分类号：按中国图书分类法，学位办网上可查　　单位代码：10335

密　级：注明密级与保密期限　　学　　号：22221351

浙江大学

硕士学位论文

中文论文题目：基于子模型抽取的资源受限场景下联邦学习

英文论文题目：Federated Learning with Submodel Extraction under Resource Constraints

申请人姓名：贾亚博

指导教师：庞超逸

合作导师：陈根浪

专业学位类别：电子信息

专业学位领域：计算机技术

所在学院：计算机科学与技术学院

论文递交日期　　2024.12.05

基于子模型抽取的资源受限场景下联邦学习

论文作者签名： _____

指导教师签名： _____

论文评阅人 1：_____ 姓名 _____

评阅人 2：_____ 姓名 _____

评阅人 3：_____ 姓名 _____

评阅人 4：_____ 姓名 _____

评阅人 5：_____ 姓名 _____

答辩委员会主席：_____

委员 1：_____

委员 2：_____

委员 3：_____

委员 4：_____

委员 5：_____

答辩日期 _____

Federated Learning with Submodel Extraction under Resource Constraints

Author's signature: _____

Supervisor's signature: _____

External reviewers: _____ Name _____

_____ Name _____

_____ Name _____

_____ Name _____

_____ Name _____

Examining Committee Chairperson:

_____

Examining Committee Members:

_____

_____

_____

_____

_____

Date of oral defence: _____

浙江大学研究生学位论文独创性声明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得 **浙江大学** 或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

学位论文作者签名: 签字日期： 年 月 日

学位论文版权使用授权书

本学位论文作者完全了解 **浙江大学** 有权保留并向国家有关部门或机构送交本论文的复印件和磁盘，允许论文被查阅和借阅。本人授权 **浙江大学** 可以将学位论文的全部或部分内容编入有关数据库进行检索和传播，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后适用本授权书）

学位论文作者签名: 导师签名:

签字日期： 年 月 日 签字日期： 年 月 日

勘误表

致谢

序言

摘要

随着智能设备和互联网的发展，终端设备产生了越来越多的数据，其中包含着各种各样的用户隐私。而如今模型的训练需要越来越多的数据，这些数据依赖于终端设备用户产生。随着公众对于数据安全的担忧加剧，将终端设备数据集中到服务器端变得异常困难。联邦学习（Federated Learning, FL）提出的非中心化训练框架有效地解决了此问题，在联邦学习中所有的终端设备合作训练模型，具体来说，终端设备使用本地数据训练模型，然后将训练完成的模型上传到服务器端聚合。然而，随着模型规模的不断增大，多数的终端设备受限于计算机资源与计算机性能不能完整的训练整个模型。这将导致很多数据无法参与模型训练。

为了解决资源受限引起的边缘设备无法训练模型的问题，提出了在边缘设备上训练全局模型的子模型的方法。本文基于子模型抽取方向提出了两种不同方法，分别解决了客户端分布质量不同场景下的资源受限联邦学习问题。首先，提出了基于数据分布的子模型抽取联邦学习范式（Distribution-aware Sub-model Extraction, FedDSE）。相较于根据预定策略抽取子模型引起的神经元分别朝不同方向优化所造成的竞争关系，根据客户终端设备数据的不同在全局模型上激活的神经元的不同来挑选激活值高的神经元，组成子模型，在每个训练轮次都能自动根据边缘设备数据分布不同抽取特定的子模型，有效解决了子模型训练造成的神经元冲突。经过大量实验证明，相较于其他方法 FedDSE 在低质量客户端分布场景下模型效果取得显著提升。其次，在高客户端分布场景中通过约束全局模型与子模型梯度更新方向，提出了一种基于梯度的子模型抽取的联邦学习（Grad-based Sub-model Extraction, FedGSE）。此方法考虑在每个轮次的更新中，挑选出全局模型在反向传播过程中梯度大的神经元来组成子模型，确保每次子模型的更新与全局模型的更新最接近，从而保证了子模型与全局模型朝着相同方向优化。在多个数据集上 FedGSE 对比以往的方法均有较大的提升。

关键词：联邦学习；子模型抽取；分布感知；梯度更新优化

Abstract

With the development of smart devices and the internet, terminal devices generate increasing amounts of data, which contain various types of user privacy. At the same time, model training requires more and more data, which depends on the data generated by users of terminal devices. As public concerns about data security intensify, centralizing terminal device data on servers has become exceptionally difficult. Federated Learning (FL), a decentralized training framework, effectively addresses this issue. In FL, all terminal devices collaborate to train the model, where each device uses local data to train the model, and the model is aggregated on the server. However, as the model parameters continue to grow, many terminal devices, limited by their computing resources and performance, are unable to fully train the entire model. This results in much data being excluded from the model training process.

To address the issue of edge devices being unable to train models due to resource limitations, mothod for training global model's sub-models on edge devices have been proposed. This paper presents two different methods based on the sub-model extraction approach, each solving the resource-constrained federated learning problem in scenarios with varying client distribution quality. we propose Distribution-aware Sub-model Extraction for FL (FedDSE). Unlike the competition that arises from neurons being optimized in different directions when sub-models are extracted based on predefined strategies, FedDSE selects neurons with higher activation values on the global model based on the data distribution of client devices. It forms sub-models by choosing neurons that are activated differently across devices. This method automatically extracts specific sub-models in each training round based on the varying data distributions of edge devices, effectively solving the neuron conflicts caused by sub-model training. Extensive experiments demonstrate that FedDSE achieves significant improvements over other methods in scenarios with low-quality client distributions. Secondly, in high-quality client distribution scenarios, we propose a gradient-based sub-model extraction method (FedGSE), which constrains the gradient update directions of the global model and sub-models. In each round of updates, this method selects neurons with large gradients during backpropagation from the global model

to form sub-models. This ensures that the updates of the sub-models align closely with the global model updates, thus optimizing both the sub-model and the global model in the same direction. Through comparisons with recent methods on multiple datasets, FedGSE shows significant improvements over traditional approaches.

Keywords: Federated Learning; Resource constrained; Sub-model Extraction; Distribution-aware; Gradient Update Optimization

缩略词表

英文缩写	英文全称	中文全称
FL	Federated Learning	联邦学习
Non-IID	Non-Independent and Identically Distributed	非独立同分布
FedDSE	Distribution-aware Sub-model Extraction	基于数据分布子模型抽取
FedGSE	Grad-based Sub-model Extraction）	基于梯度子模型抽取
BP	BackPropagation	反向传播算法
SGD	Stochastic Gradient Descent	随机梯度下降
CSL	Client sub-model to Set Labels	客户子模型设定标签
SPL	Server Picks samples through Labeled dataset	服务器挑选标签
MACs	Multiply-Accumulate Operations	乘加运算数
AD	Accumulated Differences	累计误差

目录

图目录

表目录

1 绪论

1.1 课题的研究背景与意义

随着智能化设备的爆发，全球每天产生海量的数据，这些数据来自大量的手机终端、传感器与其他的各种终端信息。在海量信息的基础之上，深度学习模型蓬勃发展。深度学习（Deep Learning）模型[1-2]一般需要喂养大量规范化数据来保证模型可以学习到有用的知识。深度学习预训练模型已经应用到社会的方方面面，包括人脸识别[3-5]、自动驾驶[6-7]、生成式对话智能体[8-9]、医疗[10-12]、金融[13-14]等等方向，给人类社会的发展带来了巨大的便利。

模型的能力很大程度上取决于数据的质量，然而这些数据的提供者大部分是来自于各种边缘设备，现存的训练方法需要将客户产生数据集在中心聚集[15]，然后在中心训练完整模型，如图1.1所示，客户端在中心化训练模型方式中仅仅需要将自己产生的数据收集，然后上传给中心服务器即可。诚然，基于中心化的训练方式可以大大提高模型训练效率，但是也暴露了严重的隐私安全问题——所有的终端产生的数据都要在中心侧保存处理，难免出现隐私泄露等等一系列问题[16]。中国在 2021 年制定了《中华人民共和国个人信息保护法》[17]明确了个人信息的收集、存储、使用、加工、传输、提供、公开等各环节的合规要求。法律还规定了个人信息处理者的义务，以及对违法行为的处罚措施，旨在保护公民的隐私权。《欧盟通用数据保护条例》[18]（GDPR），该条例对个人数据的收集、存储、处理和传输提出了严格的要求，违规者将面临高额罚款。GDPR 对全球范围内的信息安全管理产生了深远影响。为了解决上述的隐私安全问题，谷歌在 2017 年提出了一种全新的深度模型训练的范式——联邦学习[19-20]（Federated Learning, FL）。联邦学习是一种协同多个边缘设备训练模型，最终在中心服务器上聚合模型更新参数的一种保护隐私的训练范式。如图1.1所示，在联邦学习的框架中，客户端本地的数据不需要上传到服务器端，随之而来的便是，模型的训练也同样需要在本地训练，最终模型参数在中心侧聚合。其核心之处在于，在不直接将边缘设备数据上传的情况下，实现了在中心侧模型的训练。这也使得联邦学习广泛的应用在医疗[21-25]、金融[26-29]等高度注重隐私保护且又需要深度学习模型赋能的场景。

虽然联邦学习可以跳过数据聚集过程而直接在端侧训练模型，然后更新参数到中心侧，但是在实际过程中存在诸多挑战。其中重要的一个是不同边缘设备产生的数据异构性问题。在传统的中心化训练模型的过程中，我们将数据中的每个训练批次随机划分，保证我们每次训练数据分布相似，也就是数据同构。但是在联邦学习的范式中，数据孤立分布在边缘设备，很容易形成非独立同分布的数据集[30]（Non-Independent and Identically Distributed, Non-IID）。这将导致边缘设备训练的模型具有自己数据分布的参数特征[31-33]，在聚合过程中导致中心侧的模型泛化能力不行。例如在金融场景中，由于地理位置分布迥异，在西部地区训练出来的模型与在东部沿海地区的模型必然存在显著的差异，导致在聚合过程中影响整体模型的泛化能力，既不能很好的处理西部金融数据，也不能处理东部数据的两难问题。

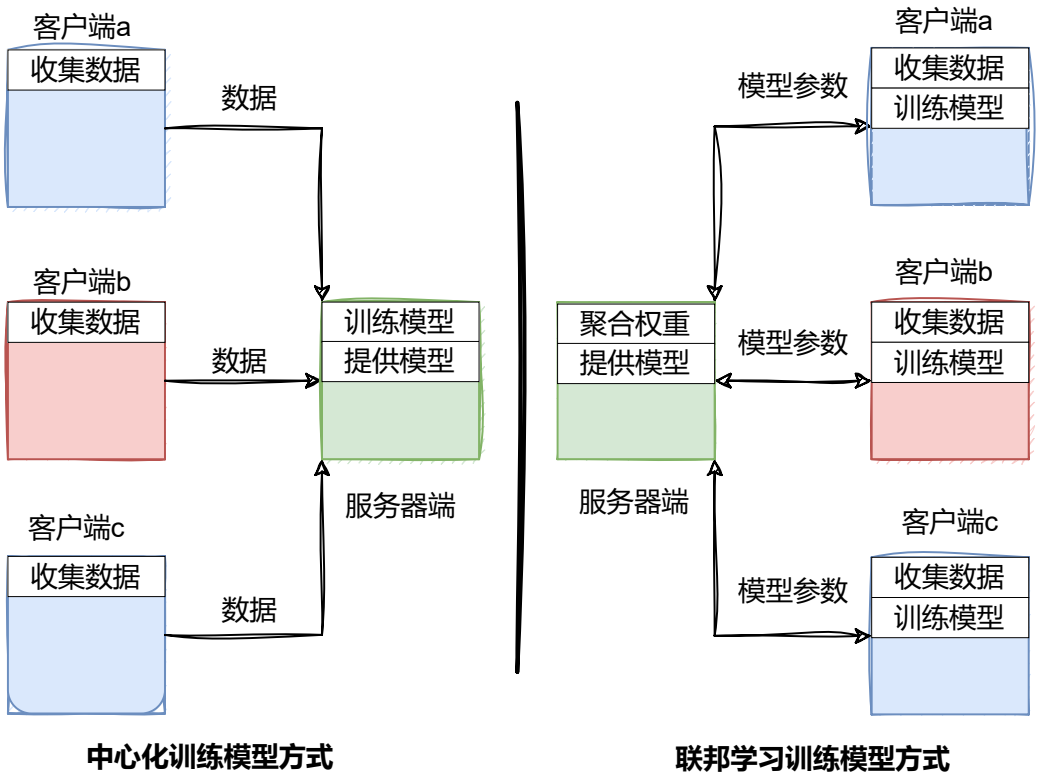


图 1.1 中心训练与联邦学习对比

随着生成式模型的广泛应用，如今深度学习模型的参数规模越来越大，另一个问题也随之而来。边缘设备受限于资源限制已经无法承受训练现今越来越庞大模型[34-38]，然而边缘设备的数据资源对于整个模型的训练有这重要的贡献。这些问题主要集中在：

- 计算资源受限[34-35,38]。现在的模型参数在千万百万级别都有，在一些比如手机、性能老旧的电脑、汽车传感器等等边缘设备上匮乏的资源计算空间与计算能力均不能支撑庞大的训练目标。
- 通讯成本限制[39-40]。在联邦学习中边缘设备需要频繁与中心设备共享参数等信息，庞大的模型参数以为着在每个训练轮次都需要进行大量的高带宽的通讯，对于整个训练来说是庞大的开销。
- 隐私保护带来的数据异质性。由于数据不能在中心侧聚合导致每个终端都有自己特质的数据带来的训练偏差问题。

上述的问题本质都是异质性的问题，分别是设备资源异质性与数据资源异质性，并且二者往往是同时出现的。也是限制现阶段联邦学习发展的主要的瓶颈问题。解决资源异质性问题为联邦学习在不同计算能力的终端侧应用铺平道路，使得联邦学习的训练框架可以应用到各式各样的设备上，不在因为计算资源不足的原因导致很多终端设备的数据不能参与训练。另一方面，解决数据异质性的问题使得联邦学习得到的模型的结果不在偏向于某些特定数据，而是具有更好的泛化性，训练出的模型更能集合所有数据的特征。

1.2 国内外研究现状

时至今日，人工智能迅猛发展，保护隐私的分布式训练范式联邦学习成为了人们研究的热点，其中的热点问题数据异构和资源受限更是最新急需解决的重要问题。Transformer 架构[41]的出现更是将模型的规模提升到更高的级别，并且训练数据的规模也得到了前所未有的增长。这些架构在计算机视觉[42]，自然语言处理[43]取得了重大的成功，模型规模的扩大，训练数据的增多对于联邦学习来说是一个巨大的挑战。本章首先介绍传统的联邦学习，然后介绍资源受限联邦学习的处理方法，同时介绍数据异构的解决方案。

1.2.1 经典联邦学习方法

传统联邦学习是一种分布式机器学习技术，其核心思想是在多个拥有本地数据的数据源之间进行分布式模型训练，而无需交换本地数据。各方仅通过交换模型参数或中间

表 1.1 不同用户的兴趣设备信息

用户	用户 A	用户 B	用户 C
设备	智能手机	个人电脑	智能汽车
运行内存	8GB	32GB	8GB
计算能力	低	中	低
兴趣	猫，松鼠， 狗等小动物	黑神话悟空， 穿越火线等游戏	奔驰，理想， 特斯拉等汽车

结果来构建基于虚拟融合数据下的全局模型，从而实现数据隐私保护和数据共享计算的平衡。传统的联邦学习的数据分布基于一个假设就是所有参与者提供的数据拥有相似的数据分布结构，也就意味着所有边缘设备训练的数据是相近的。另外一个默认的事实是，传统的联邦学习中每个边缘设备中训练的模型都是一模一样的。我们使用表1.1不同用户设备和兴趣信息来说明传统联邦学习在实际中遇到的困难，与资源受限下联邦学习有明显的区别：

- 边缘设备数据分布差异大。如表1.1所示三个用户感兴趣的部分非常具有自己的特色，用户 A 对小动物感兴趣，A 的数据分布对于动物数据占据大部分，使用 A 的数据训练的模型则偏向动物数据特征；同理，根据 B、C 的数据训练的模型也偏向各自的数据分布。例如，用户 B 的数据偏向游戏，而用户 C 则是偏向汽车领域。最终的模型则受到数据特征的影响泛化性下降。

- 边缘设备计算资源差距大。在表1.1中所示三个用户使用的设备可以看出用户 A 使用的是智能手机[44]在运行内存还有是否存在可计算 GPU 上均不如用户 B 的个人电脑。也就意味着用户 B 可以训练比用户 A 更大的模型。反之，如果模型参数过大，用户 A 没有计算资源去训练中心侧需要的模型。

1.2.2 基于知识蒸馏的资源受限联邦学习

知识蒸馏是一种有效的方法来解决边缘设备的异构性[45]。具体而言，FedDF[46]利用知识蒸馏从一组使用客户端私有数据训练的本地模型中提取知识。每个本地模型在未标记的公共数据集上的逻辑输出随后被用于通过知识蒸馏在服务器上训练学生模型。类似

地，DS-FL[47]在服务器上采用未标记的公共数据集，并引入了一种基于蒸馏的半监督联邦学习。该技术旨在通过为公共数据生成伪标签来提高性能。FedGKT[48]引入了组知识转移，即使在客户端没有任何公共数据的情况下，也能有效地将知识从小型客户端模型转移到服务器上的大型模型。Fed-ET[49]提出了一种加权共识蒸馏方案，并结合了多样性正则化。该方案通过利用小型客户端模型的知识来训练大型服务器模型。

知识蒸馏的方法能解决计算资源异构的问题，但是由于在边缘设备需要选择合适大小的异构模型，然而当前存在的模型不一定能最佳适配边缘设备计算资源的情况，可能导致计算资源的浪费；另一方面，知识蒸馏的方法需要中心侧拥有一定量的边缘设备数据相似的数据，这些数据可以是有标签和无标签的。

1.2.3 基于子模型抽取的资源受限联邦学习

除了知识蒸馏方法之外，对于边缘设备没有足够的计算资源来训练中心侧的全局模型的问题，最简单的方法便是对于每个边缘设备按照其计算能力的大小从全局模型中抽取相应大小的子模型以供边缘设备计算。2021 年 Diao 等人提出了一种基于预先分配规则的子模型抽取方法 HeteroFL[50], 对于深度学习模型此方法依据边缘设备的计算能力按照全局模型每层神经元的先后顺序抽取前面的神经元组成子模型层，每层按照上述的思路进行抽取，最后组成完整的子模型发送给边缘设备。如图1.2中最下方所示，每个轮次中同一个客户端所获得的子模型都是相同的构造，不会因为通讯轮次不同而发生改变，图中不管是第 J 轮或者是 $J+1$ 轮都是 $\{0,1,2\}$（上面客户端）和 $\{0,1,2,3\}$（下面客户端）参与训练并且保持不变。HeteroFL 横向的抽取每层的神经元巧妙的解决了边缘设备无法计算全局模型的困境，并且可以根据边缘设备自己的计算能力设定需要抽取子模型的大小，具有一定的自适应性。同期的方法 FjORD[51]使用了和 HeteroFL 相同的思想来解决资源受限的联邦学习场景。这种方法有个很明显的缺点，处于后面的神经元在整个联邦学习的训练过程中被训练的次数远少于位于每层前面的神经元。在图1.2所示中，最后一个神经元永远没有得到训练的机会。

为了解决上述按照顺序挑选神经元的缺点，一种简单的按照深度学习模型每层随机挑选神经元的方法被提出了。简单来说就是在每层的基础上使用随机的方法根据边缘设备的能力来挑选神经元，我们称这种方法为 Federated Dropout[52]。这种方法大致解决了每个神经元训练次数不一致的问题，但是这种随机的方法不能保证每个神经元训练的次

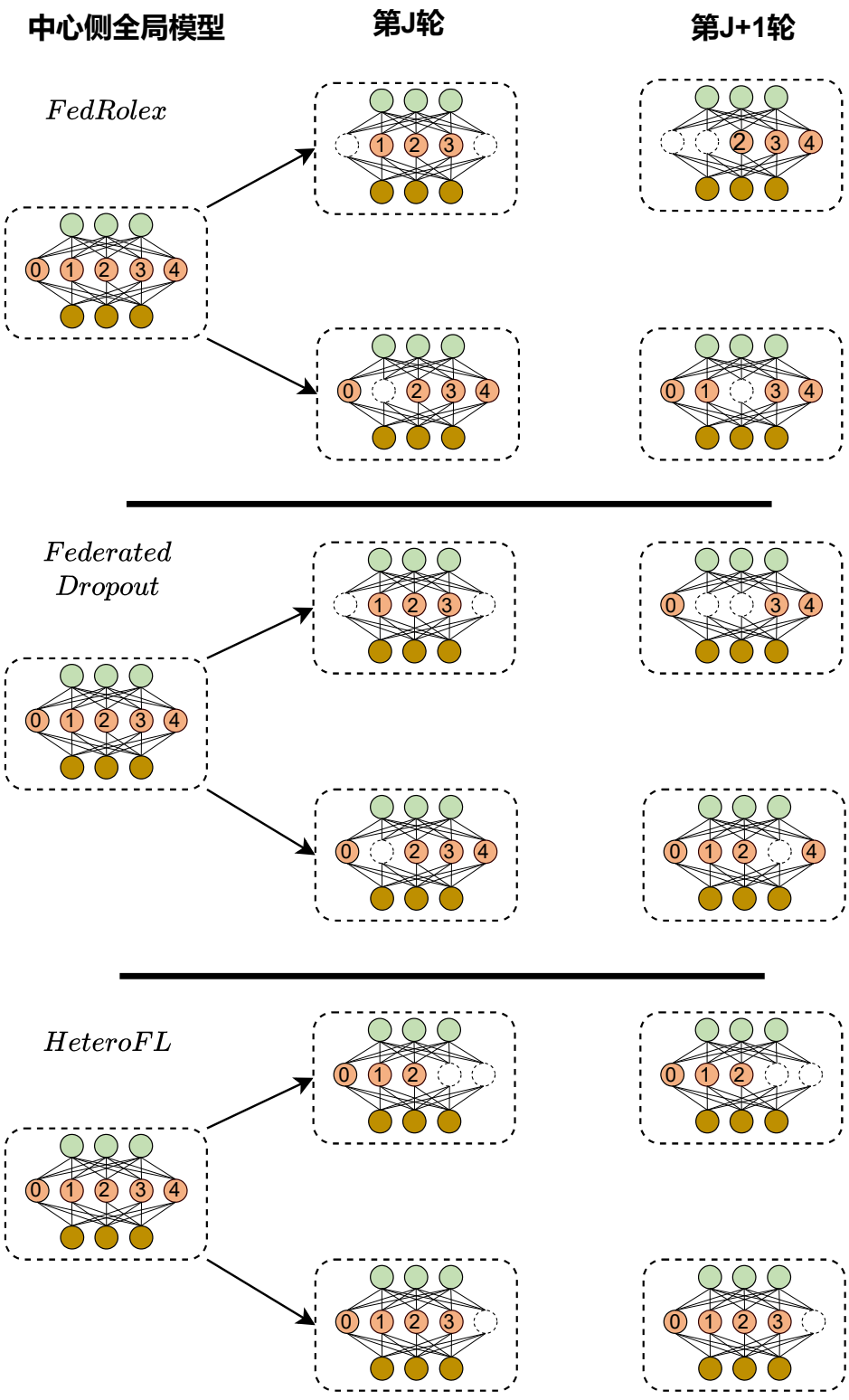


图 1.2 子模型抽取的三种方法

数一定是相同的，只是在概率上保证大概率是一样的。如图1.2中所示，在不同轮次的抽取过程中，相同两个轮次之间的子模型抽取没有任何的规律可言，都是通过随机数产生的。也就是 Federated Dropout 方法存在很大的不稳定性，模型训练的结果存在一定的随机性。

Alam 等人提出了一种解决基于顺序方法训练不均的方法叫做 FedRolex[53], 此方法解决了 HeteroFL 与 FjORD 方法中的问题。FedRolex 不在是每个边缘设备都是固定的按照顺序的分配子模型，而是在 HeteroFL 的基础上，按照当前训练的轮次，在上一个轮次的基础上将每层挑选的神经元的序号向后平移一个单位，这样在每个轮次训练过程中，训练的神经元都是上一个轮次训练的下一个序号的神经元，当序号达到末尾的时候就挑选头部的神经元。保证了每个神经元都可以均匀的参与到模型的训练过程中。如图1.2中所示，上一轮的神经元序号在本层平移到一下个序号上，在图中就是从序号 $\{1,2,3\}$ 变成了挑选序号为 $\{2,3,4\}$ 的神经元，保证了每个神经元都可以得到平均的训练。

上述所提及到的方法属于在深度学习模型的横向进行了子模型抽取，而 Kim 等人在 2023 年提出了基于深度抽取子模型的方法[54]。此方法不同于上述的方法在横向在每层的维度上抽取特定的神经元，而是根据边缘设备的计算能力在深度上抽取不同的层来组成子模型。按照深度方向抽取有一些缺点，不能灵活的按照边缘设备的计算能力抽取子模型，因为深度学习模型多个层组层一个逻辑上的块，所以应该在块的维度上抽取，这样会导致子模型的抽取不自由。

