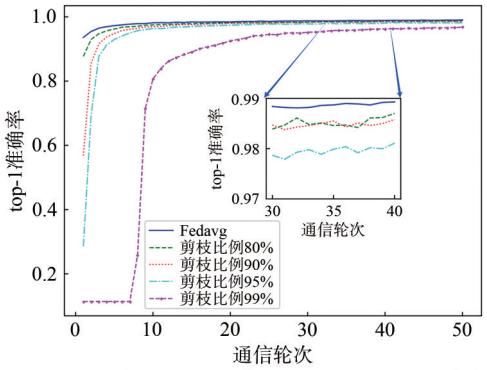
本文将没有任何剪枝操作的Fedavg算法作为基准算法框架，目标稀疏度对应的剪枝比例定义为剪去的参数量占总参数量的比例。在MNIST数据集和不同深度模型情况下，与4个不同剪枝比例的

CEMP-FL 算法（CEMP-FL 80%、CEMP-FL 90%、CEMP-FL 95%和CEMP-FL 99%）进行性能对比，在CIFAR-10 数据集和 VGG-D深度模型情况下，与5个不同剪枝比例的CEMP-FL 算法（CEMP-FL 70%、CEMP-FL 80%、CEMP-FL 90%、CEMP-FL 95%和CEMP-FL 99%）进行性能对比。

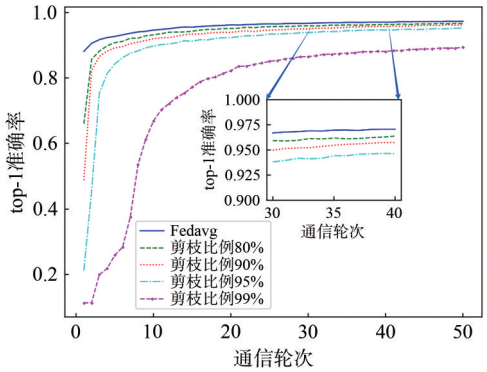
Fedavg 算法与CEMP-FL 算法不同剪枝比例下的通信轮次与top-1 准确率关系、通信成本与top-1 准确率关系的实验结果分别如图4(a)~(d)所示。

由图4(a)至图4(c)可以看出，在MNIST 数据集的两个深度模型情况下，不同剪枝比例CEMP-FL 算法（CEMP-FL 80%、CEMP-FL 90% 和 CEMP-FL 95%）相同通信轮次的准确率与Fedavg 算法基本持平或者性能略有下降，只有在深度剪枝情况即CEMP-FL 99%时，才出现相对于Fedavg 算法性能明显下降的现象，同样在CIFAR-10数据集和VGG-D模型情况下，只有在深度剪枝情况即 CEMP-FL 99%时，才出现相同通信轮次性能明显下降的现象，这说明CEMP-FL 算法在很宽泛的剪枝比例情况下都保持优异的性能，这特别适合在联邦学习架构下使用轻量化的模型从而实现通信高效的目的。最后，在图4(d)中，本文选用较为复杂的CIFAR-10数据集和VGG-D模型，讨论Fedavg算法和CEMP-FL算法在不同剪枝比例情况下通信成本和准确率的关系，本文发现除了深度剪枝情况CEMP-FL 99%，相同的通信成本，不同剪枝比例 CEMP-FL 算法（CEMP-FL 70%、CEMP-FL 80%、CEMP-FL 90%和CEMP-FL 95%）分类性能都要优于Fedavg 算法，同样也意味着对应相同的分类性能，不同剪枝比例CEMP-FL 算法都只需要相对较小的通信成本，这再一次验证了CEMP-FL 算法在联邦学习架构下具有良好的通信效率。

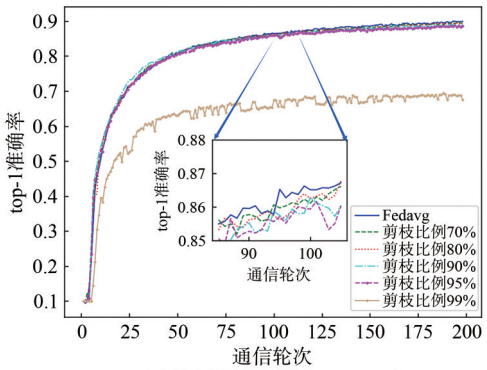
为了进一步验证CEMP-FL算法在联邦学习架构下的通信高效，CEMP-FL 算法将与另外5种联邦学习架构下的剪枝算法（Fedavg、FedSparsify-Global、一次性剪枝算法OMP、Sub-FedAvg和Ditto）进行通信性能对比，其中，通信成本比定义为基准Fedavg通信成本与待评价算法通信成本之比，↑表示越大越好。不同算法通信成本对比（MNIST 数据集+LeNet_5_Caffe 模型）见表5，不同算法通信成本对比（MNIST 数据集+LeNet_5_Caffe 模型）见表6，不同算法通信成本对比（CIFAR-10数据集和VGG-D