1.3 本文研究内容

本文主要研究边缘设备计算能力受限情况下联邦学习框架研究，主要是基于子模型抽取这一类方法的延续，并在子模型抽取的过程中考虑到边缘设备数据异构分布做出优化。我们可以看到在以上的子模型抽取的方法中，都是根据预先制定好的规则去实现每个轮次如何挑选子模型，也就是意味着在实验前我们可以预测每个轮次所抽取的子模型，这种设计没有根据边缘设备数据的异质性，本文主要将预先设定的抽取规则转变为自适应的抽取规则，具体的研究内容如下：

- 本文在子模型抽取的所有方法中，首先提出了动态的在每轮通讯中自适应的去抽取最合适子模型的方法，改进了预先制定规则方法，提高了资源受限联邦学习场

景下训练的模型上限。

- 本文提出了一种基于边缘设备数据分布来动态抽取子模型的方案 FedDSE。具体来讲 FedDSE 摆脱了提前定制的抽取方案，而是根据边缘设备在自己终端处数据推理运算得出的神经元激活值作为挑选子模型的标准。神经元在不同数据集上的表现出不同的激活值，激活值表示了对于当前数据的敏感程度，筛选出激活值大的神经元作为子模型的组成部分，可以避免神经元在训练过程中的冲突情况，从而取得较好的效果。并且充分考虑了联邦学习在实际情况中的数据异质性问题。

- 从子模型与中心侧模型优化方向作为出发点，本文又提出了一种基于梯度的联邦学习框架。在 FedDSE 中我们考虑了边缘设备数据异构的情况，基于此提出了激活值作为参考挑选神经元的思路，在此基础上提出了基于反向传播中梯度挑选神经元的方法 FedGSE，以梯度为考虑要点可以是参数优化方向与中心侧模型一致。此方法首先在中心侧构建了一个全局数据集，这个数据集放在中心侧，然后筛选出合适部分的子数据集在全局模型上反向传播得到梯度，计算出的梯度作为挑选标准筛选子模型，由于反向传播过程需要庞大的计算能力，所以将这一过程在中心侧实现，全局数据集正好可以作为反向传播使用的数据。

1.4 本文结构安排

本文共包括五个章节，每个章节的主要内容如下：

第一章：绪论。绪论主要介绍了联邦学习出现的背景以及要解决的问题，联邦学习发展至今遇到的关键技术难题；然后介绍了国内外解决技术难题的论文与相关工作与不足之处；最后引申出本文主要做出的贡献以及成果。

第二章：理论技术与公式化表达。本章主要介绍本文中用到的相关技术的理论化表达以及相关的优化目标，为本文的表达提供前置知识。首先介绍联邦学习经典的联邦学习以及共同的优化目标函数，然后介绍本文中需要训练的深度学习模型卷积神经网络以及神经元的定义。

第三章：FedDSE 方法详细阐述。本章将介绍本文提出的方法 FedDSE。首先会介绍在以往的方法中的问题做出分析，通过思想实验与简单的预先实验来揭示此前方法中存在的问题，然后提出 FedDSE 方法详细实现方式完整的讲述基于激活值的子模型抽取方

法，在详尽的实验中证明方法的可靠性与优越性，通过大量的消融实验证明参数的重要程度，最后总结本章方法的优缺点。

第四章：FedGSE 方法详细阐述。本章内容在第三章的基础上做出改进解决了资源受限的问题，从一个新的视角优化策略方向考虑解决问题的方式，之后详尽的介绍了 FedGSE 方法公共数据集的收集，相似数据生成，子模型筛选等具体细节，然后从理论上分析了此方法的可行之处，最后通过相关的实验证明了此方法。并通过消融实验对比了众多参数的重要性程度。最后总结方法的优缺点。

第五章：总结与展望。本章概括性的总结了本文方法的技术要点与创新性的地方，并且提出了方法的不足之处与未来需要改进的地方。为该领域的发展提供了后续研究发展的启发。

2 理论技术与公式化表达

本章将详细从基础理论知识与应用方式介绍联邦学习、深度神经网络以及子模型全局优化三个重要的知识。上述的技术是理解后续论文的重要基础，构成了本文的研究核心部分，对于学习全文具有重要的意义。

2.1 联邦学习公式化与优化目标

联邦学习[19]是由谷歌在 2017 年提出的一种创新性的机器学习范式。联邦学习是一种服务器-客户端模式的训练方式，众多客户端负责训练模型，在服务器端聚合全局模型。与传统的分布式学习，联邦学习允许地理位置分散的多台客户端设备协作优化共同的全局模型。通过将训练过程放在客户端进行，就实现了联邦学习的重要目的，就是有效保护了客户端产生的训练数据。这些数据不会往服务器上传输，高效避免了隐私泄露。FedAvg 是由 McMahan 于 2017 年提出的联邦学习经典算法。其算法思想在于在服务器端使用加权平均的方法处理客户端传输的全局模型的梯度更新，从而多轮次更新服务器端的全局模型，直到模型收敛。如图2.1所示，FedAvg 算法在第 t 轮次的执行流程可以分为四个阶段：

（1）全局模型广播。首先，中心服务器根据设定参数随机选择一定比例的边缘设备参与训练，被选中的边缘设备从中心服务器下载 t 轮次的最新全局模型参数 $\mathbf{W}_{t-1}$。参考图2.1中的①过程。

（2）边缘设备本地模型训练。边缘设备即客户端 n 从中心服务器得到 t 轮模型参数 $\mathbf{W}_t$ 之后，本地客户端模型载入 $\mathbf{W}_t$，然后使用客户端 n 本地数据集 x_n, y_n 训练模型。使用以下公式来更新本地模型：

$$\mathbf{W}_t^n = \mathbf{W}_t^n - \eta \nabla f_n(\mathbf{W}_t^n; x_n, y_n) \tag{2-1}$$

其中，η 为训练过程中的学习率，f_n 为本地模型的损失函数，$\mathbf{W}_t^n$ 表示经过本地数据集训练之后的客户端 n 的 t 轮次的模型参数。本地模型在训练的过程中也是经过多个轮次训练，并将最终的参数梯度更新上传到服务器。

（3）梯度更新上传。如图2.1所示中的②所示，每个边缘设备将训练完成的 $\mathbf{W}_t^n$ 上传到服务器端。

（4）中心侧参数聚合。中心服务器将收集到的所有参与训练的边缘客户端的梯度更新，然后在服务器端使用加权平均的方式将参数聚合成一个参数并更新为下一轮全局模型参数 $\mathbf{W}_{t+1}$。

$$\mathbf{W}_{t+1} = \sum_{n \in \mathcal{N}_t} \frac{D_n}{D} \mathbf{W}_t^n \tag{2-2}$$

其中 D_n 表示边缘设备 n 上的训练数据总数，$\mathcal{N}_t$ 表示在 t 轮次中所选择的参与训练的边缘设备集合，则：

$$D = \sum_{n \in \mathcal{N}_t} D_n \tag{2-3}$$

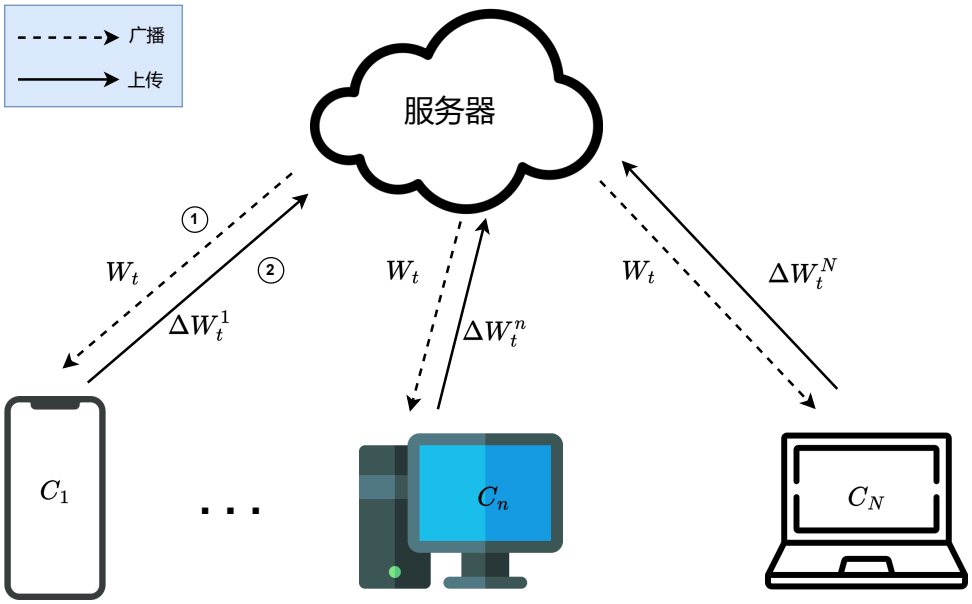


图 2.1 FedAvg 算法

联邦学习算法就是不断重复步骤（1）到步骤（4）重复 T 次，逐渐最小化下面的公式：

$$\min_{\mathbf{w}} F(\mathbf{w}) = \sum_{n \in \mathcal{N}_t} \frac{D_n}{D} F_n(\mathbf{w}) \tag{2-4}$$

$$F_n(\mathbf{w}) = \frac{1}{D_n} \sum_{i=1}^{D_n} f(\mathbf{w}; x_n^i, y_n^i) \tag{2-5}$$

其中，x_n^i 和 y_n^i 表示在边缘设备 n 上的第 i 条数据。由公式2-4和公式2-5可知，联邦学习的目标就是找到在多次重复训练上述步骤中使得在选中的边缘设备中目标损失函数的

值最小的模型的参数，并且可以观察到联邦学习的损失函数是由参与训练的边缘设备的损失函数加权平均组合而成的损失函数，边缘设备的损失函数在本地数据集上产生，这样避免了边缘设备的数据上传到中心侧，从而极大的保护了边缘设备的数据隐私安全。公式2-4和公式2-5 是所有联邦学习所优化的目标，而步骤（1）-（4）是联邦平均算法（FedAvg）的主要过程与相关的参数处理方法。以下是算法伪代码部分：

算法 2.1 FedAvg

Input: 全局模型参数 $\mathbf{W}$, 学习率 η, 总通讯次数 T, 本地客户端训练次数 E

Output: T 轮的全局模型参数 $\mathbf{W}_{T+1}$

初始化全局模型参数 $\mathbf{W}_1$
procedure SERVER-SIDE OPTIMIZATION
    for 对每个轮次 $t \in \{1, \cdots, T\}$ **do**
        随机挑选所有客户端的子集 $\mathcal{N}_t$
        for 对于每个挑选到的客户端 n 并行执行 **do**
            $\mathbf{W}_{t+1}^{n} \leftarrow ClientLocalUpdata(\mathbf{W}_t)$
        end for
        更新全局模型参数 $\mathbf{W}_{t+1} = \sum_{n \in \mathcal{N}_t} \frac{D_n}{D} \mathbf{W}_t^n$
    end for
end procedure
procedure CLIENTLOCALUPDATE($\mathbf{W}_t$)
    从服务器侧接收 $\mathbf{W}_t$
    初始化 $\mathbf{w_{t,0}^n} = \mathbf{W}_t$
    for 从 1 到 E 迭代轮次 e **do**
        更新本地模型 $\mathbf{w_{t,e+1}^n} = \mathbf{w_{t,e}^n} - \eta \nabla_{\mathbf{w_{t,e}^n}} f_n(\mathbf{w_{t,e}^n})$
    end for
    return $\mathbf{w_{t,E+1}^n}$
end procedure

其中函数 Server-side Optimization 运行在中心服务器端，总共运行 T 个轮次，在每个轮次随机选择跟下一轮次相同比例的客户端执行模型训练，也即执行 ClientLocalUpdata

程序，中心侧服务器端接收客户端程序返回的参数并进行加权聚合。在客户端侧从服务器端接收当前轮次 t 的全局模型参数，在本地数据集上对全局模型进行 E 个轮次的训练，将最终的训练结果上传到服务器端。

2.2 卷积神经网络相关知识

2.2.1 深度学习

深度学习，作为机器学习领域的一个关键分支，采纳了神经网络作为其核心架构，专注于从图像、语音、文本等多种数据类型中挖掘潜在的规律与特征。一个典型的深度学习框架，往往包括了输入层、多个中间隐藏层和输出层。如图2.2所示的是一个四层的全连接神经网络，包括了一个输入层、两个隐藏层和一个输出层，每个隐藏层包含八个神经元。与以往需要人工精心设计与提取特征的机器学习方法不同，深度学习能够直接接纳原始数据作为输入，并通过多层非线性转换机制，自动提炼出更高层次的特征表示。这一转变不仅显著提升了特征的质量，还极大地减轻了人工操作的负担。此外，在多个应用领域内，深度学习技术已经展现出了超越传统机器学习方法的卓越性能。它凭借海量的数据与强大的计算能力，能够训练出极为复杂的模型，从而灵活应对各种复杂多变的实际应用场景。

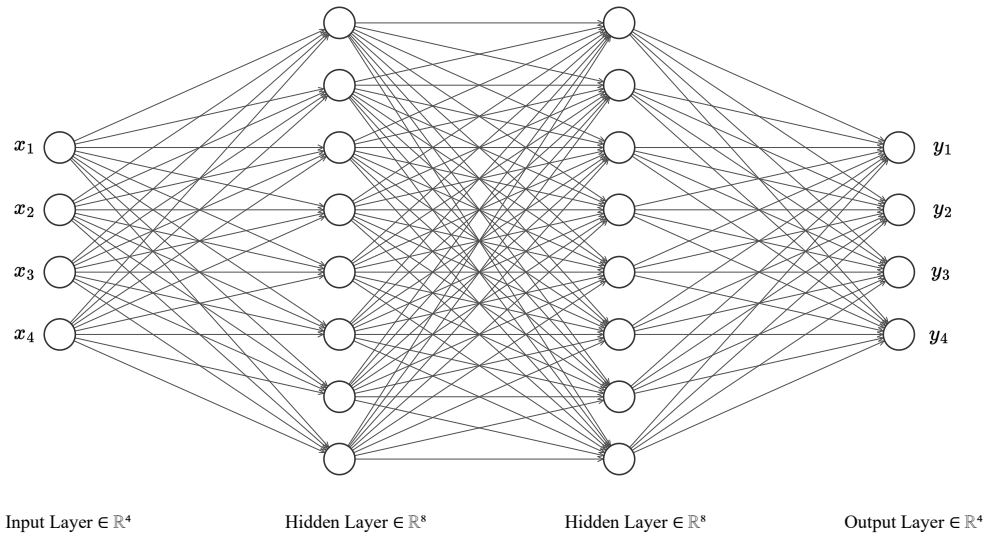


图 2.2 深度学习四层全连接神经网络

在图2.2中可以看到全连接神经网络中每层都包含一定数量的神经元，每个神经元代表一个值。除了输入层与输出层之外的每一个中间隐藏层都与上层下层神经元存在联系。详细的结构如图2.3所示，对上层的每个神经元做加权求和，在加上上层的偏差值，图中上层共有四个神经元与四个加权参数和一个偏差值 b, 最后经过非线性映射输出下一个神经元，非线性映射通过激活函数来实现，常用的有 ReLU 之类的，最后输出图中的下层神经元 y。

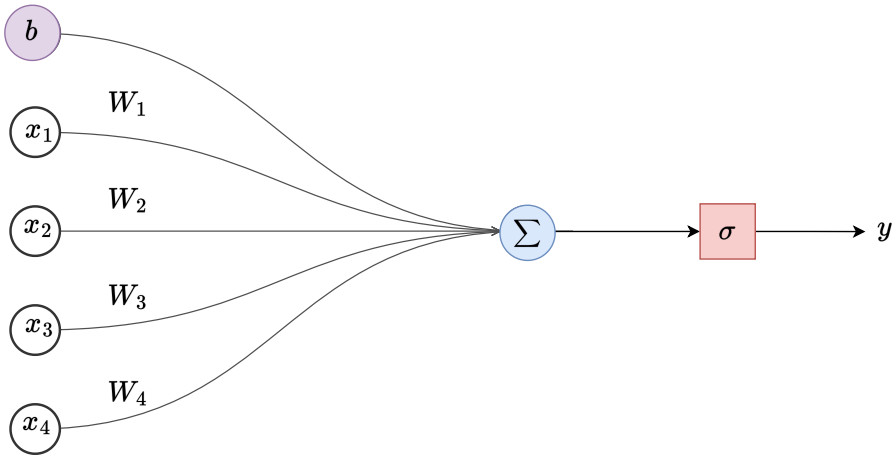


图 2.3 全连接神经元结构图

上述描述了深度学习神经网络的前向传播过程，前向传播对神经元需要学习的东西进行了预测。而在训练过程中主要使用反向传播算法[55]（BackPropagation，BP）进行训练，反向传播通过随机梯度下降的方法减小前向传播预测的标签与真是标签的误差，主要实现方式是不断修正神经网络中的权重与偏差来最小化优化目标。

2.2.2 卷积神经网络

卷积神经网络（Convolutional Neural Networks, CNNs）是一种专门用于处理数据有网格结构的深度学习模型，最早在图像识别和处理任务中取得了显著的效果。CNN 的核心是卷积层和池化层的堆叠，通过这些层提取图像的空间和位置特征。与传统的全连

接网络不同，CNN 使用局部连接和权重共享，使其在处理高维图像数据时具有较少的参数、更低的计算成本，同时保持了强大的特征提取能力。

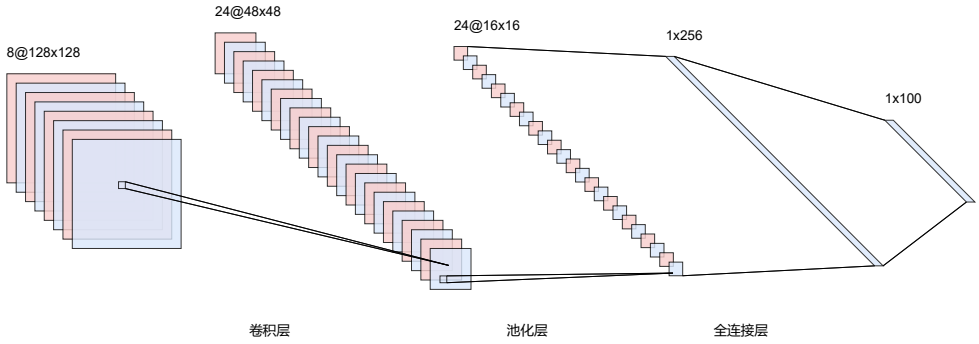


图 2.4 卷积神经网络示意图

如图2.4所示，卷积神经网络主要包括输入层、卷积层、池化层、激活层、全连接层以及输出层构成。输入的特征信息主要有卷积层和池化层主要用来提取特征信息，这些层在整个卷积神经网络中可以不断的有规律的堆叠，全连接层以及输出层根据任务的不同输出不同的信息，主要针对任务做出改变。下面将详细介绍卷积神经网络的各个主要模块：

（1）卷积层

卷积层（Convolutional Layer）是卷积神经网络的核心组件，其作用是提取输入数据的局部特征。卷积层的运作基于卷积运算，通过多个卷积核（filter）滑动窗口来扫描输入数据的不同区域，从而提取数据的局部信息，比如边缘、纹理和形状等特征。卷积层一般有卷积核、步幅（Stride）以及填充（Padding）等参数。卷积核，也称为滤波器，是一个小的矩阵（通常为 3x3、5x5 等），其中的参数是可以训练的权重。卷积核在输入数据上逐点滑动（即做卷积操作），通过对局部区域进行加权求和得到特征映射。不同的卷积核会提取不同的特征，比如边缘、颜色变化、纹理等。步幅决定了卷积核在输入数据上每次移动的步长。填充是指在输入数据的边缘添加额外的像素，使得卷积核在边缘位置时也能充分计算。积操作生成的输出结果称为特征图，它展示了卷积核在整个输入数据上的响应。特征图的大小由卷积核大小、步幅、填充方式共同决定。

卷积核的参数在整个输入上共享，显著减少了网络的参数量，提升了训练效率。而且卷积核只连接局部区域，减少了计算量，同时有助于模型专注于图像的局部特征。卷

积操作使得特征对平移有一定不变性，提升了模型对输入变化的鲁棒性。卷积层能够有效地提取数据中的空间特征，通过多个层级的卷积操作，网络可以逐步抽象出更高层次的特征。

（2）池化层

池化层（Pooling Layer）是卷积神经网络中的一个重要组件，其主要作用是对特征图进行下采样，从而减小特征图的尺寸和参数量，缓解过拟合，提高计算效率，同时保持主要特征信息。池化层通过聚合局部区域的特征，帮助模型在识别物体特征时获得平移不变性。池化层会减小输入特征图的空间维度（宽度和高度），从而减少后续层的计算和内存需求。例如，如果输入图像大小为 64×64，通过一次 2x2 池化操作可以将其尺寸缩减为 32×32，大大减少后续层的计算量。并且池化操作通过减少特征图的尺寸，减小了模型的自由度，从而降低了过拟合的风险。在特征图降维的同时，它还保留了数据的主要信息，确保模型仍能有效学习输入数据的主要特征。池化层还有助于增强模型对输入数据的平移不变性，即使输入图像发生小幅移动或变形，池化后的特征图依然能很好地描述这些信息。

池化层一般有三种池化方式，**最大池化（Max Pooling）**、**平均池化（Average Pooling）**以及**全局平均池化**[56]**（Global Average Pooling）**。最大池化是最常见的池化操作，它在池化窗口内取最大值，保留最显著的特征。最大池化常用于提取图像的边缘或亮度等重要信息，忽略细微的变化。平均池化是在池化窗口内取平均值，将窗口内的数值平均化。这种方法会更平滑地表示特征图的变化，通常用于减少数据的噪声。全局平均池化（Global Average Pooling, GAP）会在整个特征图上计算平均值，即每个通道只有一个数值。GAP 常用于分类任务中的最后一层，将每个通道的信息汇总为一个数值，用于全连接层或分类输出。

（3）全连接层

全连接层（Fully Connected Layer, FC Layer）是神经网络中的一种关键层，通常用于将特征向量转换为分类、回归或其他形式的输出。在全连接层中，每个输入节点和输出节点之间都有连接，因此称为“全连接”。全连接层通常位于网络的末端，它接收来自卷积层或池化层的高层次特征并进行整合，用于最终的决策输出。全连接网络的核心是矩阵运算，具体运算公式是2-6，在上一小章节详细的介绍了全连接神经网络的具体

细节。

$$Y = W \cdot X + b \tag{2-6}$$

（4）激活函数

激活函数[57]（Activation Function）是神经网络中的关键组件，它的作用是引入非线性，使得神经网络能够拟合复杂的非线性关系。若没有激活函数，神经网络只能表示线性映射，无法有效解决复杂的分类或回归问题。激活函数通常位于每一层神经元的输出之后，接受线性组合的输入并进行非线性变换，输出结果用于下一层的计算。常见的激活函数有 Sigmoid、Tanh、ReLU、Swish 等等，下面将详细介绍这些激活函数。

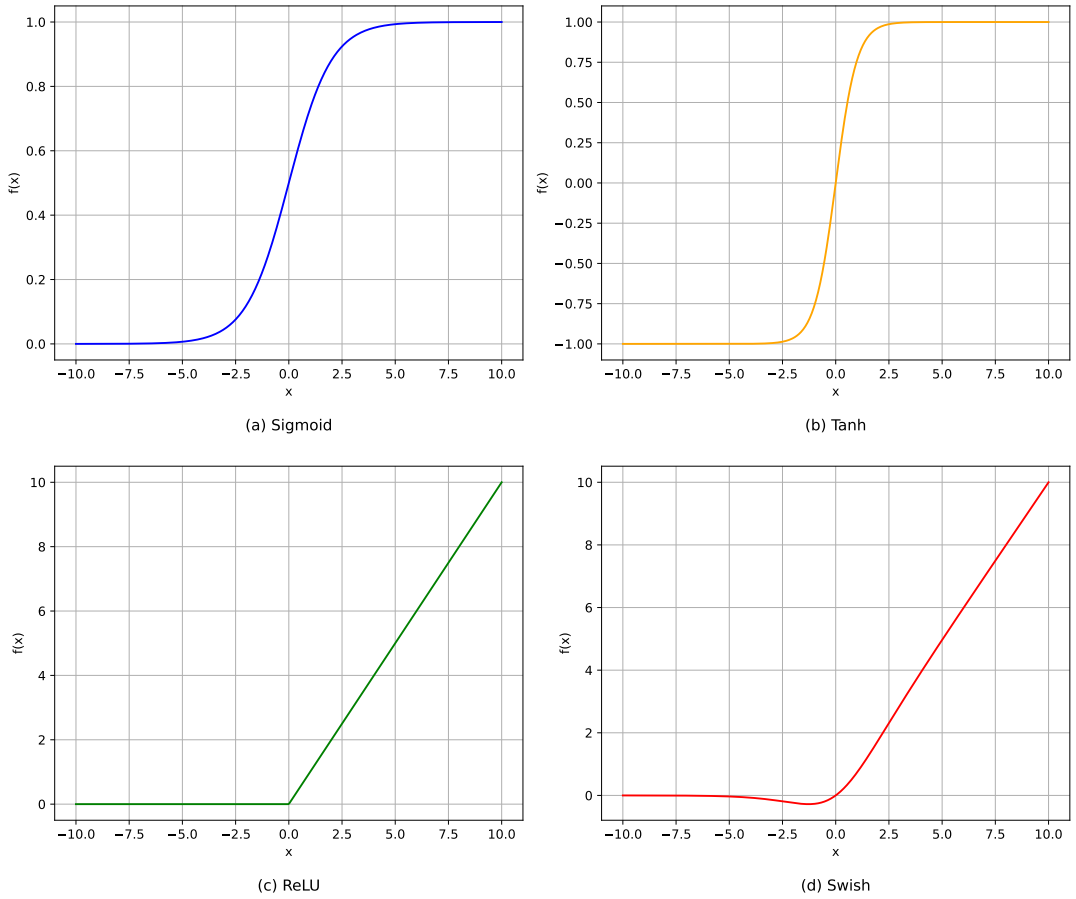


图 2.5 四种常用的激活函数

Sigmoid 函数将输入映射到 (0,1) 区间，适合概率输出（如二分类任务）。早期广泛应用，但较少用于深层网络，因为容易导致梯度消失问题，尤其是在深层网络中，导致学习过程缓慢。较为明显的是 Sigmoid 函数的梯度在接近 0 或 1 时趋近于 0，导致梯度消失。并且 Sigmoid 函数输出是非对称的，可能导致反向传播时梯度更新不稳定。其图

像可见图2.5(a) 所示，下面是 Sigmoid 函数的公式：

$$f(x)=\sigma(x)=\frac{1}{1+e^{-x}} \tag{2-7}$$

Tanh 函数将输入映射到 (−1,1) 区间，且输出是零中心的，有利于梯度更新。相较于 Sigmoid，Tanh 函数可以减少梯度消失问题，但在深层网络中仍然可能出现梯度衰减。对于较大的输入，Tanh 的梯度也趋近于 0，导致梯度消失。其图像可见图2.5(b) 所示，下面是 Tanh 函数的公式：

$$f(x)=\tanh(x)=\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{2-8}$$

ReLU 函数（Rectified Linear Unit）输出范围 $[0,+\infty]$，将负数部分直接设为 0，保持正数部分不变，具有稀疏激活的特点（即部分神经元被激活）。计算简单、高效，梯度在正区间不会衰减，因此常用于深层网络中，显著加速训练过程。但是当神经元的输入为负时，ReLU 的梯度为 0，神经元无法更新，可能导致一部分神经元“死亡”。其图像可见图2.5(c) 所示，下面是 ReLU 函数的公式：

$$f(x)=\max(x,0) \tag{2-9}$$

Swish 函数是由 Google 提出的，结合了 ReLU 和 Sigmoid 的特性，既包含非线性又不会像 ReLU 那样直接截断。Swish 的输出不再严格限制为正值，且对于小的负输入也有较小的梯度，适合深层网络，可以带来更好的训练效果。其图像可见图2.5(d) 所示，下面是 Swish 函数的公式：

$$f(x)=\sigma(x)\cdot x=\frac{x}{1+e^{-x}} \tag{2-10}$$

除此之外，maxout[58]、ELU[59]等激活函数也有着广泛的使用。

2.2.3 卷积神经网络模型

本章介绍本论文算法所使用的卷积神经网络模型，分别是简单的多层卷积神经网络与 ResNet 系列的 ResNet18 与 ResNet34。下面分别详细的介绍本文用到模型的架构。

（1）四层 CNN 模型

如图2.6所示，本文所用的 cnn 模型由 4 个块加上一个全连接层组成，每个块先经过一个卷积层对上层传来的特征图进行特征提取，随后对提取到的特征图做标准化处理，标准化处理之后经过 ReLU 激活函数，在块的最后一层是池化层。四层块中的卷积层是

不同尺寸的卷积核。图片从经过四块之后，最后经过全连接层，全连接层根据分类任务的类别数，改变全连接层最后的输出数，本文中用到十分类。

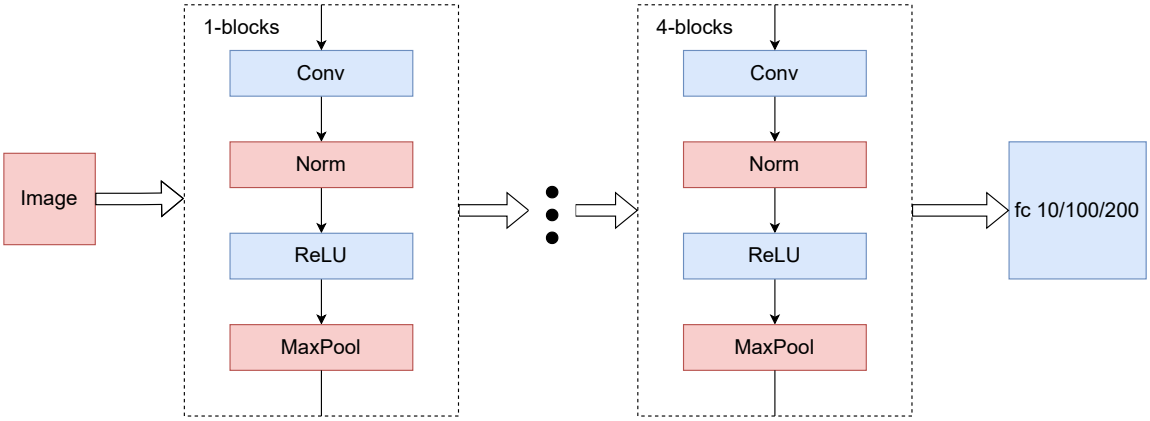


图 2.6 简单 CNN 模型架构

（2）ResNet 系列模型

ResNet（Residual Network）系列模型是由微软研究院提出的一种深度神经网络架构，主要应用于图像识别和分类任务。ResNet 的核心思想是引入残差连接（Residual

表 2.1 ResNet 系列模型详细信息

layer name	output size	18-layer	34-layer
conv1	112×112	7×7, 64, stride 2	
conv2_x	56×56	3×3 max pool, stride 2	
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 4$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 6$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 3$
	1×1	average pool, 1000-d fc, softmax	
FLOPs		1.8×10^9	3.6×10^9

Connection），即跳跃连接，用于解决深层神经网络中的梯度消失和模型退化问题。随着ResNet模型层数的增加，它依旧能保持较好的性能，甚至在更深的网络中取得了更高的准确率。下面是对ResNet系列模型的详细介绍。

在普通的深层神经网络中，随着层数增加，梯度可能会逐渐变小，从而使得模型很难学习深层特征。而ResNet通过残差连接的设计，使得信息能够直接从前层传递到后层，帮助梯度更容易地传递，避免了梯度消失问题。残差块（Residual Block）是ResNet的基本单元。其核心公式为：

$$\mathbf{y} = \mathrm{F}(\mathbf{x}) + \mathbf{x} \tag{2-11}$$

如图2.7所示，输入 $\mathbf{x}$ 在经过Conv1层个ReLU之后再经过Conv2层，在下一个ReLU层之前两者的输出相加和，之后经过最后一层输出到下一个块。在最后一块之后是一个全连接层，根据自己所需要分类的类别设置输出的数量，例如在CIFAR10上就设定为10。

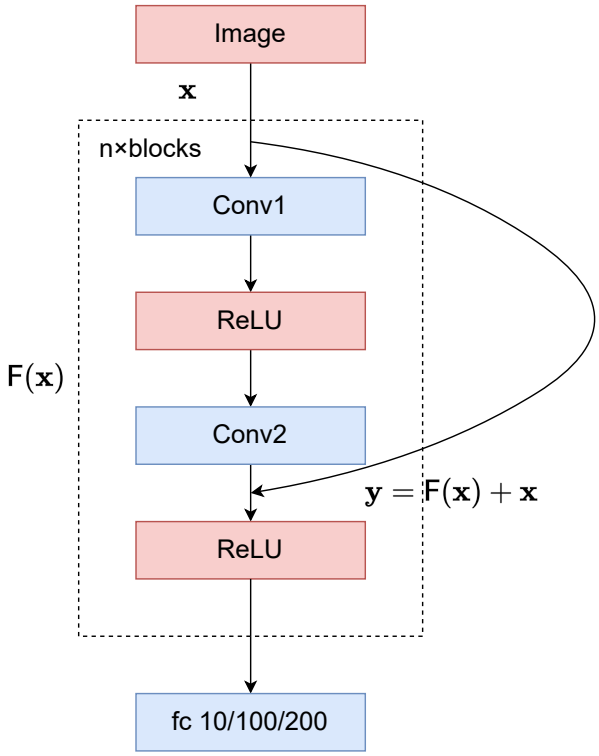


图 2.7 残差结构示意图

ResNet18和ResNet34的具体结构如表2.1所示，都是五层结构，除了第一个块的卷积核是 7×7 之外，其余的卷积核都是 3×3。ResNet18和ResNet34的不同之处就是每块中的相同的卷积层堆叠层数不同。

2.3 本章小结

本章首先介绍了联邦学习的经典框架以及联邦学习需要优化的目标并给出了联邦学习中经典算法 FedAvg 的伪代码实现。其次介绍了可以应用到联邦学习框架的深度学习模型，详细的介绍了计算机视觉中用到的卷积层、全连接层、激活函数等等需要用到的基础知识，其中着重介绍了本论文需要用的模型，分别是简单的四层卷积神经网络和 ResNet 系列模型的基础架构模型内部细节。

3　基于数据分布感知激活值的子模型抽取联邦学习

3.1　引言

3.1.1　神经元冲突现象

在章节1.2.3详细介绍了联邦学习在资源受限情况基于子模型抽取方向最新的研究，都在解决资源受限的问题上提出了完整的解决方案。这些解决方法的特点是基于提前预定好的抽取规则，这些方案的一个严重的缺点是会造成**神经元冲突**。图3.1作为例子具体介绍了冲突过程。图片中是一个拥有六个可训练参数的简单的深度学习神经网络，图

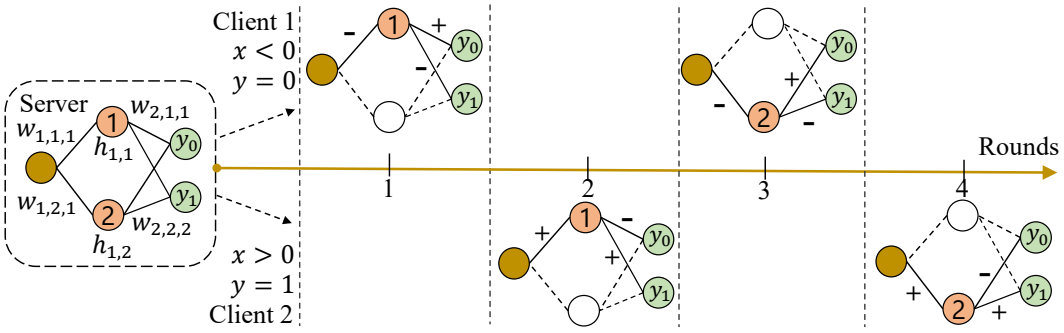


图 3.1 当前方法引起神经元冲突

中参数分别为 $W_{1,1,1}$、$W_{1,2,1}$、$W_{2,1,1}$、$W_{2,1,2}$、$W_{2,2,1}$ 以及 $W_{2,2,2}$ 六个参数，这个神经元要训练一个网络根据 x 的不同 y 值输出 0 或者 1。我们假定存在两个客户 Client 1 和 Client 2，其中 Client 1 拥有的数据是 $\{x<0,y=0\}$, 而 Client 2 中拥有的数据是 $\{x>0,y=1\}$。在第一轮中 Client 1 选择了编号为 1 的神经元训练，因为 $x<0$，要想使 $y=0$ 也就是输出 y_0 的概率变大，模型会朝着将 $W_{1,1,1}<0,W_{2,1,1}>0$ 的方向上靠近，这样保证 x 在分别与 $W_{1,1,1}<0,W_{2,1,1}>0$ 相乘之后输出的值是个很大的正数，也就是选择神经元 y_0 也就是 $y=0$ 的概率变大。而在保证 $W_{1,1,1}<0,W_{2,1,1}>0$ 的情况下，保证 $W_{2,1,2}<0$，这样 x 在分别与 $W_{1,1,1}<0,W_{2,1,2}<0$ 相乘之后输出的值是个很小的数，也就是选择神经元 y_1 也就是 $y=1$ 的概率变小。经过 Client 1 的训练之后，可以得到 $W_{1,1,1}<0,W_{2,1,1}>0,W_{2,1,2}<0$ 这样的结论, 如图3.1中的第一轮符号所示。在第二轮的训练中，Client 2 抽取了与 Client 1 一样的子模型，也就是选择了序号为 1 的神经元，但是由于 Client 2 的数据是 $\{x>0,y=1\}$，

正好与 Clinet 1 保持相反的数据分布，要想使得在 $x>0$ 的情况下，输出 y_1 也就是 $y=1$ 的概率增大，模型会朝着将 $W_{1,1,1}>0, W_{2,1,1}<0$ 的方向上靠近，这样保证 x 在分别与 $W_{1,1,1}>0, W_{2,1,1}<0$ 相乘之后输出的值是个很小的数，也就是选择神经元 y_1 也就是 $y=1$ 的概率变大。同样的，选择 $W_{1,1,1}>0, W_{2,1,2}>0$ 保证选择神经元 y_0 也就是 $y=0$ 的概率变小。这样在 Client 2 中得到了与 Client 1 相反的优化方向 $W_{1,1,1}>0, W_{2,1,1}<0, W_{2,1,2}>0$，如图3.1中的第二轮符号所示。按照联邦学习最后聚合的算法，将 Client 1 与 Client 2 的参数进行聚合，可以发现同一个神经元在两种不同的数据下出现了竞争的情况，不同边缘设备数据将神经元按照不同甚至相反的方向优化，这势必会造成训练效果的下降。

基于预先制定分配神经元方法所带来的神经元冲突问题，可以观察在联邦学习中不同边缘设备带来不同的数据对全局模型中激活神经元的分布情况，来自适应的给不同的客户端分配子模型。

3.1.2 分布现象观察

为了观察不同数据在深度学习模型中激活神经元的分布，设计了实验来观察不同激活值的现象。我们设计了一个三层的多层感知机在数据集 EMNIST 上进行实验，三层感知机神经元的数量分别是 50、24 以及 10，然后统计了在训练过程中每层神经元中的平均激活值。这次联邦学习实验设定了五个客户端，每个客户端分别包含两类别的数据。具体的对比如图3.2和图3.3所示，图中每行的 layer1、layer2 和 layer3 分别表示多层感知机的第一层、第二层和第三层，每个图标的横轴表示当前层的神经元的序号，例如在 layer1 是 $1-50$ 的序号，纵坐标表示当前神经元在多个训练批次过程中的激活值的平均值，每行图片比较了不同两个客户端之间的激活值之间分布的不同分布，图3.2和图3.3 记录了所有两个客户端对比的数据。通过观察不同层不同神经元激活值的分布可以观察看客户端之间存在的冲突。

观察图3.2中的第一行的 layer1 图，可以发现对于相同序列的神经元来说，Client 0 和 Client 1 激活程度十分不同，两者几乎大部分神经元的激活值都是不同的，体现在图中两条曲线基本不存在重合的地方，例如编号 18 的神经元在 Client 1 中激活值的大小接近与 1，而在 Client 0 中激活值的大小却接近与 0。如果 Client 0 和 Client 1 同时优化 layer1 层的多层感知机，势必在编号 18 的神经元出起冲突，Client 0 要求输出接近 1，才能更好的预测，而相反的是 Client 1 要求输出接近 0 才能更好的预测自己的数据，这种

情况在聚合的过程中必然对于 Client 0 和 Client 1 都是一种削弱的情况。

3.1.3 启发

受这一发现的启发，我们提出了一种简单但有效的方法——FedDSE，通过基于每个客户端的数据分布提取子模型来减少客户端之间的冲突。本方法的主要思想是让每个客户端根据其本地数据集上的激活情况自适应地从整个模型中提取神经元，其中选择激活幅度最大的神经元。通过这种方式，冲突可以最小化，因为每个客户端都被分配了最适合其数据分布的神经元，而不是那些被其他具有不同分布的客户端激活的神经元。在不同数据集和模型上的实验结果表明，在资源受限的约束下，相比基线方法可以显著提高训练效率。我们的贡献如下：

- 本文考虑了联邦学习（FL）中的数据异质性，并基于子模型提取进行了研究。我们的研究发现，具有不同数据分布的客户端往往会激活不同的神经元，如果神经元分配不当，会导致它们之间的冲突。
- 我们提出了一种新颖的训练方法 FedDSE，根据每个客户端的数据分布为其提取子模型。在 FedDSE 中，子模型的神经元是根据它们在每个客户端本地数据集上的激活水平选择的，从而能够为每个客户端分配最合适的神经元。
- 为了验证所提出方法的效率，FedDSE 在与最新方法比较中取得最优。

3.2 算法设计

3.2.1 符号设计

我们考虑一个具有 L 层的深度神经网络，第 l 层包含 m_l 个神经元，我们将模型的权重参数表示为 $\mathbf{w}$，将第 l 层的参数表示为 $\mathbf{w}_l = [w_l, b_l]$，其中 w_l 和 b_l 分别是权重和偏置。对于第 l 层的第 i 个神经元，我们计算其激活输出为 $h_{l,i} = \sigma(w_{l,i}\mathbf{h}_{l-1} + b_{l,i})$，其中 $\sigma(\cdot)$ 是非线性激活函数（例如 ReLU），$w_{l,i}$ 和 $b_{l,i}$ 表示该神经元的权重和偏置，$\mathbf{h}_{l-1}$ 表示前一层所有神经元的输出，即 $\mathbf{h}_{l-1} = [h_{l-1,1}, \ldots, h_{l-1,m_{l-1}}]$，在二维的卷积神经网络中，$h_l$ 代表一个二维的特征图。为简单起见，我们将网络的所有权重表示为 $\mathbf{w} = [\mathbf{w}_1, \ldots, \mathbf{w}_L]$。我们假设有 N 个客户端，每个客户端 n 只能访问其自己的私有数据集 $\mathbb{D}_n := \{x_i^n, y_i\}$，其中 x_i

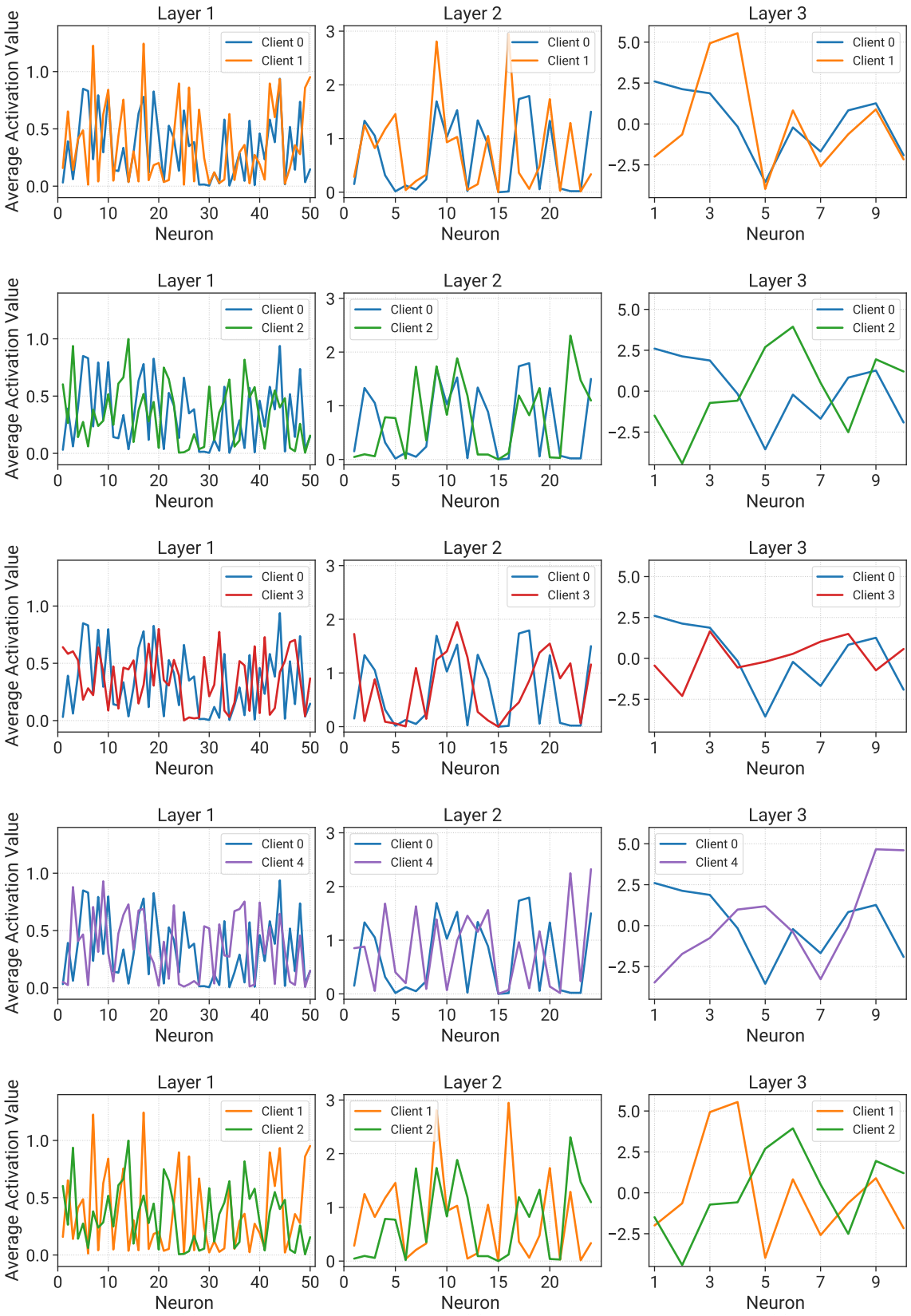


图 3.2 Client0、Client1、Client2、Client3 与 Client4 每层神经元激活值对比 1

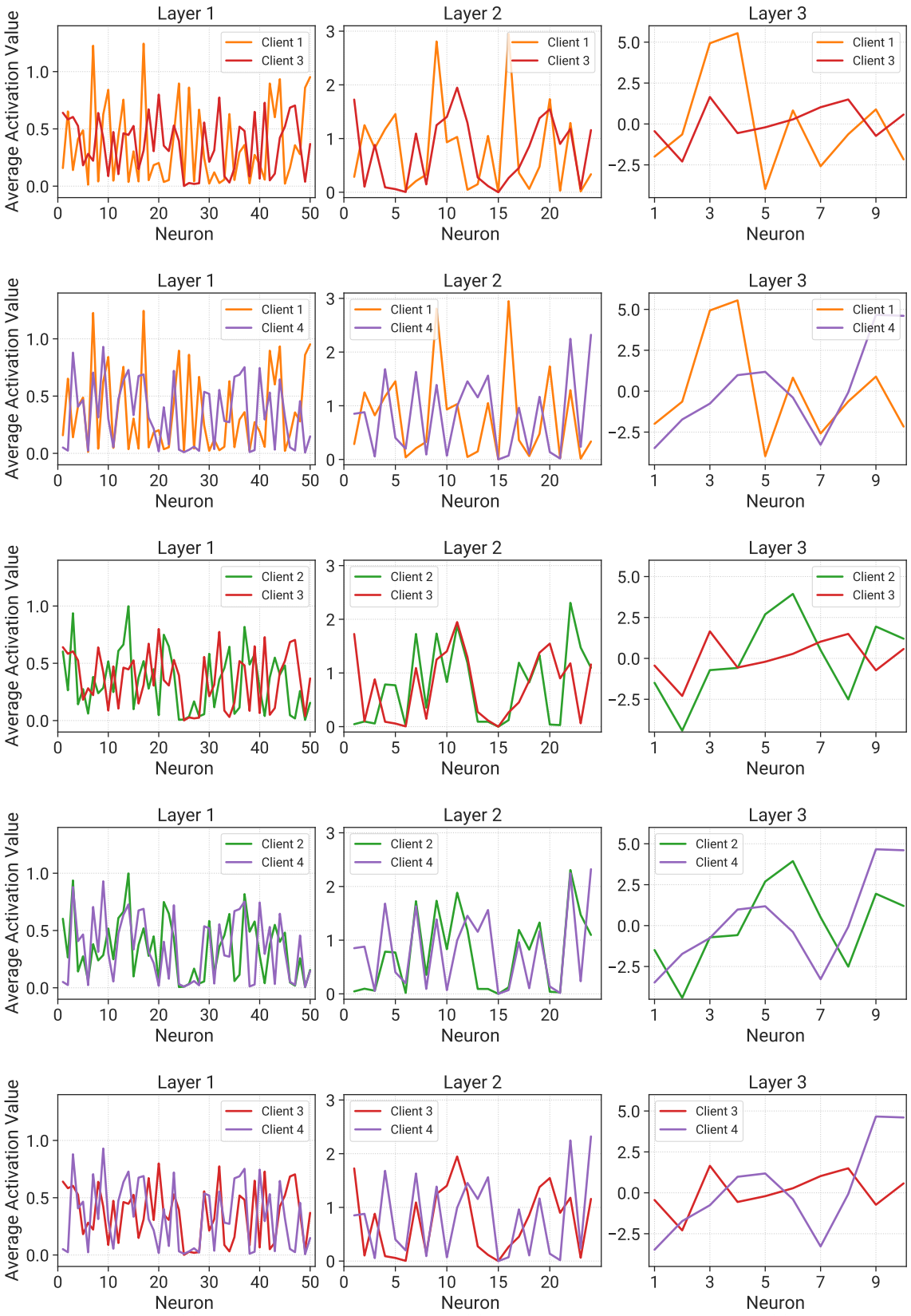


图 3.3 Client0、Client1、Client2、Client3 与 Client4 每层神经元激活值对比 2

表示第 i 个输入数据样本，$y_i \in \mathscr{C} = \{1,2,\cdots,C\}$ 表示 x_i 对应的标签。数据集 $\mathbb{D}_n$ 中的数据样本数量用 D_n 表示。$\mathbb{D} = \{\mathbb{D}_1,\mathbb{D}_2,\cdots,\mathbb{D}_N\}$，其中 $N = \sum_{n=1}^{N} D_n$。我们的目标是通过最小化整个数据集 $\mathbb{D}$ 上的损失来训练一个全局模型 $\mathbf{w}$、目标是优化公式2-4与公式2-5。