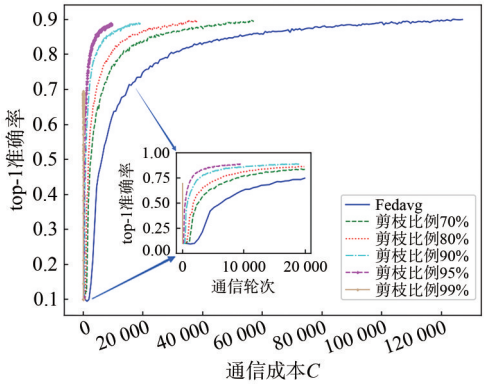
(a) Fedavg与不同剪枝比例的CEMP-FL在MNIST数据集LeNet_5_Caffe模型上的通信轮次与top-1准确率关系



(b) Fedavg与不同剪枝比例的CEMP-FL在MNIST数据集LeNet_300_100模型上的通信轮次与top-1准确率关系



(c) Fedavg与不同剪枝比例的CEMP-FL在CIFAR-10数据集VGG-D模型上的通信轮次与top-1准确率关系



(d) Fedavg与不同剪枝比例的CEMP-FL在CIFAR-10数据集VGG-D模型上的通信成本*C*与top-1准确率关系

图4　Fedavg算法与CEMP-FL 算法不同剪枝比例下的通信轮次与top-1准确率关系、通信成本与top-1准确率关系

模型）见表 7，其中标明了各剪枝算法总的通信成本、通信成本比和准确率。

表 5　不同算法通信成本对比（MNIST 数据集和 LeNet_5_Caffe 模型）

模型	总的通信成本/MB	通信成本比（↑）	准确率
Fedavg	205.70	1.00	98.02%
CEMP-FL80%	78.54	2.62	98.01%
CEMP-FL90%	61.71	3.33	98.02%
CEMP-FL95%	44.88	4.58	98.02%
CEMP-FL99%	28.06	7.34	97.79%
FedSparsify-Global	161.57	1.27	98.01%
OMP	88.83	2.23	98.02%
Sub-FedAvg	136.20	1.51	97.79%
Ditto	171.42	1.20	97.88%

表 6　不同算法通信成本对比（MNIST 数据集和 LeNet_300_100 模型）

模型	总的通信成本/MB	通信成本比（↑）	准确率
Fedavg	209.00	1.00	96.07%
CEMP-FL80%	108.90	1.92	96.03%
CEMP-FL90%	82.50	2.53	96.11%
CEMP-FL95%	40.43	5.17	95.02%
CEMP-FL99%	8.25	25.30	89.16%
FedSparsify-Global	151.36	1.38	96.10%
OMP	106.70	1.96	96.03%
Sub-FedAvg	138.40	1.51	95.57%
Ditto	154.66	1.35	96.03%

表 7　不同算法通信成本对比（CIFAR-10 数据集和 VGG-D 模型）

模型	总的通信成本/MB	通信成本比（↑）	准确率
Fedavg	124 412.20	1.00	90.01%
CEMP-FL80%	66 663.10	1.87	90.01%
CEMP-FL90%	42 133.40	2.95	90.02%
CEMP-FL95%	26 261.20	4.74	90.01%
CEMP-FL99%	14 429.30	8.62	90.07%
FedSparsify-Global	54 526.78	2.28	90.05%
OMP	68 939.75	1.80	90.00%
Sub-FedAvg	26 703.04	4.65	90.01%
Ditto	108 860.68	1.14	90.05%

由表 5 可以看出，当不同剪枝比例 CEMP-FL 算法在准确率与其他算法基本一致时，通信成本比高于其他算法，比如 CEMP-FL 99% 通信成本比达到了 7.34，CEMP-FL 95% 通信成本比达到了 4.58，远大于其他联邦学习剪枝算法，说明 CEMP-FL 算法在确保算法分类准确率的前提下，相对于其他剪枝算法显著地提升了通信效率，需要注意的是，由于 CEMP-FL 99% 的准确率没有办法进一步提升，CEMP-FL 95% 在保证分类准确度的情况下取得了最大的通信成本比和通信有效性。