3.2.2　算法总体设计

受到上述发现的启发，我们提出根据每个客户端的数据分布提取一个子模型，详细的流程在算法3.2中展示。我们的方法 FedDSE 具有以下创新点。首先，考虑到充足的下载带宽，我们允许每个客户端 n 从服务器拉取整个模型 $\mathbf{w}$。其次，基于神经网络的基本特性，即推理消耗的内存远少于训练，每个客户端 n 仅通过在部分本地数据集上推理来选择神经元。最后，基于不同层的神经元激活幅度差异很大的观察结果，每个客户端以逐层的方式提取神经元，这不需要缓存前一层神经元的激活值。如算法3.2所示，首先在

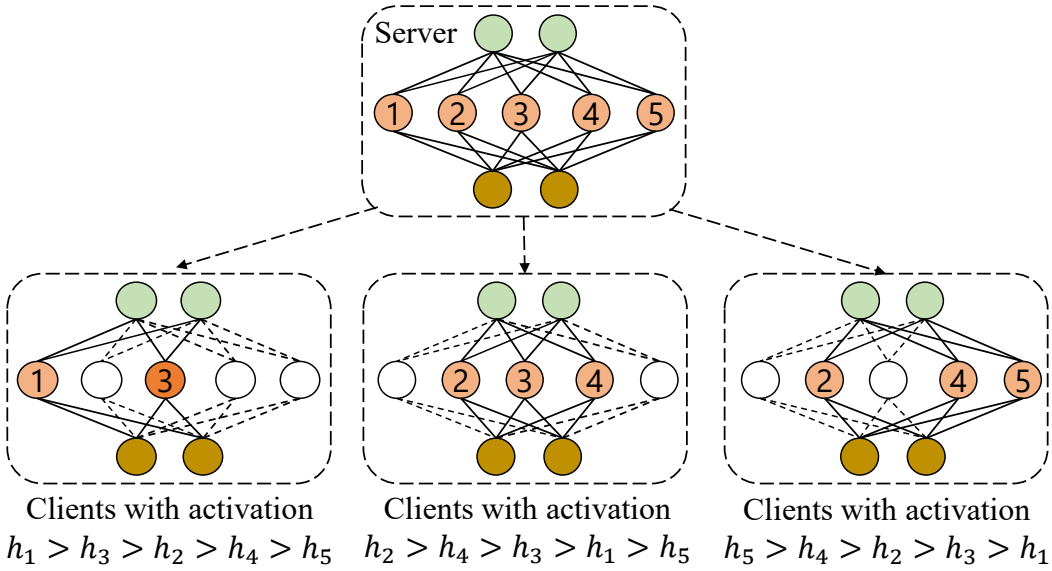


图 3.4 FedDSE 方法抽取子模型

服务器端初始化全局模型，然后再每一个轮次 t 中按照固定比例从所有的客户端中挑选出子集 $\mathscr{N}_t$ 参与本轮次训练。服务器向挑选到的客户端发送全局模型参数，然后客户端在自己的数据上训练模型。唯一不同的是，在传统的 FedAvg 中，客户端训练的是整个全局模型，而在资源受限场景下，客户端训练的是全局模型抽取后的子模型。

FedDSE 的核心算法部分就在于从全局模型抽取子模型的方法，具体来说，对于每一层 l，客户端 n 仅保留前 r 比例的神经元（以加权采样的方式），并裁剪其他神经元以

获得子模型，通过如下公式获得：

$$\mathbf{w}^n = \mathbf{w} \odot \mathbf{M}^n \tag{3-1}$$

其中 $\odot$ 表示逐元素乘法，$\mathbf{M}^n$ 是掩码。如果参数 $w_{l,i}$ 的神经元 $h_{l,i}$ 被裁剪，则 $\mathbf{M}^n_{l,i} = 0$，否则 $\mathbf{M}^n_{l,i} = 1$。每个神经元的采样概率基于其激活值确定。我们对每个神经元 i 的激活 h_i 应用 softmax 函数，得到其采样概率，公式如下：

$$p(i) = \frac{e^{h_i/T}}{\sum_{j=1}^{m} e^{h_j/T}} \tag{3-2}$$

其中 T 是温度。显然，激活值越大的神经元越有可能被采样。如图3.4所示，当服务器端将全局模型分发给客户端的时候，最左边的客户端选择了编号为 1 和 3 的神经元，而裁剪了编号为 2、4 以及 5 的神经元。本客户端这一层的激活值的排序情况是 $h_1 > h_3 > h_2 > h_4 > h_5$，根据公式3-2选择编号为 1 和 3 的概率大于其他的神经元，结果便如图所示。图3.4中的后两个子图也是如上述方法得到的。具体 $\mathbf{M}^n$ 的计算过程参考算法3.1。特别地，当温度 $T \to \infty$ 时，神经元以均匀方式选择，而当温度 $T \to 0$ 时，神经元以 Top-K 方式选择，即选择激活值最高的神经元 $\|h_{l,i}\|$。

当客户端抽取好本地的子模型之后，按照下面的公式在本地数据上训练更新模型：

$$\mathbf{w}^n = \mathbf{w}^n - \eta \nabla_{\mathbf{w}^n} f_n(\mathbf{w}^n) \tag{3-3}$$

其中 $f_n(\mathbf{w}^n)$ 表示数据小批量的损失，η 是学习率。然后，服务器接收所有客户端的子模型并聚合它们以更新全局模型：

$$\mathbf{w} = \sum\nolimits_{n \in \mathcal{N}_t} \mathbf{P}^n_t \odot \mathbf{w}^n \tag{3-4}$$

其中 $\mathcal{N}$ 表示选定的客户端集合，$\mathbf{p}^n$ 为每个子模型参数元素赋予权重。我们设置：

$$\mathbf{p}^n_{l,i} = \frac{1}{|\mathcal{N}_{l,i}|} \tag{3-5}$$

其中 $\mathcal{N}_{l,i}$ 表示参与选择参数 $w_{l,i}$ 训练客户端集合里面数据数量。最终经过 T 轮次的训练，在服务器中心侧聚合出一个完整的符合要求的全局模型。

3.3 实验过程与结果分析

为了验证我们的方法的高效性与准确性，我们选择了基于卷积神经网络的图片分类实验，这也是大多数联邦学习论文所用到的实验。

算法 3.1 GetMaskFedDSE

Input: 客户端 n 的计算能力系数 r，本地数据集 $\mathbb{D}=\{\mathbf{x},\mathbf{y}\}$, t 轮的全局模型 $\mathbf{w_t}$

Output: 客户端 n 在 t 轮的掩码 $\mathbf{M^{n,t}}$

procedure GETMASK(r_n, $\mathbb{D}_n^p$, $\mathbf{w_t}$)
  $\mathbf{h^n} = f(\mathbf{w_t};\mathbf{x},\mathbf{y})$
  for 对于每层激活值 $l \in \{1,\cdots,L\}$ **do**
    $\mathbf{S}_{sorted} \leftarrow \mathbf{sort}([h_{l,1},\ldots,h_{l,m_l}])$
    $\mathbf{S}_{top-r} = \mathbf{S}_{sorted}[1 : r \cdot m_l]$
    if $\mathbf{h_{l,i}} \in \mathbf{S}_{top-r}$ **then** $\mathbf{M_{l,i}^{n,t}} = 1$
    else $\mathbf{M_{l,i}^{n,t}} = 0$
    end if
  end for
  return $\mathbf{M^{n,t}}$
end procedure

3.3.1 数据集与模型选择

我们在三个模型和四个主流数据集上评估了所提出的 FedDSE 的性能，包括用于 EMNIST[60]的简单的卷积神经网络，用于 CIFAR-10 和 CIFAR-100[61]的 ResNet18[62]，以及用于 TinyImageNet[63]的 ResNet34。与 HeteroFL[64]类似，我们采用了静态批量归一化，并在每个卷积层之后使用了一个标量模块。简单的卷积神经网络的通道数分别为 $\{64,128,256,512\}$, 具体的结构如图2.4所示。而模型 ResNet18 与 ResNet34 则与表2.1中的架构一致。数据集方面我们使用了四个数据集，分别是 EMNIST、CIFAR-10、CIFAR-100 以及 TinyImageNet，具体信息如表3.1所示，从上到下数据量增加分类数量逐渐增加，数据集难度有所提升。

3.3.2 实验设置

数据异质性 为了模拟现实情况中的数据异质性的特征，我们遵循 HeteroFL 中的设定方法[31,64]，给不同的客户分配固定的数据种类，就是不同的客户中仅仅包含我们设定分配种类的数据，而不会包含其他种类的数据，以此来模拟现实中的中的非独立同分布

算法 3.2 FedDSE

Input: 全局模型参数 $\mathbf{w}$, 学习率 η, 总通讯次数 T, 本地客户端训练次数 E

Output: T 轮的全局模型参数 $\mathbf{w}_{T+1}$

初始化全局模型参数 $\mathbf{w}_1$
procedure SERVER-SIDE OPTIMIZATION
  for 对每个轮次 $t \in \{1, \cdots, T\}$ **do**
    随机挑选所有客户端的子集 $\mathcal{N}_t$
    for 对于每个挑选到的客户端 n **并行执行 do**
      $\mathbf{w}_{t+1}^n \leftarrow$ **ClientLocalUpdata**$(\mathbf{w}_t)$
    end for
    更新全局模型参数 $\mathbf{w}_{t+1} = \sum_{n \in \mathcal{N}_t} \mathbf{P}_t^n \odot \mathbf{w}_t^n$
  end for
end procedure
procedure CLIENTLOCALUPDATE($\mathbf{w}_t$)
  从服务器侧接收 $\mathbf{w}_t$
  $\mathbf{M}_t^n \leftarrow$ **GetMask**$(\mathbf{w}_t)$
  初始化 $\mathbf{w}_{\mathbf{t,0}}^{\mathbf{n}} = \mathbf{w}_t \odot \mathbf{M}_t^n$
  for 从 1 到 E 迭代轮次 e **do**
    更新本地模型 $\mathbf{w}_{\mathbf{t,e+1}}^{\mathbf{n}} = \mathbf{w}_{\mathbf{t,e}}^{\mathbf{n}} - \eta \nabla_{\mathbf{w}_{\mathbf{t,e}}^{\mathbf{n}}} f_n(\mathbf{w}_{\mathbf{t,e}}^{\mathbf{n}})$
  end for
  return $\mathbf{w}_{\mathbf{t,E+1}}^{\mathbf{n}}$
end procedure

表 3.1 不同数据集信息

数据集名称	训练集（条）	测试集（条）	类别（类）
EMNIST	60000	10000	10
CIFAR-10	50000	10000	10
CIFAR-100	50000	10000	100
TinyImageNet	100000	10000	200

（non-IID）特性。根据每位客户分配种类的数量，我们将设定两种情况，分别是**高数据异质性**和**低数据异质性**，顾名思义，高数据异质性客户端分配到的数据种类更少，也就意味着数据更具体本地特色，与其他客户端数据分布差距更大；那么，低数据异质性设定中每个客户分配更多种类的数据，也就是不同客户之间重复种类的数量增多，数据的泛化性更好，与其他客户端数据分布差别相较高数据一致性小。我们根据不同的数据集来设定不同数据异质性所需要的种类数，如表3.2所示，分别表示列出了四个数据集高低数据异质性的情况，例如在 EMNIST 数据集，高数据异质性给客户端分配的种类数是 2 种，而低数据异质性分配则是 4 种。

表 3.2 不同数据集信息

数据集	EMNIST	CIFAR-10	CIFAR-100	TinyImageNet
高数据异质性	2	2	8	16
低数据异质性	4	4	16	32

模型异质性 在模拟现实环境中的边缘设备的资源分布场景中，FedDSE 主要应用于低质量客户端分布场景，指的是在所有的客户计算资源分布中不存在达到中心侧服务器水平的客户端，意味着中心侧模型不会被任何一个客户端完整训练。因此我们设定了五种不同客户端计算资源分别是 $\mathbf{r}=\{0.99,0.5,0.25,0.125,0.0625\}$，为了模拟低质量客户端分布[31,64]，本章实验我们设定最高的比例为 0.99。其中 r 中的 0.5 表示客户端资源能力有服务器端的 $\frac{1}{2}$，服务器端可以训练整个全局模型的所有神经元，而 $r=\frac{1}{2}$ 的客户端可以训练全局模型一半的神经元。同时，我们设定上述中的五种设定都是均匀分布的，也就是说每种分布都占客户端总数的百分之二十。

baseline 方法 我们比较了近期的四种比较主要的方法，分别是 HeteroFL、FedRolex、Federated Dropout 以及 DepthFL 作为 FedDSE 的对比方法。为了保证比较的公平性，我们对所有方法使用了相同的学习率、本地训练轮数以及通信轮数。每种数据集对应的具体训练参数见表3.3。其中简单数据集 EMNIST 的训练轮次是 800，其余的数据集的训练轮次都是 2500。数据中的推理数据数量是指每次抽取子模型时，使用本地数据推理得到激活值过程中使用的数据数量，本实验中用到了本地全部数据进行推理。本地训练次数是指在客户端训练子模型的次数。所有的训练过程使用的都是随机梯度下降（Stochastic Gradient Descent, SGD）优化器。

表 3.3 不同数据集训练参数设置

	EMNIST	CIFAR-10	CIFAR-100	TinyImageNet
本地训练次数 (E)	2	2	2	2
学习率	0.001	0.001	0.001	0.001
训练轮次	800	2500	2500	2500
优化器	SGD	SGD	SGD	SGD
动量	0.9	0.9	0.9	0.9
权重衰减	5.00E − 04	5.00E − 04	5.00E − 04	5.00E − 04
推理数据数量	全部	全部	全部	全部

训练配置以及平台 对于 EMNIST、CIFAR-10、CIFAR-100 以及 TinyImageNet，我们应用了边界框裁剪来增强图像[65]。在每个通讯轮次中从设定的 100 个客户端中选出百分之十进行训练，$frc = 10\%$。选定的客户的计算能力符合均匀分布。实验在实验在 PyTorch 框架[66]上进行，本章实验设备是 Nvidia 3090。

评估指标 对于图像分类任务，我们采用全局准确率作为评估指标，定义为服务器模型在整个测试集上的准确率。

3.3.3 训练细节

掩码交叉熵损失 已经证明，联邦学习 (FL) 的失败可以归因于在多次迭代中局部训练的局部模型参数之间的权重差异。这种权重差异主要体现在神经网络的最终分类层。因此，与使用包含所有类的完全交叉熵损失不同，每个局部模型的训练只与它们各自的类有关。允许每个局部模型根据可用的局部标签信息专注于特定的子任务。为了实现这一点，引入了掩码交叉熵损失，它涉及在应用损失函数之前将模型的输出掩蔽掉。

放缩模块 由于通过多个轮次优化客户端模型，因此在不同计算复杂度水平下的局部模型的参数可能会出现偏差，并在尺度上发生变化。这种现象被称为偏移[67]。在推断阶段，为了直接利用全模型，采用了偏移率为 q。该技术在训练阶段对输入值进行 $\frac{1}{1-q}$ 的缩放。本文引入放缩模块, 插在参数层之后, 在激活层之前。该缩放模块在训练阶段以的 $\frac{1}{1-q}$ 因子对输入值进行缩放。在全局聚合之后，保证全局模型可以直接使用。

静态批归一化 批归一化[68-69](Batch Normalization, BN) 在深度学习模型中被广泛

使用。然而，许多主流算法避开了 BN。与 BN 相关的一个重要问题是需要对每个隐藏层的表示进行连续估计。将这些统计数据传递给服务器的过程会导致通信成本增加，并引发隐私担忧。静态批归一化 (static Batch Normalization, sBN) 被应用于优化隐私受限的异构模型。在训练阶段，sBN 只是简单地对批量数据进行归一化处理，并没有对运行估计值进行跟踪。因为每个通信轮都是独立的，sBN 适用，在完成训练过程后，服务器依次查询本地客户端。

3.3.4 实验结果

表3.4和表3.5分别展示了我们的方法与上述四个基线方法在高数据异质和低数据异质性情况下的对比结果。FedDSE 的温度设置为 0。五个方法全部从 EMNIST 到 Tiny-Imagenet 准确率都是逐渐下降的，而且在 EMNIST 数据集上所有方法准确率可以达到百分之九十往上，而在数据集 TinyImagenet 上仅仅只能达到百分之十五左右。由此我们将 EMNIST 分为简单数据集，将 CIFAR-10 设定为中等数据集，而将 CIFAR-100 和 TinyImagenet 认定为困难数据集。

表 3.4 低质量客户端分布场景中在高数据异质性下不同方法准确率对比

方法	EMNIST	CIFAR-10	CIFAR-100	TinyImageNet
HeteroFL	93.21	38.13	8.00	5.72
Federated Dropout	87.96	50.16	10.47	10.17
FedRolex	91.41	55.61	14.02	15.39
DepthFL	92.34	49.42	11.22	12.45
FedDSE	**95.34**	**58.19**	**16.61**	**22.19**

表3.4也就是在高数据异质情况下，FedDSE 在所有的数据集中取得了最好的准确率。相对于其余四个方法中最好的分别提高了 2.13%、2.58%、2.59% 和 6.80%。对简单数据集和中等数据集的提升大多数在百分之二左右，而对于困难数据集 TinyImagenet 的提升有百分之六点八之多。可以看出，FedDSE 相较于其他四个方法都是准确率更高，特别是在困难数据集上的提升更为显著。这表明，当类别数量相对较少时，我们的方法能够准确捕捉并激活相关神经元进行训练，使得 FedDSE 方法显著好于其他方法。整体来看，Federated Dropout 于其他方法相比表现的最差，这也与方法中随机选择神经元有很

大的关系；HeteroFL 在简单数据集上表现稳定，但是对于中等和困难数据集准确率出现了灾难性下滑，由于其在中等与困难数据集上有神经元未参与训练造成的；FedRolex 是除了我们方法之外最优的方法，对于所有等级数据集表现稳定且准确率较高；DepthFL 表现稳定但效果一般。

表 3.5 低质量客户端分布场景中低数据异质性下不同方法准确率对比

方法	EMNIST	CIFAR-10	CIFAR-100	TinyImageNet
HeteroFL	97.61	47.01	11.16	10.52
Federated Dropout	97.63	58.16	16.21	19.18
FedRolex	98.61	68.04	20.81	19.44
DepthFL	97.75	55.93	17.87	20.99
FedDSE	**98.65**	**70.82**	**22.93**	**22.88**

表3.5是在低数据异质情况下五个方法的对比情况。FedDSE 是所有方法中准确率最高的，相对于其余四个方法中最好的分别提高了 0.04%、2.78%、2.12% 和 1.87%。在低数据异质这种简单的情况下，在 EMNIST 上所有方法的差距缩小，而在中等和困难数据集上不同方法的差距也仅有百分之二左右。FedRolex 方法是除了我们方法之外表现最好的，HeteroFL 方法在简单场景下表现不佳，与存在未参与训练神经元有关。总体来说在简单场景下，所有所有方法的差距在缩小，因为低数据异质情况下各个客户端数据分布更加相似导致联邦学习的训练难度下降。

图3.5描述了五个方法在不同数据集上的对应高数据异质性与低数据异质性的准确率。非常直观的我们可以观察到高数据异质性的准确率远远低于低数据异质性的准确率，这是因为在低数据异质性条件下，客户端数据集均匀分布，模型收敛更快，显著减少了训练开销。其中，Federated Dropout 方法高低数据异质性在不同数据集中对准确率的影响较大，而 HeteroFL、FedRolex 与 DepthFL 则在所有数据集中高低数据集影响较为稳定。FedDSE 在最高难度的数据集上 TinyImagenet 上，高低数据集对准确率的印象较小，两者差距很接近，这意味着我们的方法在困难数据集和高数据异质情况下会取得较好的效果，达到了与低数据异质性相同的准确率。

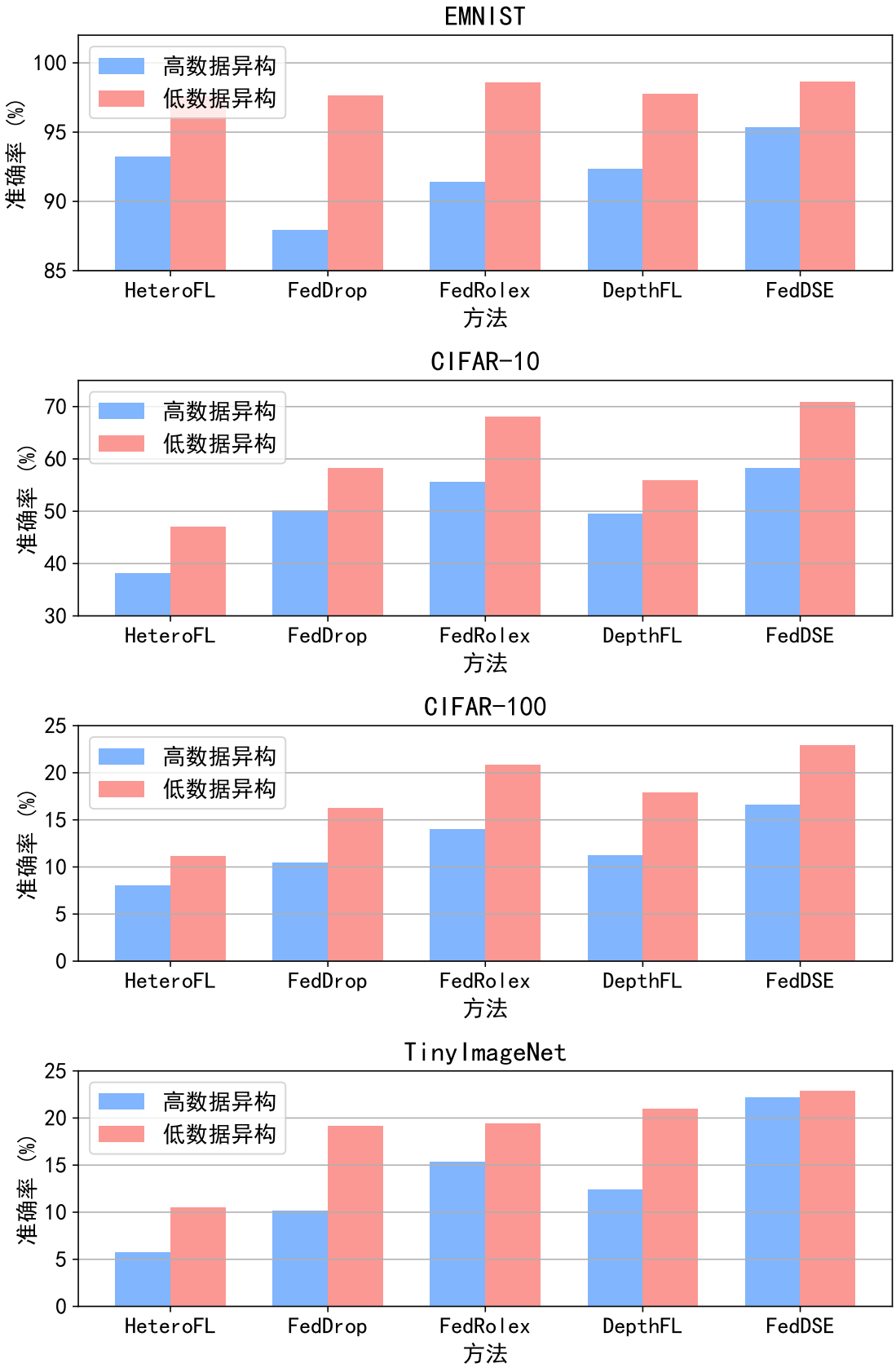


图 3.5 高低数据异质性对比图

3.4　消融实验

3.4.1　客户端计算能力分布对模型能力的影响

在上述实验中，客户端容量的分布是均匀设置的，就是每个计算能力客户端的占比都是百分之二十。为了观察不同的客户端占比对最终结果的影响，我们选择了两种客户端计算能力分别是 $\beta=\{\frac{1}{2},\frac{1}{16}\}$。然后我们定义 ρ 为计算能力为 $\frac{1}{2}$ 的客户端占总共客户端的比例，例如 $\rho=0.8$ 表示计算能力为 $\frac{1}{2}$ 的客户端占到所有客户端的比例是 0.8，计算能力为 $\frac{1}{16}$ 的占客户端总数的 0.2。最后 $\frac{1}{2}$ 计算能力的用户占比从零到一观察对准确率的影响，我们在简单数据集 EMNIST 和中等数据集 CIFAR10 上进行了实验。

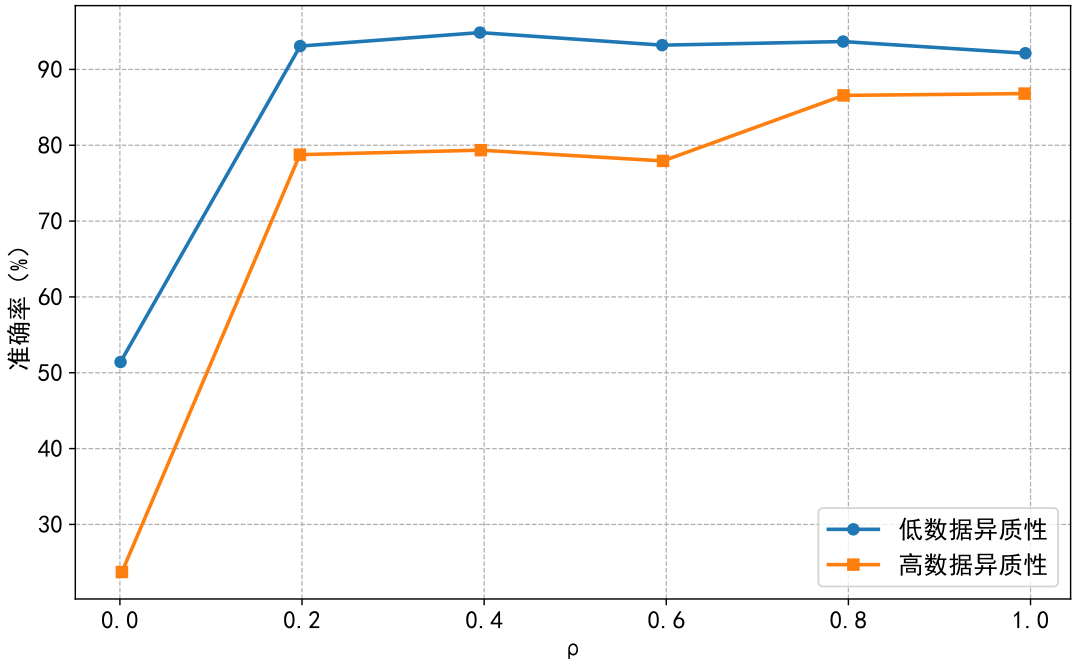


图 3.6 EMNIST 上 ρ 对准确率影响曲线图

在 EMNIST 上的结果如图3.6所示，我们可以观察到，首先高数据异质性的准确率均比低数据异质性差，这与先前所进行的实验吻合，低数据异质性的数据分布更加均匀，更容易训练出泛化性好的模型。其次，随着高计算能力客户端占比的提高实验结果准确率一般也随之提高，这与整体参与训练的客户端的训练参数增加有关，训练参数的增加意味着更多的信息可以在训练的过程中习的，模型的能力也会随之提高。但是在简单数据 EMNIST 上，在 $\rho=0.2$ 时候模型的准确率基本上已经达到了顶峰。这是因为在简单数据集上当训练的参数量到达一定数量时，也就是 $\rho=0.2$ 时这些参数已经可以训

练出够用的模型，这个时候增加参数量已经不是提高准确率的瓶颈。在低数据异质性上当 $\rho = 0.2$ 之后，模型的准确率微微上升之后，之后出现轻微的下降。而在高数据异质性的情况下，在 $\rho = 0.2$ 之后，增加参数量准确率还有提升，这与训练难度增加有一定的关系。

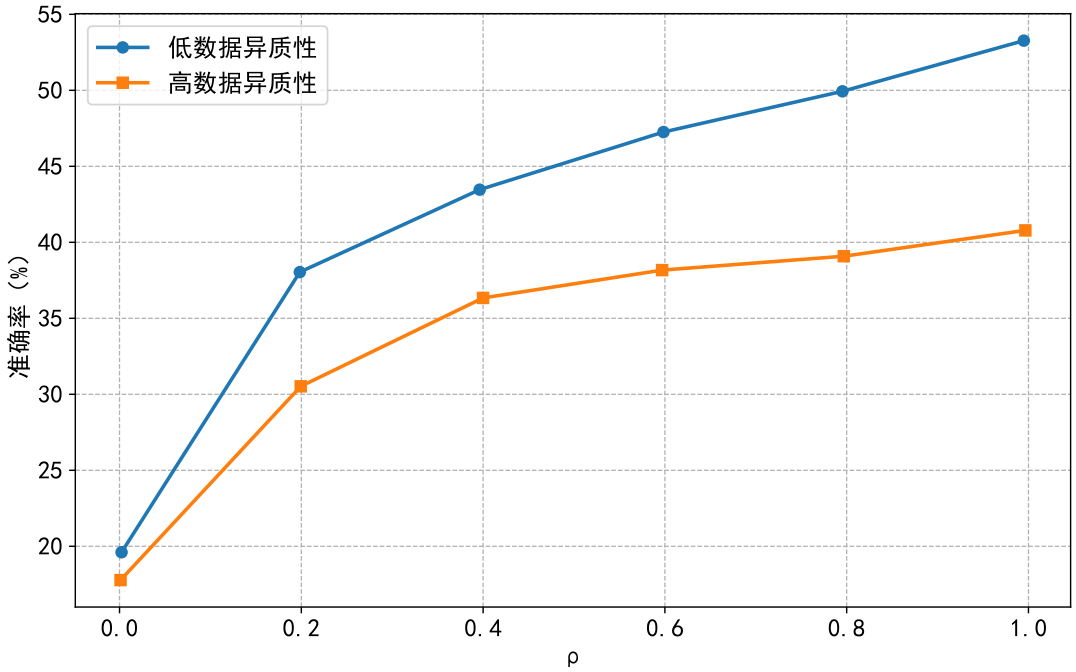


图 3.7 CIFAR10 上 ρ 对准确率影响曲线图

表3.7展示了在中等数据集 CIFAR10 上，分别在高低数据异质性下 ρ 对模型准确率的影响。在中等数据集 CIFAR10 上随着参数的增加，模型的准确率随之增加。低数据异质性准确率远高于高数据异质性场景，与 EMNIST 的基本情况相似。但是与之不同的是，随着参数的增加中等数据集的准确率是一直增长的，没有 EMNIST 上 $\rho = 0.2$ 的瓶颈，在 $\rho > 0.2$ 之后，模型的准确率也是逐步上升的，增大模型的参数准确率还有上升的趋势。这说明在中等数据集上，无论是高数据异质性还是低数据异质性场景参数数量是影响模型准确率的瓶颈，通过提高训练模型的参数量就可以提高模型的准确率。总结来说，图3.6和图3.7 证明了在资源受限联邦学习的训练中特别是在任务比较复杂情况下，训练的参数不够是影响模型准确率的重要因素，可以增加训练模型的训练参数来提升模型能力。

3.4.2 统计异质性的影响

在上述实验中，我们定义了高低数据异质性。也就是通过设定每位客户本地数据集中包含的训练数据的种类数来确定数据的异质程度，也就是客户端数据分布的均匀程度，种类数越少意味着数据分布单一，均匀程度较差，训练难度大；相反种类数量越大意味这数据分布均匀，模型泛化性好。例如在 EMNIST 中高数据异质性的种类数是二，低数据异质性的种类数是四，如表3.2所示。本小节我们讨论客户端的数据异质性程度对于模型训练最终准确率的影响。我们采用 EMNIST 数据集在简单的卷积神经网络上进行实验，设定客户端的包含种类数分别为 $\{2,4,6,8,10\}$。

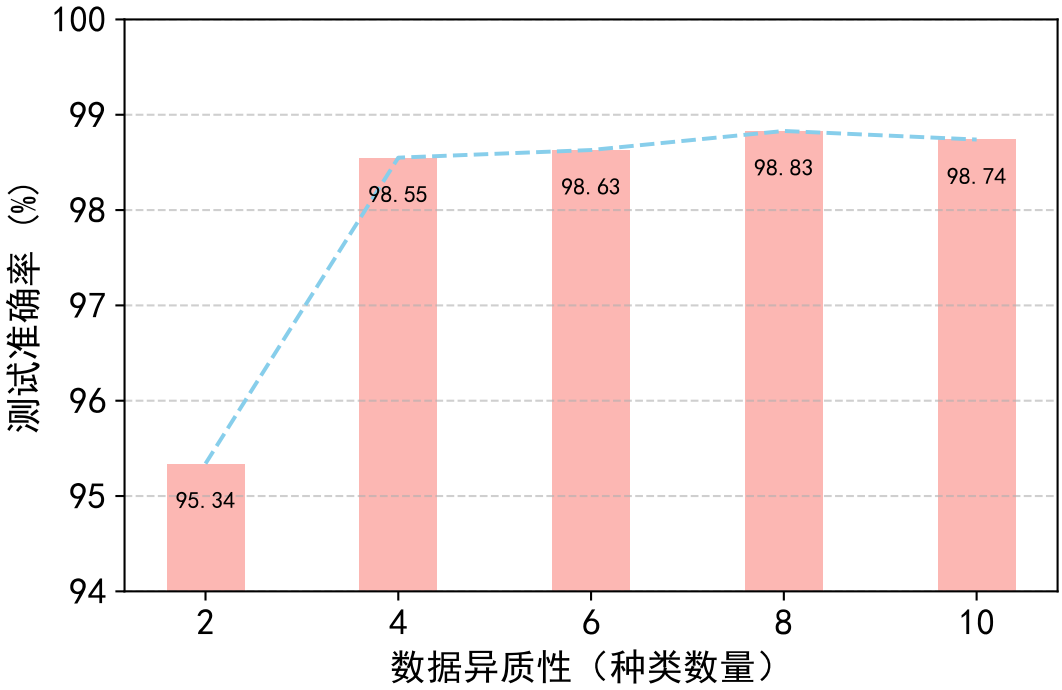


图 3.8 数据异质性对准确率的影响

与参数多少对模型最终准确率的影响类似，在简单数据集 EMNIST 上模型准确率对于数据分布均匀程度也存在一定的阈值。如图3.8中所示，在种类数为 2 的时候，训练的模型的准确率为 95.34%, 当种类数增加到 4 的时候，模型的准确率提升到 98.55%, 提升了 3.21%，而当种类继续提高的时候，也就是数据更加均匀时，模型的准确率依然在种类为 4 的结果附近徘徊，没有得到大的提升。这说明在简单任务上当数据均匀性增加到一定程度的时候，模型的准确率增长会出现瓶颈。在大多数实际情况中，十分类问题中客户用到的也仅仅是四类，这也符合实际情况，也就是说我们的方法可以在实际中得

到很好的应用。

3.4.3 客户挑选比例对准确率的影响

为了观察每轮中中心服务器挑选参与训练客户端比例对模型准确率的影响，我们在

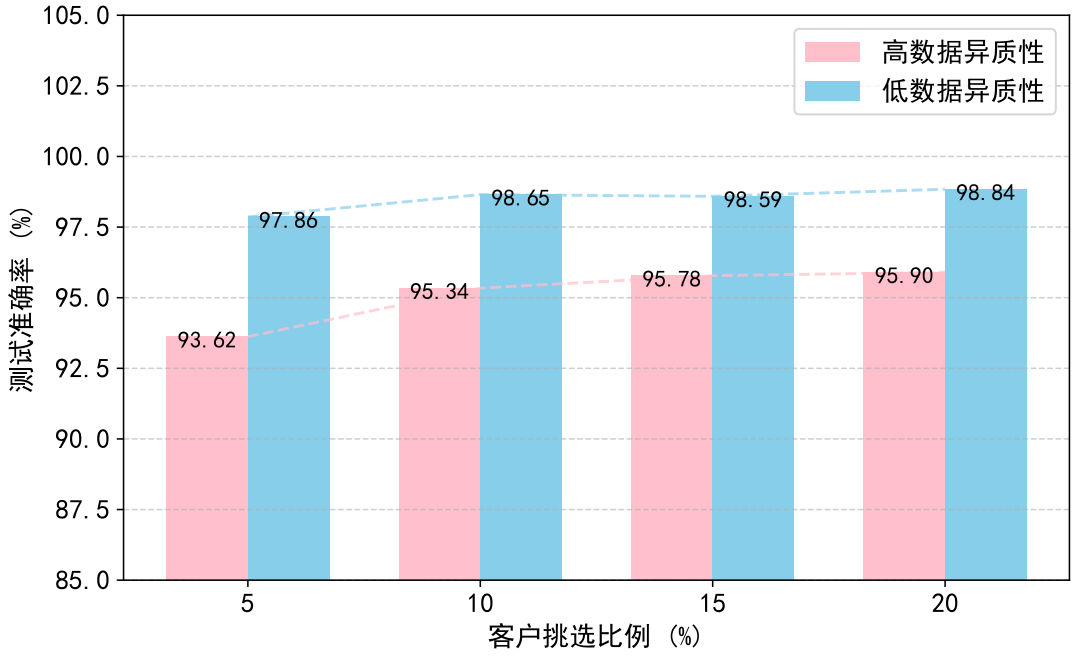


图 3.9 数据异质性对准确率的影响

简答数据集 EMNIST 上将挑选比例 frc 设定为 $\{5,10,15,20\}(\%)$，分别在高数据异质和低数据异质情况下比较 frc 对模型准确率的影响。

如图3.9所示高数据异质性情况下当减小 frc 到 5% 的时准确率仅有 93.62%，而将 frc 提高到 10% 时准确率提高到 95.34%，此后增加每个轮次训练的客户的数量对模型准确率的增长效果不佳，也就是说参加训练的客户端的数量也有一个阈值，达到之后边际效应收紧。同理这样的情况出现在低数据异质情况下，由于低数据异质且是在简单数据集上进行实验，frc 的增加对模型准确率的影响没有在高数据异质情况下明显。也就是说在 frc $=5\%$ 的时候已经达到了低数据异质情况下的阈值。通过上述分析我们观察到可以选择一个合适的 frc 的数值使得既不增加模型训练的资源消耗，又能训练出与更多客户参与训练模型一样的效果。

3.4.4　用于推理数据数量对准确率的影响

在 FedDSE 中我们在抽取子模型过程中，首先要对全局模型进行推理，然后再推理中通过激活值来选取每层抽取的神经元，用来推理数据的大小对模型准确率也有很大的影响，如算法3.1中第二行中用来获取 **h** 中 $\{x,y\}$ 的数量。我们设定推理数量按照 $\{64,128,256,\text{all}\}$，其中 all 表示整个完整本地数据集的数量。同样在 EMNIST 数据集上，在高低数据异质性环境下进行实验。

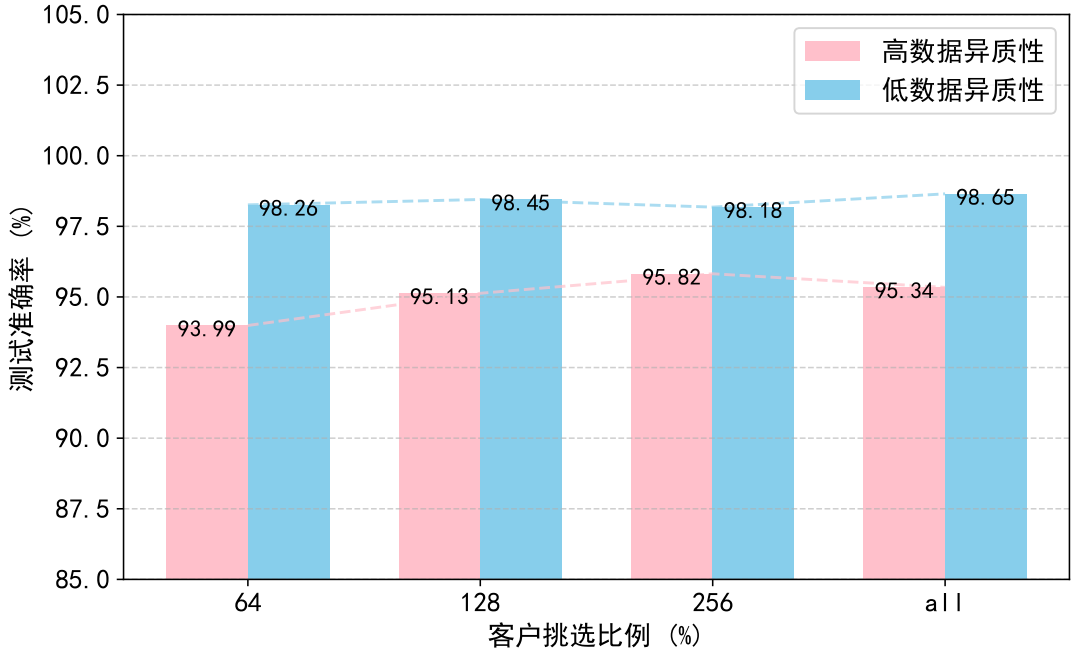


图 3.10 用于推理数据数量对准确率的影响

具体结果如图3.10所示，在高数据集异质的情况下，当推理数据的数量减少到 64 的时候，模型的准确率下降到了 93.99%，而当推理数据数量上升到 128 的时候，模型的准确率上升到了 95.13%。此后继续增加推理数量对模型的准确率影响较为不显著，也就意味着使用 128 条数据对模型进行神经元抽取已经可以很好的选择出活跃的神经元，与增加推理数据情况下选择的神经元没有较大的差别。然而这个现象在低数据异质性情况下不明显，说明低数据异质情况下，使用 64 条数据已经满足挑选活跃神经元需求，所以模型的准确率没有发生大幅提升与下降。

3.4.5 温度对模型准确率的影响

在实际应用中，我们也可以自适应地选择温度，以同时实现基于激活的选择和均衡训练选择的双重优势。为验证这一点，我们还进行了实验，将 FedDSE 与硬 TopK 和软 TopK 进行对比，实验结果如表3.6所示（基于 EMNIST 数据集）。其中，同构 (1/4) 表示所有客户端都是同质的，且只能训练完整模型的 1/4；而异构能力（Heterogeneous capability）设置与表3.4 相同。而数据异质性中的同分布是指数据遵循独立同分布，也就是所有的客户端的数据分布相同。

表 3.6 温度对模型准确率的影响

模型计算能力	方法	数据异质性		
		高	低	同分布
同构 (1/4)	FedRolex	93.35	97.29	97.04
	FedDSE (T=0)	81.25	89.74	88.05
	FedDSE (T=1)	**96.59**	**98.21**	**97.83**
同构 (1/2)	FedRolex	97.76	98.52	98.74
	FedDSE (T=0)	91.51	96.53	95.24
	FedDSE (T=1)	**98.45**	**99.16**	**99.09**
异构	FedRolex	91.41	98.61	98.67
	FedDSE (T=0)	**95.34**	**98.65**	**98.69**
	FedDSE (T=1)	94.60	97.86	98.15

从表中可以观察到，$T=0$ 和 $T=1$ 在不同场景下分别表现更优。通常情况下，较高的温度更适用于客户端能力均匀的场景，例如在同构能力都为 1/4 的情况中，温度 $T=1$ 模型准确率的结果要远远好于当温度为 $T=0$ 的时候，特别是在高数据异质性的情况下，提高了接近十五个百分点；同样的情况还出现在同构模型能力为 1/2 情况下，也是在高异质情况下提高了接近七个百分点。而较低的温度则更适用于客户端能力不均的情况，例如在我们异构情况下，也就是客户端均匀分布 $\mathbf{r}=\{0.99,0.5,0.25,0.125,0.0625\}$ 情况下，$T=0$ 要比 $T=1$ 效果更好。此外，值得注意的是，我们的方法始终优于当前最先

进的基线方法，即 FedRolex。

3.5 本章小结

本文聚焦于联邦学习（Federated Learning, FL）中的子模型提取问题。我们观察到，由于统计异质性，客户端倾向于激活模型的不同神经元。这可能导致在不适当的子模型提取方式下，神经元之间出现竞争问题。为了解决这一挑战，我们提出了一种新的联邦学习子模型提取方法，该方法利用神经网络和边缘设备的激活分布特性，通过选择激活值最大的神经元，自适应地将其分配给不同的客户端。我们通过实验结果验证了其有效性，表现优于其他方法。

4 基于梯度子模型抽取的联邦学习方法

4.1 引言

在上一章节中我们基于深度学习中神经元最基础的生物学意义，就是在生物中神经元的活跃程度代表着其在当前状态下的重要性。在深度学习的模型中，将模型在推理过程中前向传播的输出值作为其活跃程度的依据，这也是上一章节中给不同边缘设备抽取子模型的重要依据。尽管上一章节中叙述的方法已经取得了很好的成绩，但确实是仅仅从启发性从整个模型训练过程中的前向传播过程考虑，没有考虑在整个训练过程中极为重要的反向传播过程的重要性。在反向传播过程中产生的各个神经元的梯度，这些梯度信息决定了模型的优化方向。

毫无疑问，在联邦学习中边缘设备与中心服务器端均是同构模型的话，模型最终取得效果是好于在资源受限情况下边缘设备训练与中心服务器侧不同的异构模型，也就是抽取的子模型。在这个基础之上，我们可以得到上一章节中的自适应算法并未考虑联邦学习整体的优化方向，也就是考虑在资源受限的情况下，自适应抽取子模型算法要做到与边缘设备资源充足情况下，模型的优化方向也就是梯度更新最为接近，这样我们就能得到与充足资源情况下相差最少的模型。反观论文中描述的以前的方法中，从预先制定规则出发考虑解决问题的策略，HeteroFL 设计了一个可以在资源受限情况下可以运行的系统，通过按层抽取神经元的可靠稳定的设计；FedRolex 仅仅考虑了在自己抽取规则之下，保证每个神经元受到的训练次数是一样的，从而保证模型最终的效果不至于出现很大的波动与偏差。在上一章节的方法中虽然采用了自适应的方法，但是仅仅是从前向传播也就是激活值的角度去考虑，而没有考虑到梯度更新的优化约束。综上这些方法都从来没有考虑到优化约束的问题。

站在局部与总体优化的角度来看，本章节提出了一个基于梯度的逼近子模型模型与服务器侧中心模型优化方向的方法 FedGSE。其核心点在于根据反向传播过程中计算的每层中神经元梯度，根据边缘设备的计算能力选择其中梯度绝对值大的组成边缘设备的子模型，从而使得子模型与中心侧全局模型更新最为接近。因为需要在中心侧模型上进行反向传播计算梯度，边缘设备的计算资源不足以支撑此等规模的运算，因此创建一个

样本量不大的公共数据集，也就是中心侧数据集使得反向传播的执行过程可以在中心侧运行，并且仅广播给边缘设备子模型参数即可完成，降低了中心侧与边缘设备之间的传输通讯量。总体来说本章节的贡献可以总结如下：

- 本文从优化的角度出发，指出了子模型本地更新与全局模型之间的差异是导致全局模型收敛效率降低的原因。为了缩小这一差距，我们提出了一种新方法 FedGSE，该方法选择梯度值较大的神经元，以确保全局模型和子模型的最小本地更新差异。
- 我们提供了数学证明，表明在相同训练情境下，FedGSE 算法能够实现全局模型和子模型之间最小的更新梯度差异。这证明了经过训练的子模型参数与全局模型参数最为接近。
- 为了评估我们提出的方法的有效性，我们将其与各种数据集和设置下的最近方法对比。大量实验结果表明，FedGSE 显著优于其他方法。