从表 6 和表 7 可以看出，在两个数据集的两个深度模型情况下，不同剪枝比例 CEMP-FL 算法在准确率与其他算法基本一致时，通信成本比明显高于其他算法，比如 MNIST 数据集和 LeNet_300_100 模型上，CEMP-FL 99% 因分类准确度有所下降而排除在外，CEMP-FL 95% 取得了 5.17 的最大成本比；在 CIFAR-10 数据集和 VGG-D 模型上，CEMP-FL 99% 在准确度达到 90.07% 的情况下，取得了 8.62 的最大成本比，通信效率的提高非常明显。

通过上述实验，本文提出的 CEMP-FL 算法采用粗剪枝结合精细剪枝的方法，并结合网络剪枝的层平衡策略和稀疏参数压缩，全局汇聚模型在保证较高分类性能的同时，通过模型轻量化方法有效地降低了通信成本，实现了在联邦学习框架中的高效通信。

6　结束语

本文针对无线场景中联邦学习通信效率问题，提出了通信高效的联邦学习模型剪枝架构和单次层平衡网络剪枝算法，通过不同粒度的剪枝策略，结合稀疏参数压缩方法，减少了通信过程中传输的深度模型参数量，同时有效地减少了终端侧训练样本分布差异所带来的剪枝偏差。更进一步，设计了网络剪枝的层平衡策略，在考虑层间参数相对平衡的情况下，当稀疏度很大时，有效地推迟了深度模型坍塌。综上，通过上述方法有效地提高了联邦学习的通信效率，保证了联邦学习在无线场景中地有效部署。后续工作将在联邦学习架构下，在 Transformer 等大模型中设计更有效的剪枝策略。

参考文献：

[1] MCMAHAN B, MOORE E, RAMAGE D, et al. Communication-efficient learning of deep networks from decentralized data[C]//Proceedings of the Artificial Intelligence and Statistics. PMLR, 2017: 1273-1282.

[2] YANG Q, LIU Y, CHEN T J, et al. Federated machine learning: concept and applications[J]. ACM Transactions on Intelligent Systems and Technology, 10(2): 1-19.

[3] DUAN M M, LIU D, CHEN X Z, et al. Self-balancing federated learning with global imbalanced data in mobile systems[J]. IEEE Transactions on Parallel and Distributed Systems, 2021, 32(1): 59-71.

[4] KONEČNÝ J, MCMAHAN H B, YU F X, et al. Federated learning: strategies for improving communication efficiency[J]. arXiv preprint, 2016, arXiv: 1610.05492.

[5] HAMER J, MOHRI M, SURESH A T. Fedboost: a communication-efficient algorithm for federated learning[C]// International Conference on Machine Learning. PMLR, 2020: 3973-3983.

[6] LEE N, AJANTHAN T, TORR P H S. SNIP: single-shot network pruning based on connection sensitivity[J]. arXiv preprint, 2018, arXiv: 1810.02340.

[7] ZHU M, GUPTA S. To prune, or not to prune: exploring the efficacy of pruning for model compression[J]. arXiv preprint, 2017, arXiv: 1710.01878.

[8] SINGH S P, JAGGI M. Model fusion via optimal transport[J]. arXiv preprint, 2019, arXiv: 1910.05653.

[9] PALIHAWADANA C, WIRATUNGA N, WIJEKOON A, et al. FedSim: similarity guided model aggregation for Federated Learning[J]. Neurocomputing, 2022, 483(C): 432-445.

[10] LI T, HU S, BEIRAMI A, et al. Ditto: fair and robust federated learning through personalization[C]// International Conference on Machine Learning. PMLR, 2021: 6357-6368.

[11] MARFOQ O, NEGLIA G, BELLET A, et al. Federated multi-task learning under a mixture of distributions[J]. Advances in Neural Information Processing Systems, 2021, 34: 15434-15447.

[12] XIE C, HUANG K, CHEN P Y, et al. DBA: distributed backdoor attacks against federated learning[C]// International Conference on Learning Representations. 2020.

[13] YIN H X, MALLYA A, VAHDAT A, et al. See through gradients: image batch recovery via GradInversion[C]//Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Piscataway: IEEE Press, 2021: 16332-16341.

[14] LI Z H, ZHANG J X, LIU L Y, et al. Auditing privacy defenses in federated learning via generative gradient leakage[C]//Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Piscataway: IEEE Press, 2022: 10122-10132.