4.2 FedGSE 算法

本小节将详细介绍 FedGSE 方法中的各种细节。FedGSE 采用梯度作为抽取子模型的标准来解决联邦学习资源受限的场景，首先，我们需要在中心侧建立一个跟边缘设备数据集相似且包含大部分或者所有边缘设备需要用到的公共数据集，这个数据集用来在中心侧执行反向传播来获取各个神经元梯度。在获得公共数据集前提下，我们采用 SPL 或者 CSL 方法来根据边缘设备的历史模型来预测与边缘设备数据分布相似的公共子数据集，具体的 SPL 和 CSL 算法可见图4.1, 伪代码见算法4.2与算法4.3，然后使用公共子数据集在中心侧模型上执行反向传播产生梯度来抽取子模型，可见图4.2与算法3.1。之后，中心侧服务器将模型发送给客户端，在客户端侧使用边缘设备本身的数据集训练，从而完成整个模型的训练，整个 FedGSE 的整体代码见算法3.2。本小章节从公共数据集构建、相似公共子数据集产生、子模型抽取、端侧训练以及模型聚合来详细介绍 FedGSE 算法流程。本章节所有资源受限情况下联邦学习优化的目标相同，也就是优化公式2-4与公式2-5。

算法 4.1 FedGSE 算法

Input: 全局模型参数 $\mathbf{w}$, 学习率 η, 总通讯次数 T, 所有边缘设备计算能力 $\{r_0,\cdots,r_{N-1}\}$, 公共数据集 $\mathbb{D}^P$, 本地客户端训练次数 E

Output: T 轮的全局模型参数 $\mathbf{w}_{T+1}$

初始化全局模型参数 $\mathbf{w}_1$
procedure SERVER-SIDE OPTIMIZATION
  for 对每个轮次 $t \in \{1,\cdots,T\}$ **do**
    随机挑选所有客户端的子集 $\mathcal{N}_t$
    for 对于每个挑选到的客户端 n **并行执行 do**
      生成相似数据集 $\mathbb{D}_n^P \leftarrow$ **GetSimilarDataCSL** $(\mathbb{D}^P, \mathbf{w^n})$ or $\mathbb{D}_n^P \leftarrow$ **GetSimilarDataSPL** $(\mathbb{D}^P)$
      $\mathbf{M^{n,t}} \leftarrow$ **GetMask**(r_n, $\mathbb{D}_n^P$, $\mathbf{w_t}$)
      抽取子模型 $\mathbf{w_t^n} \leftarrow \mathbf{w_t} \odot \mathbf{M^{n,t}}$
      $\mathbf{w_{t+1}^n} \leftarrow$ **ClientLocalUpdata**(n, $\mathbf{w_t^n}$)
    end for
    更新全局模型参数 $\mathbf{w_{t+1}} = \sum_{n\in\mathcal{N}_t} \mathbf{P}_t^n \odot \mathbf{w_{t+1}^n}$
  end for
end procedure
procedure CLIENTLOCALUPDATE(n, $\mathbf{w_t^n}$)
  从服务器接受 $\mathbf{w_t^n}$
  for 从 1 到 E 迭代轮次 e **do**
    在本地数据集上更新本地模型 $\mathbf{w_{t,e+1}^n} = \mathbf{w_{t,e}^n} - \eta \nabla_{\mathbf{w_{t,e}^n}} f_n(\mathbf{w_{t,e}^n})$
  end for
  return $\mathbf{w_{t,E+1}^n}$
end procedure

4.2.1 公共数据集构建

在联邦学习中，每个客户端的数据都是独特的，因此只有公共数据集中的数据与大部分客户端中的数据拥有相似的分布，或者所有客户端的数据分布是公共数据集的一个子集，才能保证我们在后续在中心侧得到的神经元梯度与使用客户端数据得到是相近的。基于这样的前提，我们可以使用三种方法来构建我们的公共数据集：

（1）鼓励客户上传小部分自己产生的数据集。边缘侧用户会产生大量的数据，并不是所有的数据都包含大量的隐私信息，用户可以自行选择小部分可以被用来处理的信息上传到中心服务器侧以便提供更好的服务，这些小部分信息可能占用户本身数据集的百分之一不到。

（2）使用覆盖面积更广的数据集。因为后面我们使用的方法可以筛选出相似数据集，因此我们可以收集覆盖范围更广更多的数据来组成更大的数据集，也就是让客户端中的数据集成为我们收集公共数据集的子集，这样也能保证我们需要公共数据集的作用。

（3）使用生成式模型生成伪数据。在边缘设备可以训练生成式模型的前提下，可以使边缘设备根据自己的数据使得生成式模型生成数据，然后将这些伪数据上传到中心侧作为公共数据集。

4.2.2 边缘侧相似数据集产生策略

在构建完公共数据集之后，我们不是使用全部的公共数据集输入全局模型中去产生抽取子模型所需要的梯度。使用全部的公共数据集首先是效率比较低下，占用计算资源与时间较多。其次，筛选出与客户端数据集分布相似的公共子数据输入全局模型产生的梯度更加接近在客户端上执行的效果。因此，我们需要设定产生分布相似公共子数据集的策略，保证

$$P_{\mathbb{D}_n^P} \approx P_{\mathbb{D}_n} \tag{4-1}$$

其中，$\mathbb{D}_n$ 表示客户端 n 的数据集，$\mathbb{D}_n^P$ 表示与客户端 n 分布相似的公共子数据集，$P_{(\cdot)}$ 表是数据分布。我们使用两种策略来实现从公共数据集中筛选出与客户端分布相似的公共子数据集：

服务器侧挑选有标签数据集 (Server Picks samples through Labeled dataset SPL) 在服务器侧我们使用 SPL 去筛选公共数据集具有真是标签的数据，具体算法如图4.1与算

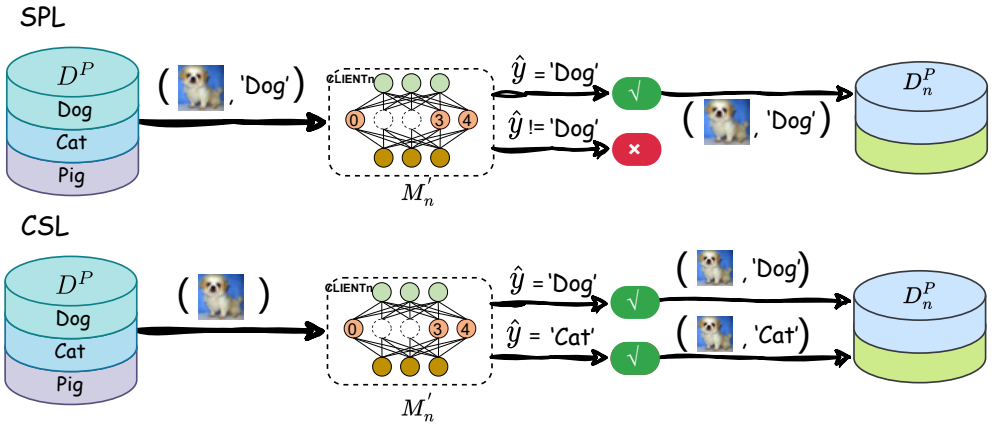


图 4.1 两种产生相似分布数据集的方法

法4.2所示。首先，公共数据集 $\mathbb{D}^P$ 中包含所有客户端所有类别的数据，将公共数据集中的所有数据都输入到客户端 i 的上次保存在服务器上的模型 M_i' 中，并预测每条数据在模型 M_i' 中的标签

$$\hat{y}_i = f(\mathbf{w}^n; x_i) \tag{4-2}$$

如果客户端历史模型预测正确，也就是 $\hat{y}_i$ 等于其真实标签 y_i，那么将该样本加入到 $\mathbb{D}_n^P$ 中；相反，如果预测结果不正确，那么就将这一条数据抛弃。客户端历史模型能预测正确说明模型大概率见过这些数据，也就是说当前数据大概率在客户端数据集的分布之中，符合我们的挑选的原则，具体代码如算法4.2所示。我们使用图4.1进行详细的讲解，图中公共数据集包含狗、猫以及猪三类数据，在 SPL 方法中，我们给模型输入了狗的照片以及真是标签“dog”，这个模型是客户端 n 在上个轮次中上传到中心侧的最新的模型，然后模型执行推理过程，预测输入狗的图片是什么，如果预测为“dog”就加入到与客户端模型相似分布的公共子数据集中，否则就直接抛弃。

客户模型设定无标签数据集 (Client sub-model to Set Labels of unlabeled public dataset CSL) 当在服务器侧使用 CSL 生成分布相似的公共子模型，具体介绍如图4.1和算法4.3所示，首先，公共数据集 $\mathbb{D}^P$ 中包含所有类别的数据，在 CSL 的算法之下，这些数据只需要保证存在即可，而不需要对应的真实标签。然后将公共数据集中的所有数据输入到模型 M_i' 之中，跟 SPL 一样，这个模型是客户端 n 最近一次上传给中心侧的模型，然后使用公式4-2在模型预测输入的数据的标签，与 SPL 不同的是，因为没有真实标签，我们直接将 $\hat{y}_i$ 作为输入数据的标签，用来使数据拟合模型分布，最后将输入数据跟预测标签

算法 4.2 GetSimilarDataSPL

Input: 公共数据集 $\mathbb{D}^P$, 客户端模型 $\mathbf{w^n}$

Output: 与客户端 n 相似分布的数据集 $\mathbb{D}_n^P$

procedure GETSIMILARDATASPL($\mathbb{D}^P, \mathbf{w^n}$)
    初始化 $\mathbb{D}_n^P = \emptyset$
    for 对于数据样例 $(x_i, y_i) \in \mathbb{D}^P$ **do**
        预测标签 $\hat{y}_i = f(\mathbf{w}^n; x_i)$
        if $\hat{y}_i == y_i$ **then**
            将样例加入 $\mathbb{D}_n^P \leftarrow \mathbb{D}_n^P \cup \{(x_i, y_i)\}$
        end if
    end for
    return $\mathbb{D}_n^P$
end procedure

算法 4.3 GetSimilarDataCSL

Input: 公共数据集 $\mathbb{D}^P$

Output: 与客户端 n 相似分布的数据集 $\mathbb{D}_n^P$

procedure GETSIMILARDATACSL($\mathbb{D}^P$)
    初始化 $\mathbb{D}_n^P = \emptyset$
    for 对于每个样例 $x_i \in \mathbb{D}^P$ **do**
        预测标签 $\hat{y}_i = f(\mathbf{w}^n; x_i)$
        将样例加入 $\mathbb{D}_n^P \leftarrow \mathbb{D}_n^P \cup \{(x_i, \hat{y}_i)\}$
    end for
end procedure

加入到公共相似分布子数据集中。我们使用图4.1下班部分详细介绍，在图中，输入的数据是一个狗的照片并且没有给出真实标签，模型跟在 SPL 中的一致的，都是上次客户端 n 最近上传的，接着就是给模型去预测这张照片的类别。在上面的路径中预测这张照片是“dog”，与我们认知是相同的，这个时候我们将狗的照片加预测标签“dog”添加到要返回的数据集中。在第二条路径中，模型预测出照片的标签是“cat”，与 SPL 方法不同的是，我们不抛弃这种情况下的数据，因为我们没有真实标签去做对比，也要讲这条数据也就是狗的照片加预测标签“cat”添加到要返回的数据集中。其实我们在这一过程要学习的是最近客户端上传模型在本地数据集上学习到的知识，我们通过预测过程将知识转移到公共数据集中，以方便我们得到数据分布相关的梯度。

4.2.3 子模型抽取

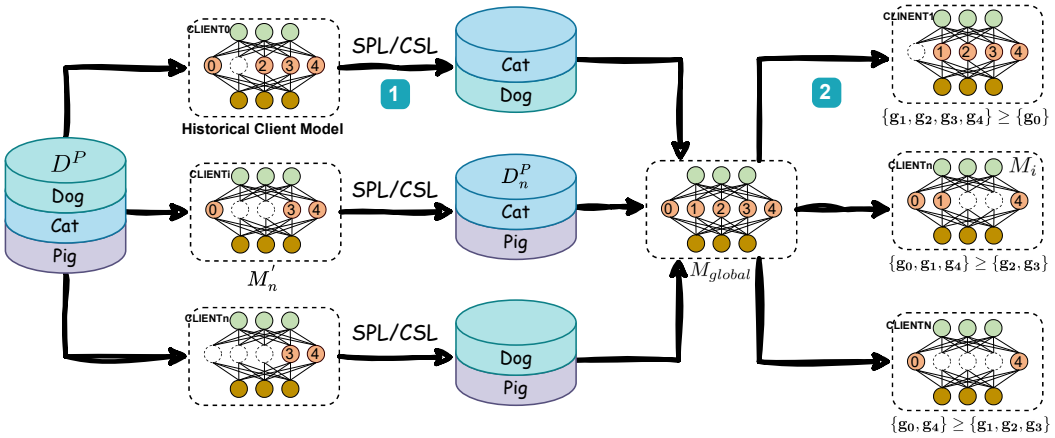


图 4.2 FedGSE 算法中子模型抽取

不同于 FedDSE 在客户端计算挑选，FedGSE 整个子模型计算挑选的过程都在中心侧服务器上进行。算法3.1详细描述了子模型抽取过程中生成神经元掩码（mask）的过程。首先在上一小章节中我们获得了相似数据集分布 $\mathbb{D}_n^P$，服务器将相似数据集分布输入到中心侧全局模型中，经过反向传播获得整个全局模型所有神经元的梯度信息 $\mathbf{g}$，然后，服务器会从全局模型中逐层挑选神经元组成子模型。

我们将激活值的梯度的绝对值定义为每个神经元的梯度值。具体来说，对于数据集 $\mathbb{D}_n^P$ 中的每条数据 (x, y)（其中标签 y 可以是来自 SPL 真实标签或者是来自 CSL 伪标签），

算法 4.4 GetMaskFedGSE

Input: 客户端计 n 算能力系数 r_n, 相似分布公共子数据集 $\mathbb{D}_n^P$, 中心侧全局模型 $\mathbf{w_t}$

Output: 客户端 n 在 t 轮的掩码 $\mathbf{M^{n,t}}$

procedure GETMASK(r_n, $\mathbb{D}_n^P$, $\mathbf{w_t}$)
  使用公式4-3计算神经元梯度 $\mathbf{g^n}$
  for 对于每层 $l \in \{1,2,\cdots,L\}$ **do**
    $\mathbf{S}_{sorted} \leftarrow \mathbf{sort}([g^n_{l,1},\ldots,g^n_{l,m_l}])$
    $\mathbf{S}_{top-r} = \mathbf{S}_{sorted}[1 : r \cdot m_l]$
    if $\mathbf{h_{l,i}} \in \mathbf{S}_{top-r}$ **then** $\mathbf{M^{n,t}_{l,i}} = 1$
    else $\mathbf{M^{n,t}_{l,i}} = 0$
    end if
  end for
  return $\mathbf{M^{n,t}}$
end procedure

第 l 层第 i 个神经元的激活值 $\mathbf{h}_{l,i}$ 的梯度是通过 $\frac{\partial f(\mathbf{w};x,y)}{\partial \mathbf{h}_{l,i}(k,v)}$ 来计算。对于卷积神经网络来说，其中第 l 层第 i 个神经元的激活值是一个特征图 $\mathbf{h}_{l,i} \in R^{K\times V}$，其宽度为 K，长度为 V，我们使用平均值的绝对值来表示神经元梯度的大小：

$$g_{l,i} = \sum_{k=0}^{K-1}\sum_{v=0}^{V-1}\Big|\sum_{(x,y)\in\mathbb{D}_n^P}\frac{\partial f(\mathbf{w};x,y)}{\partial \mathbf{h}_{l,i}(k,v)}\Big| \tag{4-3}$$

使用等式 4-3 来定义神经元梯度主要有两个原因。首先，现有的研究工作[70]已经表明，梯度较大的神经元更为重要，这与我们挑选神经元的原则是一致的。其次，在下一章节中，我们会证明选择梯度较大的神经元会使得提取的子模型的梯度更接近全局完整模型。

在每一层中，基于神经元梯度大小选择其中梯度绝对值相对较大的神经元。与其他神经元相比，梯度绝对值较大的表示改变这些神经元对模型最终的损失影响更大[70]，也就意味着这些神经元更加重要。我们规定客户端的计算能力系数为 $r(0 < r < 1)$，对于客户端 n 第 1 层，我们仅保留该层中具有高梯度绝对值的 $r \cdot m_l$ 个神经元（即 $\max_{i=1}^{r\cdot m_l}\{g_{l,1},g_{l,2},\ldots,g_{l,m_l}\}$），同时剪裁掉其他神经元。我们在上述的挑选过程中产生一个不同神经元对应的二进制掩码 $\mathbf{M}$，挑选到的神经元在对象矩阵位置填写为 1, 否则就

是 0，例如，$\mathbf{M}^{\mathbf{n}}_{\mathbf{l},\mathbf{i}}=0$ 表示第 l 层的第 i 个神经元被剪除，而 $\mathbf{M}^{\mathbf{n}}_{\mathbf{l},\mathbf{i}}=1$ 表示保留该神经元。根据这样的挑选规则也就是算法3.1中所呈现的。通过将全局模型参数与二进制掩码点乘法得到子模型参数，客户端 n 的子模型参数为 $\mathbf{w}^{\mathbf{n}}=\mathbf{w}\odot\mathbf{M}$。

4.2.4 端侧训练以及模型聚合

FedGSE 的整体工作流程如算法 4.1所示。在每一轮中，服务器首先为每个客户端建立数据集，这些数据集的分布与它们各自的本地数据集相似（第 6 行），然后为每个客户端提取子模型（第 7-8 行）。之后，服务器将子模型发送给每个选中的客户端 n（第 9 行）。接着，每个客户端 n 使用其整个本地数据集对接收到的子模型进行本地训练（第 16-18 行）。然后，每个客户端将其子模型的更新参数发送回服务器（第 19 行）。最后，服务器聚合这些参数以更新全局模型（第 11 行），更新与聚合方式详见 FedDSE 章节。

4.3 理论分析

在本节中，我们首先证明选择幅度较大的神经元梯度也等同于选择了重要的参数。然后，我们证明 FedGSE 可以最大程度地减少子模型与完整全局模型之间的梯度差异。

引理 1 *参数的梯度与其对应激活值的梯度成正比，也即，* $\frac{\partial f(\mathbf{w})}{\partial \mathbf{w}_{l,i}}=\mathbf{k}\cdot g_{l,i}$，*其中* $\mathbf{k}$ *是比例系数。*

证明：我们考虑一个将 ReLU 作为激活函数，

$$\mathbf{h}_{l,i}=\max\{0,\mathbf{w}_{l,i}^{T}\mathbf{h}_{l-1}+b_{l,i}\} \tag{4-4}$$

方程4-4表示具有 ReLU 激活函数的全连接。然后我们可以得到 $\frac{\partial \mathbf{h}_{l,i}}{\partial \mathbf{w}_{l,i}}=\mathbf{h}_{l-1}\ or\ 0$（除 $\mathbf{h}_{l-1}$ 使得 $\mathbf{w}_{l,i}^{T}\mathbf{h}_{l-1}+b_{l,i}=0$，因为 $\frac{\partial \mathbf{h}_{l,i}}{\partial \mathbf{w}_{l,i}}$ 在该点不可微）。

$$\frac{\partial f(\mathbf{w})}{\partial \mathbf{w}_{l,i}}=\frac{\partial f(\mathbf{w})}{\partial \mathbf{h}_{l,i}}\frac{\partial \mathbf{h}_{l,i}}{\partial \mathbf{w}_{l,i}}=\frac{\partial f(\mathbf{w})}{\partial \mathbf{h}_{l,i}}\mathbf{h}_{l-1}=g_{l,i}\mathbf{h}_{l-1} \tag{4-5}$$

根据反向传播的链式法则和等式4-4的结论，我们有 $\frac{\partial \mathbf{h}_{l,i}}{\partial \mathbf{w}_{l,i}}=\mathbf{h}_{l-1}$ 或者 0。很容易得出，参数的梯度与其对应激活值的梯度成正比（即引理 1）。

引理1表明，$\frac{\partial f(\mathbf{w})}{\partial \mathbf{w}_{l,i}}$ 和 $g_{l,i}$ 之间存在线性关系。这意味着选择一个相对较大的 $g_{l,i}$ 等价于选择一个相对较大的 $\frac{\partial f(\mathbf{w})}{\partial \mathbf{w}_{l,i}}$。

定理 1 *考虑到 $\mathbb{D}_n^P$ 和客户端真实本地数据集 $\mathbb{D}_n$ 有着相近的分布, 从算法4.4得到子模型参数 $\mathbf{w}_*^n$ 与全局模型参数 $\mathbf{w}$ 是最小的距离。具体来说, 我们有*

$$\|\nabla f(\mathbf{w};\mathbb{D}_n)-\nabla f(\mathbf{w}^n;\mathbb{D}_n)\| \geq \|\nabla f(\mathbf{w};\mathbb{D}_n)-\nabla f(\mathbf{w}_*^n;\mathbb{D}_n)\|$$

对于任何从全局模型 $\mathbf{w}$ 上抽取出来的子模型参数 $\mathbf{w}^n$ 都成立, 其中 $\mathbf{w}^n$ 和 $\mathbf{w}_*^n$ 拥有相同形状。

证明：我们的策略通过以下方式获得子模型的掩码 $\mathbf{M}$

$$\mathbf{M} = \mathbf{GetMaskFedGSE}(r,\mathbb{D}_n^P,\mathbf{w}) \tag{4-6}$$

其中 r 是客户端 n 的容量率。然后，我们通过以下方式获得子模型参数 $\mathbf{w}_*^n$

$$\mathbf{w}_*^n = \mathbf{M}\odot\mathbf{w} \tag{4-7}$$

我们考虑通过算法4.4获得的子模型 $\mathbf{w}_*^n$ 的第 l 层的梯度。同时定义任意子模型 $\mathbf{w}^n$ 的第 l 层的梯度

$$\{\nabla f(\mathbf{w}^n;\mathbb{D}_n^P)\}_l = \{\overbrace{g_1^n,\ldots,g_{r\cdot m_l}^n}^{r\cdot m_l},\overbrace{0,\ldots,0}^{(1-r)\cdot m_l}\} \tag{4-8}$$

$$\{\nabla f(\mathbf{w}_*^n;\mathbb{D}_n^P)\}_l = \{\overbrace{g_1^*,\ldots,g_{r\cdot m_l}^*}^{r\cdot m_l},\overbrace{0,\ldots,0}^{(1-r)\cdot m_l}\} \tag{4-9}$$

其中 m_l 是第 l 层神经元的总数。其中有 $r\cdot m_l$ 个神经元的梯度非零，$(1-r)\cdot m_l$ 个神经元的梯度为零。不失一般性，我们假设等式（4-8）是按降序排列的，也就是 $g_1 \geq g_2 \geq \cdots \geq g_{r\cdot m_l} \geq 0$，等式（4-9）也是同样的顺序。

根据等式（4-6）和等式（4-7），我们可以得到 $g_1^* \geq g_1^n,\ldots,g_{r\cdot m_l}^* \geq g_{r\cdot m_l}^n$，这是因为 FedGSE 就是挑选其中神经元梯度大组成子模型的。然后，很明显可以得出，

$$\sum_{i=0}^{m_l}|g_{l,i}-g_{l,i}^n| \geq \sum_{i=0}^{m_l}|g_{l,i}-g_{l,i}^*| \tag{4-10}$$

其中 $g_{l,i}$ 是全局模型 $\mathbf{w}$ 在第 l 层的第 i 个神经元的梯度。当第 i 个神经元被剪枝后，$g_{l,i}^n$ 和 $g_{l,i}^*$ 都按照 0 处理。等式（4-10）表明，对于每一层，$\mathbf{w}$ 和 $\mathbf{w}^n$ 之间的梯度差异小于 $\mathbf{w}$ 和 $\mathbf{w}^n$。因此，在总结所有层时，我们可以得出以下结论：

$$\|\nabla f(\mathbf{w};\mathbb{D}_n^P)-\nabla f(\mathbf{w}^n;\mathbb{D}_n^P)\| \geq \|\nabla f(\mathbf{w};\mathbb{D}_n^P)-\nabla f(\mathbf{w}_*^n;\mathbb{D}_n^P)\| \tag{4-11}$$

根据等式（4-1），$\mathbb{D}_n^P$ 与 $\mathbb{D}_n$ 相似，我们可以通过在等式（4-11）中将 $\mathbb{D}_n^P$ 替换为 $\mathbb{D}_n$ 来推导出定理 1。

定理 1证明了使用由我们的算法4.1 所选的子模型在本地数据集上进行模型更新，可以最小化使用子模型进行本地更新与使用原始全局模型进行本地更新之间的差异。

4.4 实验过程与结果分析

本章节解决的问题是章节 3的改进和提升，因此大部分的实验设定与章节 3应用同样设定。但是两种改进方法适用于不同的场景，FedDSE 方法主要解决在低质量客户端分布的场景下，指的是所有的客户端都不能训练中心侧全局模型。而 FedGSE 方法则是应用在高质量客户端分布场景下，在这种场景下客户端中分布着一部分可以训练中心侧的模型，本章节的实验结果均是在这种场景下进行的。

表 4.1 FedGSE 不同数据集训练参数设置

	EMNIST	CIFAR-10	CIFAR-100	TinyImageNet
本地训练次数 (E)	2	2	2	2
学习率	0.001	0.001	0.001	0.001
训练轮次	800	2500	2500	2500
优化器	SGD	SGD	SGD	SGD
动量	0.9	0.9	0.9	0.9
权重衰减	5.00E − 04	5.00E − 04	5.00E − 04	5.00E − 04
相似数据量	128	128	128	128

与在 FedDSE 上一样，我们在数据集 EMNIST 上训练四层卷积神经网络，同样，使用 Resnet18 处理数据集 CIFAR10 和 CIFAR100 以及使用 Resnet34 处理数据集 TinyImagenet，详见章节3.3.1。对于实验设置中的数据异质性还是分为高数据异质性与低数据异质性，参见表3.2。而 FedGSE 的模型异质性则是在高质量客户端分布场景，具体来讲就是设定五种客户端计算资源分别是 $\mathbf{r}=\{1,0.5,0.25,0.125,0.0625\}$，与上一章节不同的是本章节实验中包含了百分之二十比例的可以训练中心侧全局模型的客户端，表4.2展示了不同抽取方法中不同模型平均的参数量级和 MACs。我们对比的 baseline 与 FedDSE

是一致的，也即 HeteroFL、FedRolex、Federated Dropout 和 DepthFL。表4.1中详细描述了本章节实验所设定的参数，其中大部分的设定与 FedDSE 一致，但是最后一行相似数据表示输入在中心侧模型中用于反向传播梯度的数据量。整个训练过程在 PyTorch 框架上进行，本章节实验所用的设备是 Nvidia 4090。最后因为实验是关于分类问题，评估指标选择的是在整个测试集上模型准确率。在训练过程中，我们采取了与章节3.3.3中一致的策略，掩码交叉熵损失、放缩模块以及静态批归一化。以上是本章节实验的所有设置以及详细介绍与 FedDSE 相似以及存在区别的地方。

表 4.2 不同抽取方法平均模型参数量/MACs

抽取方法	CONV	ResNet18	ResNet34
DepthFL	1319k/35.80M	4.48M/378.7M	8.54M/745.37M
others	441.19k/16.12M	3.08M/155.12M	5.9M/331.57M

4.4.1 实验结果

表 4.3 高质量客户端分布场景中低数据异质性下不同方法准确率对比

方法	EMNIST	CIFAR-10	CIFAR-100	TinyImageNet
HeteroFL	96.11	56.08	25.95	21.82
Federated Dropout	88.76	52.57	15.79	19.25
FedRolex	93.94	57.28	21.53	22.96
DepthFL	97.69	63.90	31.38	24.85
FedGSE_CSL	**98.06**	**65.89**	**31.46**	**25.95**
FedDSE_SPL	97.44	65.82	28.87	25.33

表4.3与表4.4 展示了我们在相同实验设置下执行的实验结果，以确保公平性。结果表明，我们的方法 FedGSE 相比于其他方法中表现更优，特别是在高数据异质性情况下，相较于 HeteroFL、Federated Dropout 和 FedRolex 在 EMNIST、CIFAR10、CIFAR100 和 TinyImageNet 上的最佳表现，分别提升了 1.33%, 8.54%, 2.92% 和 2.37%。此外，我们的方法在低数据异质性情况下也表现出色，相较于其他三种方法的最佳表现分别有

表 4.4 高质量客户端分布场景中低数据异质性下不同方法准确率对比

方法	EMNIST	CIFAR-10	CIFAR-100	TinyImageNet
HeteroFL	98.65	73.42	32.13	29.56
Federated Dropout	97.53	65.28	19.81	26.04
FedRolex	98.56	71.68	27.60	30.01
DepthFL	**99.14**	**76.13**	**33.24**	29.83
FedGSE-CSL	98.74	73.51	32.65	30.02
FedDSE-SPL	98.77	74.66	32.40	**30.49**

0.12%, 1.24%, 0.27% 和 0.39% 的提升。上述现象表明，我们的方法在高数据异质性情况下具有显著优势，这验证了我们的理论，即数据异质性越高，我们的方法越能有效选择重要神经元来建模这些数据。Federated Dropout 在大多数场景中表现相对较差，主要是因为其动态选择神经元的机制。对于简单的数据集 EMNIST，我们的方法与其他方法在准确率上的差异较小，无论是在高还是低数据异质性情况下，这是因为 CNN 模型很容易训练 EMNIST，因此所有方法都能获得相对较好的结果。对于 CIFAR10，FedGSE 展示了其在算法设计上的优越性，相较于 FedRolex 分别有 8.54% 和 2.98% 的优势。随着数据异质性的降低，优势有所缩小，但仍保持领先。尽管在困难的数据集 CIFAR-100 和 TinyImageNet 上，所有方法的表现都不如其他数据集，但我们的方法仍然领先于其他方法。总体而言，FedGSE 在低数据异质性和更具挑战性的高数据异质性场景下均优于 HeteroFL、Federated Dropout 和 FedRolex，特别是在高数据异质性情况下。尽管 DepthFL 根据表4.2具有更大的参数和 MACs，但在高数据异质性情况下的表现仍不如 FedGSE，但在低数据异质性情况下，DepthFL 取得了一定的优势。

4.4.2 对比 FedGSE-CSL 与 FedDSE-SPL

FedGSE-CSL 与 FedGSE-SPL 的核心区别在于获取相似分布数据的方法，FedGSE-CSL 使用的是客户模型设定无标签数据集（CSL），而 FedGSE-SPL 使用的是服务器侧挑选有标签数据集（SPL）。在高数据异质性的情况下，FedDSE-SPL 无法达到 FedGSE-CSL 的水平。由于客户端上传的模型是基于本地数据集进行训练的，而本地数据集包含的类别数量较少（例如，在 EMNIST 和 CIFAR10 中，类别数量为 2），因此 FedDSE-SPL 难

以选择所有应该被选中的数据。然而，在低数据异质性的情况下，客户端数据集包含的类别数量更多，如表3.2所示。这提供了一种具有鲁棒性容忍度的方法，即少量预测错误不会影响整体神经元选择。相比之下，FedGSE-CSL 的错误标签对其准确性下降起到了显著作用。

4.4.3 累积梯度差异分析

我们设计了在 EMNIST 数据集上使用简单卷积神经网络的实验，以分析不同训练轮次中的累积梯度差异。实验的目的是确定全局模型参数与子模型参数之和之间的差异。在同一轮次中使用相同的本地数据进行训练后，计算全局模型与子模型之间更新参数的差异。在图4.3中，纵坐标的值表示的是通讯轮次，本次实验总共设定了 100 个通讯

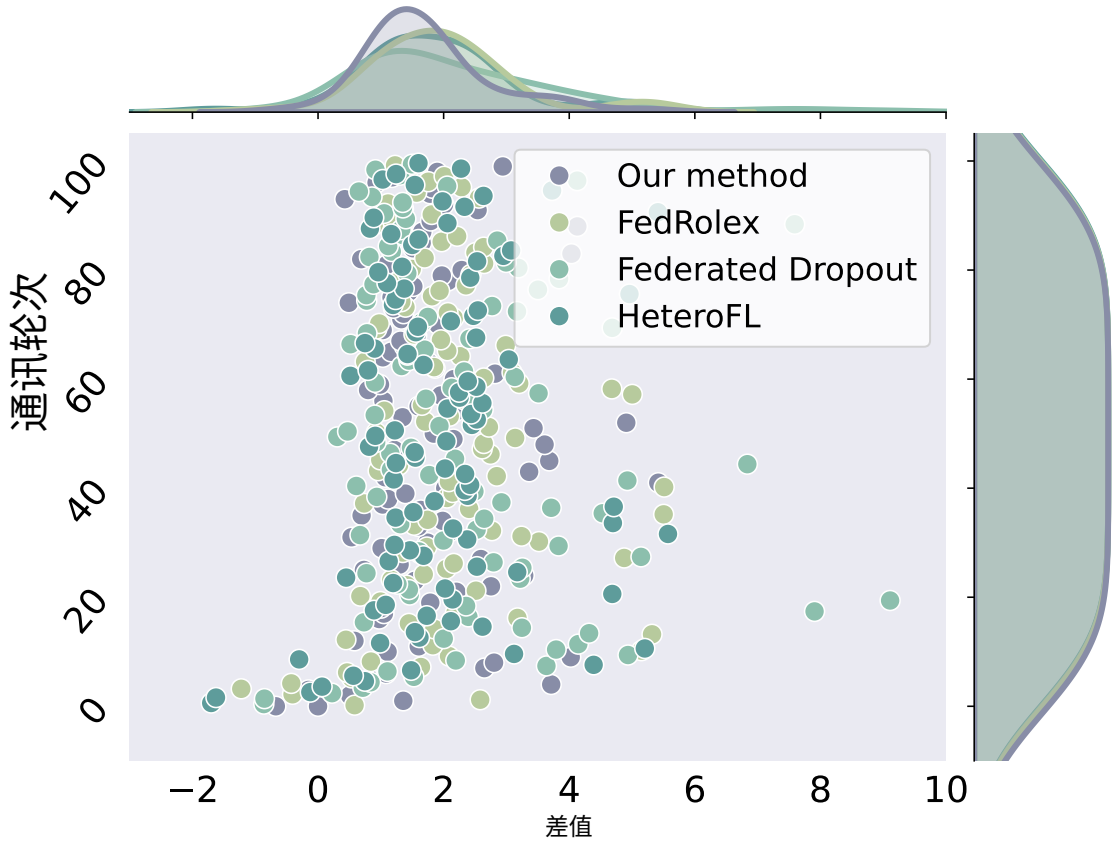
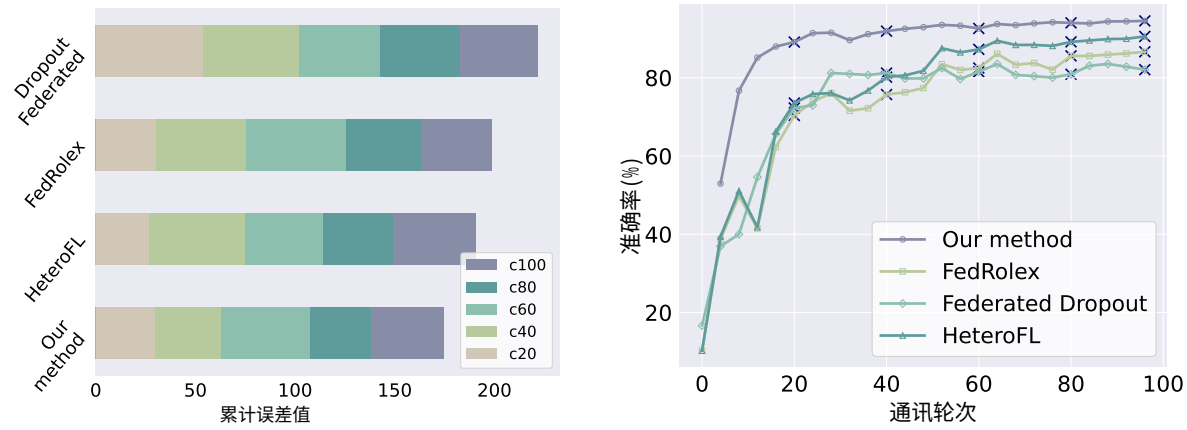


图 4.3 不同通讯轮次全局模型与子模型梯度差值

轮次，横坐标表示的是差值，也就是子模型的梯度总和与全局模型梯度总和的差值。上方的曲线图表示的是在所有轮次中统计的差值出现的次数的曲线图。通过曲线图可以发现 FedGSE-CSL 方法峰值更偏向于 0 附近也就是差值集中在较小的值附近，相比于其他

方法更为明显，峰值明显往右边偏移，这就意味着在整个训练的 100 个轮次中，整体差值 FedGSE-CSL 是远远小于其他方法的。



(a) 特定通讯轮次下累计梯度差值 **(b) EMNIST 不同方法的准确率**

图 4.4 累计误差与准确率关系

具体来说，我们对通信轮次 $\{20,40,60,80,100\}$ 中的累积误差（**Accumulated Differences** AD）进行了统计分析，也就是计算前 20、40、60 、80 以及 100 轮次之前的所有轮次误差的总和，结果如图4.4a所示。其中纵坐标表示不同方法，而横坐标表示的是具体的差值数据。可以明显观察到，在每一个统计轮次中，累计误差的大小关系均为 $\{Ad_{FederatedDropout} > Ad_{FedRolex} > Ad_{HeteroFL} > Ad_{FedGSE}\}$，这表明 FedGSE-CSL 有效地缩小了全局模型与子模型之间的更新梯度参数差距距离。可以使得客户端朝最优方向训练。根据图4.4b，准确率的结果为 $\{Acc_{FederatedDropout} < Acc_{FedRolex} < Acc_{HeteroFL} < Acc_{FedGSE}\}$，这与累积误差的恰好结果相反。毫无疑问可以合理推断出较小的累积误差往往可以得到更高的准确率。

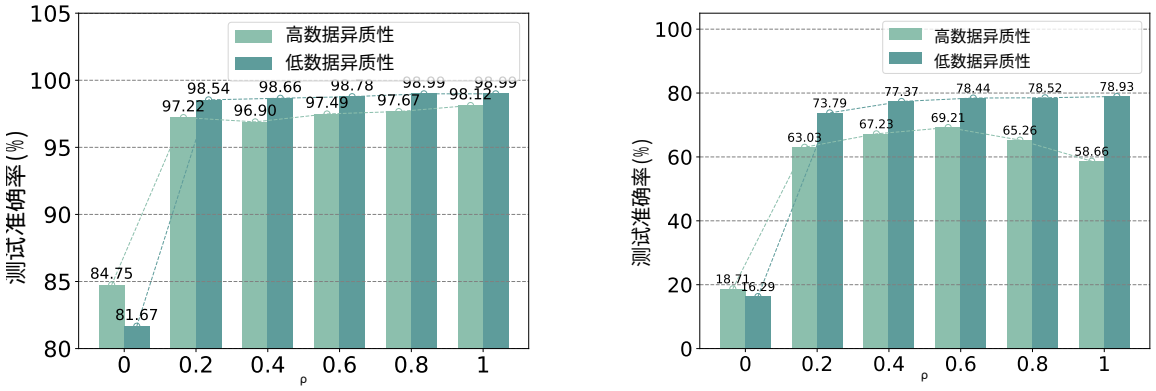
4.5 消融实验

4.5.1 客户端计算能力分布对模型能力的影响

在之前的实验中，客户端的容量是均匀设置的。为了探究客户端模型异质性分布的影响，在高客户端质量分布的情况下，我们选择了容量 $\{1,\frac{1}{16}\}$（低客户质量分布中没有 1），并调整了两者之间的分布比例（定义为 ρ），其中 $\rho=1$ 表示所有客户端都具有最大

容量模型为 $\{1\}$ 的情况，而 $\rho=0$ 表示所有客户端都具有最小容量模型为 $\{\frac{1}{16}\}$ 的情况。

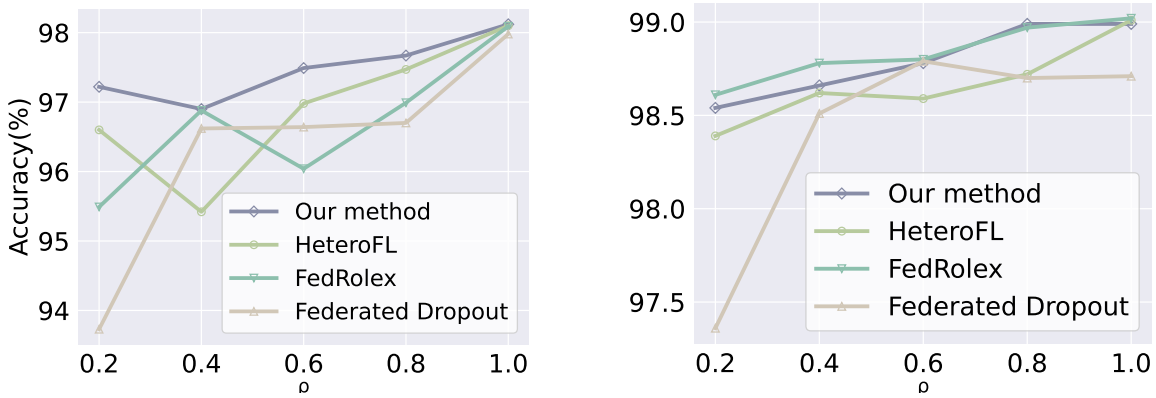
FedGSE 下高数据异质性和低数据异质性的对比分析 图4.5a和图4.5b展示了在 EM-



(a) EMNIST 下高数据异质 **(b) EMNIST 下低数据异质**

图 4.5 FedGSE 下高数据异质性和低数据异质性的对比分析

NIST 和 CIFAR10 数据集上，当 ρ 从 0 变化到 1 时，使用我们的方法时全局模型的变化情况。我们可以观察到：（1）当 ρ 从 0 到 0.2 时，无论是在高数据异质性还是低数据异质性下，EMNIST 和 CIFAR10 的全局模型准确率都经历了一个显著的提升。这表明模型规模是导致上述情况的主要原因。（2）对于简单的数据集 EMNIST，在 ρ 的广泛范

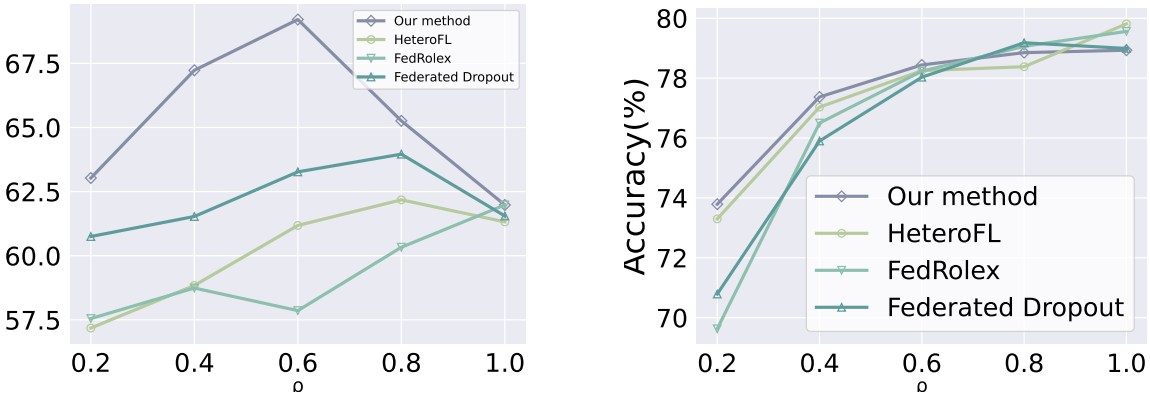


(a) EMNIST 下高数据异质 **(b) EMNIST 下低数据异质**

图 4.6 EMNIST 上 ρ 对准确率影响曲线图

围内（从 0 到 1），高数据异质性和低数据异质性之间的全局准确率存在差距。这是因

为 EMNIST 是一个简单的任务，四层卷积神经网络模型可以轻松达到 85% 的全局模型准确率。因此，全局模型准确率受到数据异质性水平的限制，而不是模型容量。这一结果揭示了数据异质性的挑战无法通过增加模型容量来解决。（3）对于 CIFAR10，高数

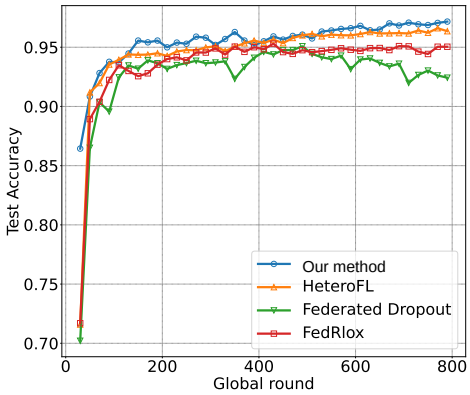


(a) EMNIST 下高数据异质 **(b) EMNIST 下低数据异质**

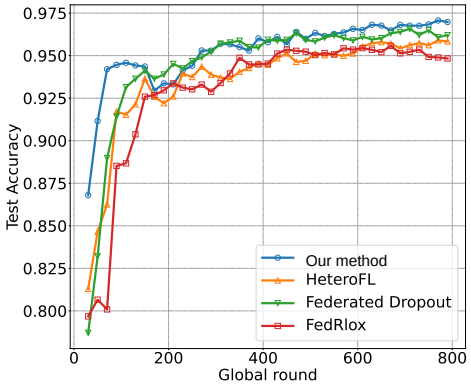
图 4.7 CIAFR10 上 ρ 对准确率影响曲线图

据异质性和低数据异质性之间的全局准确率存在较大差距。全局模型准确率受到模型最大容量和数据异质性水平的限制。具有 $\frac{1}{16}$ 容量的客户端模型根本无法解决此类问题，导致准确率极低。这导致了较大的差距。

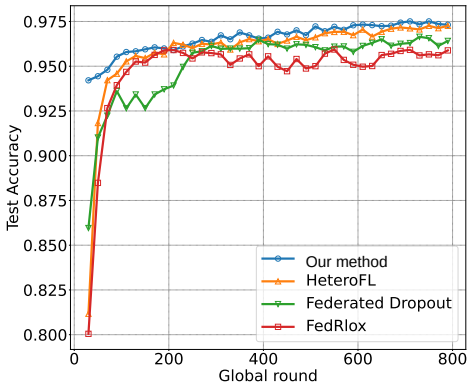
不同方法的对比分析 如图4.6a和图4.7a所示，在高数据异质性场景下，我们的方法在整个 ρ 从 0.2 到 1 的变化过程中，相较于其他方法具有压倒性的优势。这表明，增加模型的能力并不一定能够提高全局准确率，而在高数据异质性场景下，良好的算法设计可以提升模型的准确率。在困难场景下，我们的方法远远超过了其他模型，证明了我们的算法在这种情况下的优势。而在低数据异质性场景下，如图4.6b和图4.7b所示，我们的方法在大多数情况下表现良好。在这种简单场景下，大多数方法都能取得良好的结果，这表明在当前场景下，全局准确率的限制主要来自于模型的能力，提升模型的能力可以提高效果。特别是在图4.7a中，在具有高数据异质性的困难数据集上，增加模型的能力并不能稳定地提高模型的准确率。综上所述，在针对高数据异质性的严格场景中，我们的方法比其他方法更具适用性。图4.8 展示了在 EMNIST 数据集上的特殊训练过程，可以看到在大多数的训练过程中，我们的方法全程都优于其他的方法，在少部分中处于焦灼的状态。



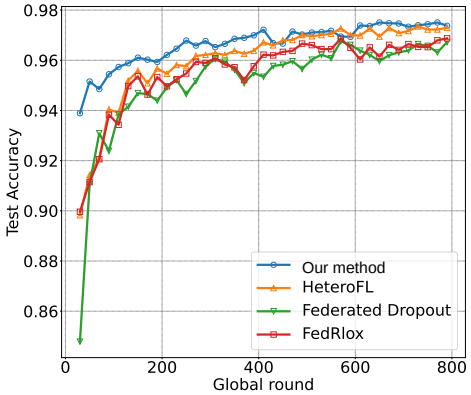
(a) $\rho = 0.2$



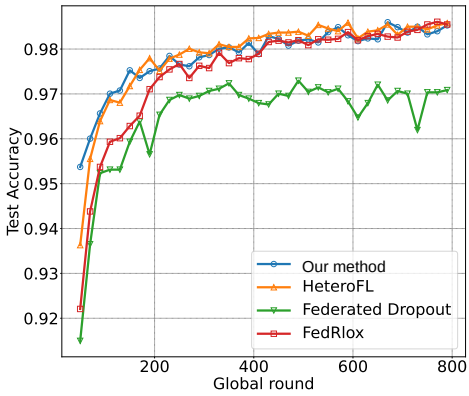
(b) $\rho = 0.4$



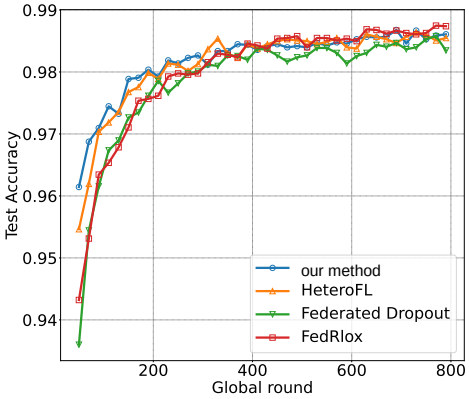
(c) $\rho = 0.6$



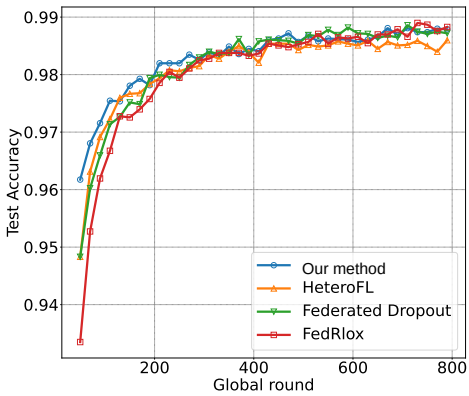
(d) $\rho = 0.8$



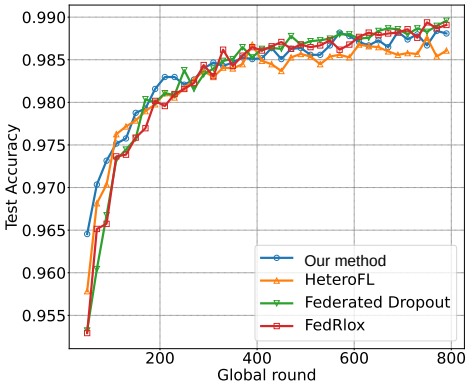
(e) $\rho = 0.2$



(f) $\rho = 0.4$



(g) $\rho = 0.6$

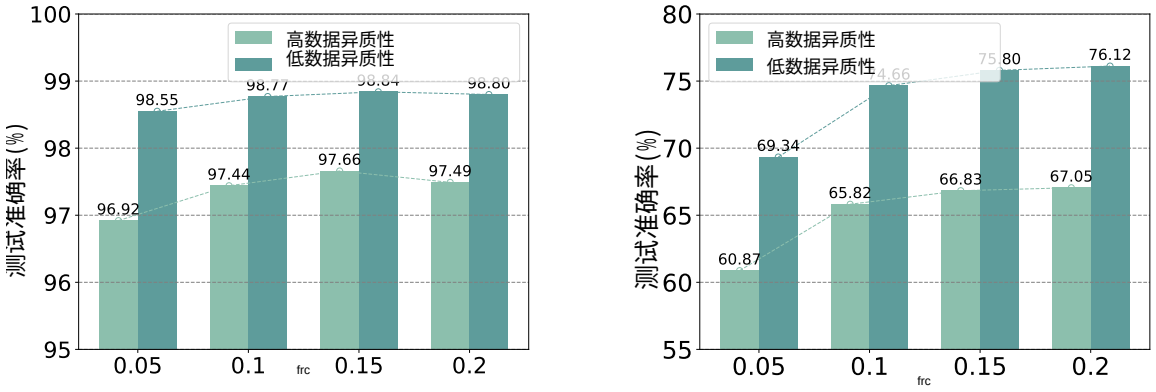


(h) $\rho = 0.8$

图 4.8 高低数据异质性下 EMNIST 数据训练细节

4.5.2 客户挑选比例对准确率的影响

为了探究每轮训练过程中选择通信客户端数量的影响，我们将 frc（表示通信客户端占总客户端的比例）设置为 $\{0.05, 0.10, 0.15, 0.20\}$。我们将客户端总数设置为 100，每轮