[15] FRABONI Y, VIDAL R, KAMENI L, et al. Clustered sampling: Low-variance and improved representativity for clients selection in federated learning[C]// International Conference on Machine Learning. PMLR, 2021: 3407-3416.

[16] BALAKRISHNAN R, LI T, ZHOU T, et al. Diverse client selection for federated learning via submodular maximization[C]// International Conference on Learning Representations. 2022.

[17] TANAKA H, KUNIN D, YAMINS D L K, et al. Pruning neural networks without any data by iteratively conserving synaptic flow[C]// Proceedings of the Proceedings of the 34th International Conference on Neural Information Processing Systems. New York: ACM, 2020: 6377-6389.

[18] RAMANUJAN V, WORTSMAN M, KEMBHAVI A, et al. What's hidden in a randomly weighted neural network? [C]//Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Piscataway: IEEE Press, 2020: 11890-11899.

[19] PATIL S M, DOVROLIS C. PHEW: constructing sparse networks that learn fast and generalize well without training data[C]// International Conference on Machine Learning. PMLR, 2021: 8432-8442.

[20] LIN J, LUO X T, HONG M, et al. Memory-friendly scalable super-resolution via rewinding lottery ticket hypothesis[C]//Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Piscataway: IEEE Press, 2023: 14398-14407.

[21] SRINIVAS S, SUBRAMANYA A, BABU R V. Training sparse neural networks[C]//Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). Piscataway: IEEE Press, 2017: 455-462.

[22] FRANKLE J, DZIUGAITE G K, ROY D M, et al. Linear mode connectivity and the lottery ticket hypothesis[C]//Proceedings of the Proceedings of the 37th International Conference on Machine Learning. New York: ACM, 2020: 3259–3269.

[23] YUAN G Z, SHEN L, ZHENG W S. A block decomposition algorithm for sparse optimization[C]//Proceedings of the Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. New York: ACM, 2020: 275-285.

[24] HE Y, DING Y H, LIU P, et al. Learning filter pruning criteria for deep convolutional neural networks acceleration[C]//Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Piscataway: IEEE Press, 2020: 2006-2015.

[25] HU R, GONG Y M, GUO Y X. Federated learning with sparsification-amplified privacy and adaptive optimization[C]// Proceedings of the Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence. California: International Joint Conferences on Artificial Intelligence Organization, 2021: 1463-1469.

[26] STRIPELIS D, GUPTA U, STEEG G V, et al. Federated progressive sparsification (purge, merge, tune) [J]. arXiv preprint, 2022, arXiv: 2204.12430.

[27] QIU X C, FERNANDEZ-MARQUES J, GUSMAO P P, et al. ZeroFL: efficient on-device training for federated learning with local sparsity[EB/OL]. 2022: arXiv: 2208.02507. http://arxiv. org/abs/2208.02507

[28] PASE F, ISIK B, GUNDUZ D, et al. Efficient federated random subnetwork training[C]// Workshop on Federated Learning: Recent Advances and New Challenges (in Conjunction with NeurIPS 2022). 2022.

[29] CHEN X Z, ZHU J Y, JIANG J B, et al. Tight compression: com-

pressing CNN model tightly through unstructured pruning and simulated annealing based permutation[C]//Proceedings of the 2020 57th ACM/IEEE Design Automation Conference (DAC). Piscataway: IEEE Press, 2020: 1-6.

[30] SONG HAN, HUIZI MAO, AND WILLIAM J DALLY. Deep compression: compressing deep neural networks with pruning, trained quantization and huffman coding. In International Conference on Learning Representations, 2016

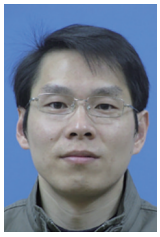
[31] VAHIDIAN S, MORAFAH M, LIN B. Personalized federated learning by structured and unstructured pruning under data heterogeneity[C]//Proceedings of the 2021 IEEE 41st International Conference on Distributed Computing Systems Workshops (ICDCSW). Piscataway: IEEE Press, 2021: 27-34.

[作者简介]

胡海峰（1973– ），男，博士，南京邮电大学通信与信息工程学院教授，主要研究方向为人工智能、网络信息处理等。

张熙（1999– ），男，南京邮电大学通信与信息工程学院硕士生，主要研究方向为联邦学习、模型剪枝、移动边缘计算等。

赵海涛（1983– ），男，博士，南京邮电大学物联网学院院长，主要研究方向为车联网络、卫星物联网、工业互联网等。



吴建盛（1979– ），男，博士，南京邮电大学计算机学院教授，主要研究方向为人工智能药物设计、软硬件协同加速等。