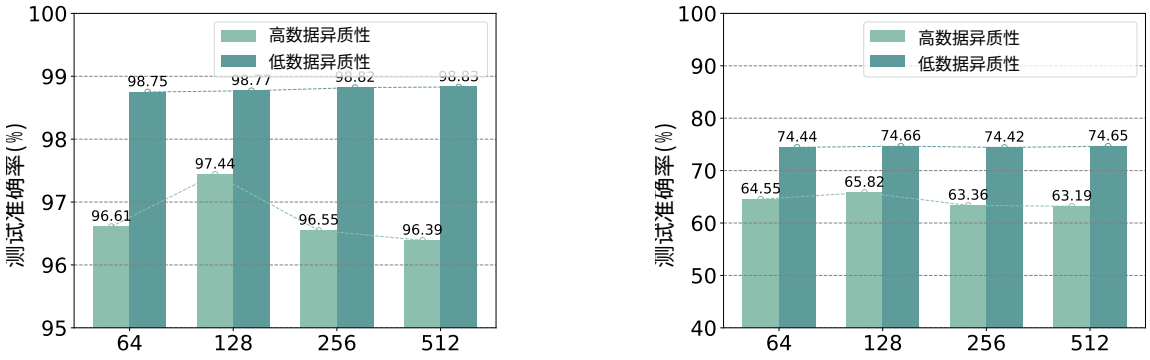
(a) EMNIST **(b) CIAFR10**

图 4.9 客户挑选比例对准确率的影响

对应的通信客户端数量分别为 $\{5, 10, 15, 20\}$。图4.9a和图4.9b展示了实验结果。我们可以得出以下三个结论：（1）当 $frc = 0.05$ 全局模型准确率明显低于 $frc = \{0.10, 0.15, 0.20\}$ 这表明，当通信客户端数量相对较少时，全局模型中更新的参数较少，从而导致准确率下降。（2）当 frc 从 0.10 增加时，全局模型准确率保持在相对相同的水平，有时甚至会下降。这是因为当 frc 超过 0.1 时，有足够的通信客户端参数在全局模型中更新，因此准确率保持在一个稳定的水平。当 frc 继续增加时，参数之间的竞争加剧，有时会导致准确率下降。（3）当 frc 增加到 0.2 时，从图4.9a和图4.9b中可以观察到，所有准确率比较都显示出与之前相比的轻微波动或显著下降。这是因为当每轮中有更多的客户端时，客户端之间可能会对同一神经元进行更新，从而导致准确率下降。

4.5.3 反向传播数据规模对准确率的影响

我们设置不同数量的相似数据探究这对全局模型准确率的影响。这些数据是指每次在抽选子模型时候，我们选择多少规模的数据进行反向传播，利用这些数据得到的梯度开展子模型抽取工作。图4.10a和图4.10b的实验结果显示，当相似数据的数量从 64 变化到 512 时，我们可以观察到低数据异质性并未受到显著影响，全局准确率保持在同一水



(a) EMNIST **(b) CIFAR10**

图 4.10 推理数据规模对准确率的影响

平。这是因为低数据异质性具有更均衡的分布，因此改变相似数据的大小对选择重要神经元的能力影响较小，当数据处于低数据异质性的时候，无论选择多少数据进行方向传播，只要不是数量过于少，都能得到比较均匀的分布数据，抽取的子模型的波动也会更少。然而，在高异质性数据上，相似数据的大小导致了全局准确率的显著波动。高异质性数据的分布更加集中，相似数据的大小和质量在选择重要神经元方面起着至关重要的作用，也就是说明如果在少量的数据中没有抽取到当前客户端最相似的数据，那么抽取的子模型就会与本地数据产生很大的偏离，从而造成训练效果变差。这意味着选择适当大小的相似数据比拥有更大的数量更为重要，因为可以最大限度的保证使用最相似数据进行反向传播。此外，数据的质量以及相似数据与客户端数据集之间的相似性也是影响准确率的重要因素。

4.5.4 统计异质性的影响

表 4.5 FedGSE 不同数据集训练参数设置

数据集	客户数据集包含种类数				
	2	4	6	8	10
emnist-conv	97.44	98.77	98.94	98.81	99.01
cifar10-resnet	65.82	74.66	75.04	76.38	74.38

为了研究数据异质性程度对模型准确率的影响，我们进行了表4.5中的实验。我们

将 s（客户端本地数据集中包含的类别数量）从 2 变化到 10。从实验结果中可以看出，当 s 从 2 变化到 4 时，全局模型准确率经历了一个显著的跳跃，之后准确率保持在一个稳定的阶段，表选为略有提升或者有着轻微的下降。实验结果表明，数据异质性程度对模型的影响存在一个阈值，当超过这个阈值时，其影响程度会降低。显然，EMNIST 和 CIFAR10 的阈值都是 $s=2$。可以观察到，在达到阈值后，准确率没有发生显著变化。这表明 FedGSE 在处理非独立同分布（non-IID）数据方面具有优势，即使在数据异质性较低且数据均匀分布的情况下，它也能达到相似的性能水平。这清楚地展示了我们算法设计的优越性。

4.6 本章小结

在本文中，我们重点研究了全局模型与提取模型之间的更新关系。然后，我们设计了 FedGSE，旨在使子模型的更新尽可能接近全局模型。我们通过理论证明，使用我们的方法更新的参数最接近全局模型。大量实验验证了我们的方法在多种方法中表现最优。然而，仍有一些局限性需要在未来的工作中进一步改进。具体来说，我们的方法需要在服务器上构建公共数据集。未来，我们将努力使用生成模型来解决这一问题。

5 总结与展望

5.1 主要工作总结

本文针对资源受限场景下的联邦学习问题，提出了一种基于子模型抽取的解决方案。具体来说，本文提出了两种不同的子模型抽取方法：基于数据分布感知的子模型抽取方法（FedDSE）和基于梯度的子模型抽取方法（FedGSE）。这两种方法分别解决了客户端分布质量不同场景下的资源受限联邦学习问题。

首先，FedDSE 方法通过分析客户端数据分布的异质性，提出了一种基于激活值的子模型抽取策略。该方法在每个训练轮次中，根据模型在客户端本地数据集的激活情况，动态选择激活值较高的神经元组成子模型。通过这种方式，FedDSE 能够有效减少客户端之间的神经元冲突，提升模型训练的效率和准确性。实验结果表明，FedDSE 在多个基准数据集上均取得了显著的性能提升，特别是在高数据异质性场景下表现尤为突

其次，FedGSE 方法从全局模型与子模型的优化方向出发，提出了一种基于梯度的子模型抽取策略。该方法通过在中心侧构建公共数据集，并在全局模型上进行反向传播计算梯度，选择梯度较大的神经元组成子模型。通过这种方式，FedGSE 能够确保子模型的更新方向与全局模型最为接近，从而进一步提升模型的训练效果。实验结果表明，FedGSE 在高数据异质性场景下表现尤为优异，显著优于现有的基线方法。

实验验证通过多个基准数据集和广泛的实验设计，全面评估了 FedDSE 和 FedGSE 两种方法的有效性。在数据集选择上，实验采用了 EMNIST、CIFAR10、CIFAR100 和 TinyImagenet 等经典基准数据集，并在非独立同分布（Non-IID）场景下模拟客户端数据分布的高异质性和低异质性环境，以还原实际联邦学习场景。实验结果表明，在低质量客户端分布的情况下，FedDSE 凭借基于激活值的动态神经元选择机制，有效缓解了客户端之间的数据分布差异对全局模型性能的负面影响，在全部实验数据集上的准确率相比传统方法有了显著的提高。在高质量客户端分布场景中，FedGSE 通过基于梯度选择的子模型抽取策略，在实验中表现出显著优势，其子模型的更新方向与全局模型保持高度一致，使得全局模型的准确率显著提升。准确率在所有数据集均有显著提升通过对全局模型与子模型的梯度差异进行分析，FedGSE 有效抑制了梯度误差的累积，并保证了

模型的优化方向一致。整体实验验证结果充分说明了 FedDSE 和 FedGSE 两种方法在解决资源受限场景下联邦学习问题的实际有效性和优越性能。

5.2 未来工作展望

尽管本文提出的方法在资源受限场景下的联邦学习中取得了显著的进展，但仍有一些问题和挑战需要进一步研究和解决：

（1）生成式模型在联邦学习中的应用：本文的 FedGSE 方法依赖于中心侧的公共数据集来计算梯度。未来的工作可以探索如何利用生成式模型在边缘设备上生成伪数据，从而减少对中心侧公共数据集的依赖，进一步提升联邦学习的隐私保护能力。

（2）模型压缩与加速：尽管本文提出的方法在一定程度上减少了客户端的计算负担，但在实际应用中，模型的规模和复杂度仍然可能成为瓶颈。未来的工作可以探索如何在子模型抽取的基础上，进一步进行模型压缩和加速，以提升联邦学习的整体效率。

（3）在实际场景中，通信成本和计算资源限制是重要的制约因素。未来可研究如何进一步压缩通信开销或在计算资源更有限的设备上应用本文方法。

参考文献

[1] 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.

[2] LIU B, CAI Y, BI H, et al. Beyond Fine-Tuning: Efficient and Effective Fed-Tuning for Mobile/Web Users[C]//DING Y, TANG J, SEQUEDA J F, et al. Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023. ACM, 2023: 2863-2873.

[3] 马彩霞, 贾春福, 蔡智鹏, 等. 面向对抗样本干扰的人脸识别隐私保护方案[J/OL]. 西安电子科技大学学报, 1-11. DOI: 10.19665/j.issn1001-2400.20241105.

[4] 李昆仑, 熊婷. 一种面向复杂场景的人脸识别与目标跟踪算法设计[J/OL]. 现代电子技术, 2024, 47(24): 167-171. DOI: 10.16652/j.issn.1004-373x.2024.24.026.

[5] 刘家森, 王绪安, 余丹, 等. 基于同态加密和边缘计算的关键目标人脸识别方案[J]. 信息安全研究, 2024, 10(11): 1004-1011.

[6] 俞浩, 范菁, 孙伊航. 异构联邦学习在无人系统中的研究综述[J/OL]. 计算机应用研究, 1-10. DOI: 10.19734/j.issn.1001-3695.2024.07.0256.

[7] 王轩慧, 吴颖, 邵凯扬, 等. 基于改进 YOLOv8s 的自动驾驶多目标跟踪检测研究[J/OL]. 汽车技术, 1-7. DOI: 10.19620/j.cnki.1000-3703.20240513.

[8] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need[C]//NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems. Long Beach, California, USA: Curran Associates Inc., 2017: 6000-6010.

[9] 徐浩南, 林立岚, 蔡霞. 基于深度强化学习的多智能体防窃听波束成形[J/OL]. 软件工程, 2024, 27(11): 1-5. DOI: 10.19644/j.cnki.issn2096-1472.2024.011.001.

[10] 万艳丽, 王颖帅, 赵姗姗. 医学大模型研究进展[J]. 医学研究杂志, 2024, 53(10): 1-6+186.

[11] 钟睿康, 李靖华, 林熙明, 等. 深度学习在非小细胞肺癌预后模型中的应用进展[J]. 中国胸心血管外科临床杂志, 2024, 31(09): 1345-1350.

[12] 哈里旦木·阿布都克里木, 冯珂, 史亚庆, 等. 深度学习在骨折诊断中的应用综述[J]. 计算机工程与应用, 2024, 60(05): 47-61.

[13] 姜富伟, 柴百霖, 林奕皓. 深度学习与企业债券信用风险[J]. 计量经济学报, 1-26.

[14] 李文颖, 潘乔, 阎希平. 基于深度学习的金融市场波动率预测模型[J/OL]. 智能计算机与应用, 2024, 14(07): 79-84. DOI: 10.20169/j.issn.2095-2163.240711.

[15] 刘建伟, 刘媛, 罗雄麟. 深度学习研究进展[J]. 计算机应用研究, 2014, 31(07): 1921-1930+1942.

[16] 谭作文, 张连福. 机器学习隐私保护研究综述[J/OL]. 软件学报, 2020, 31(07): 2127-2156. DOI: 10.13328/j.cnki.jos.006052.

[17] 新华社. 中华人民共和国个人信息保护法[EB/OL]. 2021. http://www.mod.gov.cn/gfbw/fgwx/flfg/4892505.html.

[18] REGULATION E D P. GDPR:General Data Protection Regulation[EB/OL]. 2018. https://gdpr-info.eu/.

[19] MCMAHAN B, MOORE E, RAMAGE D, et al. Communication-efficient learning of deep networks from decentralized data[C]//Artificial intelligence and statistics. 2017: 1273-1282.

[20] KARIMIREDDY S P, KALE S, MOHRI M, et al. Scaffold: Stochastic controlled averaging for federated learning[C]//International conference on machine learning. 2020: 5132-5143.

[21] 刘振涛, 李涵, 吴浪, 等. 基于联邦学习的医疗数据共享与隐私保护[J/OL]. 计算机工程与设计, 2024, 45(09): 2577-2583. DOI: 10.16208/j.issn1000-7024.2024.09.003.

[22] 刘育铭, 代煜, 陈公平. 联邦学习在医学图像处理任务中的研究综述[J]. 计算机科学, 1-13.

[23] PFITZNER B, STECKHAN N, ARNRICH B. Federated learning in a medical context: a systematic literature review[J]. ACM Transactions on Internet Technology (TOIT), 2021, 21(2): 1-31.

[24] RIEKE N, HANCOX J, LI W, et al. The future of digital health with federated learning[J]. NPJ digital medicine, 2020, 3(1): 1-7.

[25] 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(1): 12598.

[26] 聂璇, 王殊, 刘渊. 金融领域中的联邦学习技术进展、应用与挑战[J/OL]. 电脑与信息技术, 2024,

32(03): 45-50+85. DOI: 10.19414/j.cnki.1005-1228.2024.03.040.
[27] BYRD D, POLYCHRONIADOU A. Differentially private secure multi-party computation for federated learning in financial applications[C]//Proceedings of the First ACM International Conference on AI in Finance. 2020: 1-9.
[28] IMTEAJ A, AMINI M H. Leveraging asynchronous federated learning to predict customers financial distress[J]. Intelligent Systems with Applications, 2022, 14: 200064.
[29] NEVRATAKI T, ILIADOU A, NTOLKERAS G, et al. A survey on federated learning applications in healthcare, finance, and data privacy/data security[C]//AIP Conference Proceedings: vol. 2909: 1. 2023.
[30] 周传鑫, 孙奕, 汪德刚, 等. 联邦学习研究综述[J]. 网络与信息安全学报, 2021, 7(05): 77-92.
[31] LI T, SAHU A K, ZAHEER M, et al. Federated optimization in heterogeneous networks[J]. Proceedings of Machine learning and systems, 2020, 2: 429-450.
[32] LI Q, DIAO Y, CHEN Q, et al. Federated learning on non-iid data silos: An experimental study[C]//2022 IEEE 38th international conference on data engineering (ICDE). 2022: 965-978.
[33] HUANG X, LI P, LI X. Stochastic controlled averaging for federated learning with communication compression[J]. arXiv preprint arXiv:2308.08165, 2023.
[34] CAI H, GAN C, ZHU L, et al. Tinytl: Reduce memory, not parameters for efficient on-device learning [J]. Advances in Neural Information Processing Systems, 2020, 33: 11285-11297.
[35] LEE S, ISLAM B, LUO Y, et al. Intermittent learning: On-device machine learning on intermittently powered system[J]. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 2019, 3(4): 1-30.
[36] LIU B, CAI Y, ZHANG Z, et al. DistFL: Distribution-aware Federated Learning for Mobile Scenarios [J]. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol., 2021, 5(4): 168:1-168:26.
[37] WANG H, XU W, FAN Y, et al. AOCC-FL: Federated Learning with Aligned Overlapping via Calibrated Compensation[C]//IEEE INFOCOM 2023 - IEEE Conference on Computer Communications, New York City, NY, USA, May 17-20, 2023. IEEE, 2023: 1-10.
[38] GOBIESKI G, LUCIA B, BECKMANN N. Intelligence beyond the edge: Inference on intermittent embedded systems[C]//Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems. 2019: 199-213.
[39] 杨智凯, 刘亚萍, 张硕, 等. 联邦学习通信优化方法综述[J]. 网络与信息安全学报, 1-23.
[40] 郑剑文, 刘波, 林伟伟, 等. 联邦学习通信效率研究综述[J]. 计算机科学, 1-10.
[41] VASWANI A. Attention is all you need[J]. Advances in Neural Information Processing Systems, 2017.
[42] BAI J, BAI S, YANG S, et al. Qwen-vl: A frontier large vision-language model with versatile abilities [J]. arXiv preprint arXiv:2308.12966, 2023.
[43] YANG A, YANG B, HUI B, et al. Qwen2 technical report[J]. arXiv preprint arXiv:2407.10671, 2024.
[44] KAMAL N, GHOSAL P. Three tier architecture for iot driven health monitoring system using raspberry pi[C]//2018 IEEE International Symposium on Smart Electronic Systems (iSES)(Formerly iNiS). 2018: 167-170.
[45] GOU J, YU B, MAYBANK S J, et al. Knowledge distillation: A survey[J]. International Journal of Computer Vision, 2021, 129(6): 1789-1819.
[46] LIN T, KONG L, STICH S U, et al. Ensemble Distillation for Robust Model Fusion in Federated Learning[C/OL]//LAROCHELLE H, RANZATO M, HADSELL R, et al. Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual. 2020. https://proceedings.neurips.cc/paper/2020/hash/18df51b97ccd68128e994804f3eccc87-Abstract.html.
[47] ITAHARA S, NISHIO T, KODA Y, et al. Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data[J]. IEEE Transactions on Mobile Computing, 2021, 22(1): 191-205.
[48] HE C, ANNAVARAM M, AVESTIMEHR S. Group knowledge transfer: Federated learning of large cnns at the edge[J]. Advances in Neural Information Processing Systems, 2020, 33: 14068-14080.
[49] CHO Y J, MANOEL A, JOSHI G, et al. Heterogeneous ensemble knowledge transfer for training large models in federated learning[J]. arXiv preprint arXiv:2204.12703, 2022.
[50] DIAO E, DING J, TAROKH V. HeteroFL: Computation and Communication Efficient Federated

Learning for Heterogeneous Clients[C/OL]//9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net, 2021. https://openreview.net/forum?id=TNkPBBYFkXg.

[51] HORVÁTH S, LASKARIDIS S, ALMEIDA M, et al. FjORD: Fair and Accurate Federated Learning under heterogeneous targets with Ordered Dropout[C/OL]//RANZATO M, BEYGELZIMER A, DAUPHIN Y N, et al. Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual. 2021: 12876-12889. https://proceedings.neurips.cc/paper/2021/hash/6aed000af86a084f9cb0264161e29dd3-Abstract.html.

[52] CALDAS S, KONEČNY J, MCMAHAN H B, et al. Expanding the reach of federated learning by reducing client resource requirements[J]. arXiv preprint arXiv:1812.07210, 2018.

[53] ALAM S, LIU L, YAN M, et al. FedRolex: Model-Heterogeneous Federated Learning with Rolling Sub-Model Extraction[C/OL]//KOYEJO S, MOHAMED S, AGARWAL A, et al. Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022. 2022. http://papers.nips.cc/paper%5C_files/paper/2022/hash/bf5311df07f3efce97471921e6d2f159-Abstract-Conference.html.

[54] KIM M, YU S, KIM S, et al. DepthFL : Depthwise Federated Learning for Heterogeneous Clients [C/OL]//The Eleventh International Conference on Learning Representations. 2023. https://openreview.net/forum?id=pf8RIZTMU58.

[55] LECUN Y, BOSER B, DENKER J S, et al. Backpropagation applied to handwritten zip code recognition[J]. Neural computation, 1989, 1(4): 541-551.

[56] LIN M. Network in network[J]. arXiv preprint arXiv:1312.4400, 2013.

[57] SHARMA S, SHARMA S, ATHAIYA A. Activation functions in neural networks[J]. Towards Data Sci, 2017, 6(12): 310-316.

[58] GOODFELLOW I, WARDE-FARLEY D, MIRZA M, et al. Maxout networks[C]//International conference on machine learning. 2013: 1319-1327.

[59] CLEVERT D A. Fast and accurate deep network learning by exponential linear units (elus)[J]. arXiv preprint arXiv:1511.07289, 2015.

[60] LECUN Y. The MNIST database of handwritten digits[J]. http://yann. lecun. com/exdb/mnist/, 1998.

[61] KRIZHEVSKY A, HINTON G, et al. Learning multiple layers of features from tiny images[J]. 2009.

[62] SHAFIQ M, GU Z. Deep residual learning for image recognition: A survey[J]. Applied Sciences, 2022, 12(18): 8972.

[63] LE Y, YANG X. Tiny imagenet visual recognition challenge[J]. CS 231N, 2015, 7(7): 3.

[64] DIAO E, DING J, TAROKH V. Heterofl: Computation and communication efficient federated learning for heterogeneous clients[J]. arXiv preprint arXiv:2010.01264, 2020.

[65] ZOPH B, CUBUK E D, GHIASI G, et al. Learning data augmentation strategies for object detection [C]//Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVII 16. 2020: 566-583.

[66] PASZKE A, GROSS S, MASSA F, et al. Pytorch: An imperative style, high-performance deep learning library[J]. Advances in neural information processing systems, 2019, 32.

[67] SRIVASTAVA N, HINTON G, KRIZHEVSKY A, et al. Dropout: a simple way to prevent neural networks from overfitting[J]. The journal of machine learning research, 2014, 15(1): 1929-1958.

[68] BJORCK N, GOMES C P, SELMAN B, et al. Understanding Batch Normalization[C/OL]//BENGIO S, WALLACH H, LAROCHELLE H, et al. Advances in Neural Information Processing Systems: vol. 31. Curran Associates, Inc., 2018. https://proceedings.neurips.cc/paper_files/paper/2018/file/36072923bfc3cf47745d704feb489480-Paper.pdf.

[69] ANDREUX M, du TERRAIL J O, BEGUIER C, et al. Siloed federated learning for multi-centric histopathology datasets[C]//Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning: Second MICCAI Workshop, DART 2020, and First MICCAI Workshop, DCL 2020, Held in Conjunction with MICCAI 2020, Lima, Peru, October 4–8, 2020, Proceedings 2. 2020: 129-139.

[70] SELVARAJU R R, COGSWELL M, DAS A, et al. Grad-cam: Visual explanations from deep networks

via gradient-based localization[C]//Proceedings of the IEEE international conference on computer vision. 2017: 618-626.

附录

作者简历