# 联邦学习

Heiko Ludwig, Nathalie Baracaldo

2023 年 4 月 17 日

# 序

机器学习在过去 20 年取得了长足的进步，并在许多应用领域得到了应用。成功的机器学习在很大程度上取决于对高质量数据的获取，包括标记的和未标记的。

与数据隐私、安全和主权有关的担忧引起了公众和技术上的讨论，即如何在符合监管和利益相关者利益的情况下将数据用于机器学习的目的。这些担忧和立法导致人们认识到，在大型中央存储库中收集训练数据可能与维护数据所有者的隐私相矛盾。

虽然分布式学习或模型融合至少从十年前就开始讨论，但联合机器学习（FL）作为一个概念，从 2017 年开始由麦克马汉等人推广。在随后的几年里，人们进行了大量的研究–无论是在学术界还是在工业界–在撰写本书时，第一个可行的联合学习的商业框架正在进入市场。

本书旨在捕捉过去几年的研究进展和技术状况，从该领域的最初构想到首次应用和商业使用。为了获得这种广泛而深入的概述，我们邀请了领先的研究人员来讨论联合学习的不同观点：核心机器学习的观点、隐私和安全、分布式系统和具体的应用领域。

这本书的标题是《联合学习》。方法和应用的全面概述》概述了其范围。它为研究人员和从业人员深入介绍了联合学习的最重要问题和方法。一些章节包含了各种技术内容，与理解算法和范式的复杂性有关，这些算法和范式使得在多个企业环境中部署联合学习成为可能。其他章节则侧重于阐明如何选择隐私和安全解决方案，以适应特定的使用案例，而其他章节则考虑到了联合学习过程将运行的系统的实用性问题。

鉴于该主题固有的跨学科性质，我们在该书的不同章节中遇到了不同的符号惯例。可能是各方在联合机器学习中的各方，在分布式系统的观点中可能被称为客户。在本书的介绍性章节中，我们列出了我们使用的主要术语，每一章都解释了特定学科的术语在引入时如何映射到通用术语，如果是这

样的话。通过这种方法，我们使来自不同背景的读者能够理解本书，同时忠实于所涉及的特定学科的惯例。

从整体上看，这本书使读者能够获得最新研究发展的广泛的最先进的总结。

本书的编辑和一些章节的撰写需要许多人的帮助，我们要感谢他们。IBM 研究院给了我们在这个激动人心的领域工作的机会，不仅在学术上，而且还将这项技术付诸实践，使其成为产品的一部分。我们在这一路上学到了宝贵的经验，我们要感谢我们在 IBM 的同事。特别是，我们要感谢我们的主任 Sandeep Gopisetty，他给了我们空间来创作这本书。Gegi Thomas，他确保了我们的研究贡献能够进入产品：还有我们的团队成员。

各章作者提供了本书的内容，并对我们提出的修改其章节的要求很有耐心。

我们最应该感谢的是我们的家人，在撰写和编辑本书的一年多时间里，他们耐心地忍受了我们把时间投入到书中而不是他们。海科深深感谢他的妻子比阿特丽斯-拉吉奥（Beatriz Raggio），感谢她做出这些牺牲并自始至终支持他。娜塔莉深深地感谢她的丈夫和儿子圣地亚哥和马蒂亚斯-博克，感谢他们的爱和支持，感谢他们为她所有的项目，包括这本书加油。她还感谢她的父母 Adriana 和 Jesus；如果没有他们惊人和持续的支持，就不可能取得这样和更多的成就。

美国加利福尼亚州圣何塞市　　Heiko Ludwig

2021 年 9 月　　Nathalie Baracaldo

# 目录

**第一部分　联邦学习视为一个机器学习问题** **1**

**第一章　联邦学习中的个性化** **3**

1.1 导言 . . . . . 3
1.2 迈向个性化的第一步 . . . . . 4
1.2.1 微调全局模型实现个性化 . . . . . 5
1.2.2 作为一阶元学习方法的联合平均法 . . . . . 5
1.3 个性化策略 . . . . . 6
1.3.1 客户（方）集群 . . . . . 7
1.3.2 客户背景介绍 . . . . . 8
1.3.3 数据增强 . . . . . 9
1.3.4 蒸馏 . . . . . 10
1.3.5 元学习方法 . . . . . 11
1.3.6 模型的混合 . . . . . 12
1.3.7 模型正则化 . . . . . 14
1.3.8 多任务学习 . . . . . 16
1.4 个性化技术的基准 . . . . . 17
1.4.1 合成的联合数据集 . . . . . 17
1.4.2 模拟联合数据集 . . . . . 18
1.4.3 公共联合数据集 . . . . . 18
1.5 个性化是附带的参数问题 . . . . . 19
1.6 总结 . . . . . 21

**第二章　通信高效的分布式优化算法** **23**

2.1 引言 . . . . . 23

2.2 Local-Update SGD 和 FedAvg . . . . . 25
2.2.1 Local-Update SGD 及其变体 . . . . . 26
2.3 模型压缩 . . . . . 30
2.3.1 有压缩更新的 SGD . . . . . 30
2.3.2 自适应压缩率 . . . . . 34
2.3.3 模型剪枝 . . . . . 35
2.4 讨论 . . . . . 36
参考文献 . . . . . 36

**第三章 高效通信模型融合 41**
3.1 导言 . . . . . 41
3.2 模型的互换-不变结构 . . . . . 43
3.2.1 匹配平均法的一般公式 . . . . . 44

**第四章 分离学习：分布式深度学习的一种资源节约型模型和数据并行方法 45**
4.1 分离学习的介绍 . . . . . 45
4.1.1 普通的分离学习 . . . . . 46
4.2 通信效率 [15] . . . . . 47
4.3 延迟 . . . . . 48
4.4 分离学习的拓扑结构 . . . . . 48
4.4.1 多样化的配置 . . . . . 48
4.4.2 用 ExpertMatcher 选择模型 [19] . . . . . 49
4.4.3 实现细节 . . . . . 50
4.5 分离学习的协作推理 . . . . . 51
4.5.1 防止协作推理中的重构攻击 . . . . . 52
4.5.2 激活共享的差异性隐私 . . . . . 54
4.6 未来工作 . . . . . 54

# 第一部分

# 联邦学习视为一个机器学习问题

# 第一章　联邦学习中的个性化


## 摘要

典型的联邦学习（FL) 问题的表述需要学习一个适合所有各方的单一模型，同时禁止各方与聚合者分享他们的数据。然而，可能不可能学习一个适合所有各方的共同的单一模型。例如，考虑一个句子完成问题：” 我住在..... 的州”。答案显然取决于当事人，这里没有一个单一的模型是合适的。为了处理这种情况，最近的文献中提出了各种人格化策略。特别是，这个问题似乎与元学习有密切联系。我们回顾了最近的 FL 个性化技术，将其分为八组，并总结了三种策略和相应的数据集，以作为联合学习中个性化的基准。我们对联合学习中个性化的统计挑战进行了概述。在高层次上，个性化导致了模型复杂性的增加，这反过来又增加了联合学习任务的难度。我们研究了什么时候过多的个性化会阻碍个性化联合学习的标准方法学习各方的共同部分，并提出了克服此类问题的替代方法。


## 1.1　导言

集中式联合学习旨在从各方的数据中学习一个全局模型，同时保持他们的本地数据的私密性和对其个人机器的本地化。这个全局模型的优点是利用了所有各方的数据，因此对各方的测试数据具有更好的概括性。然而，在实际场景中，个别当事方的数据集往往是异质的（非 IID），从而使一个全局模型的性能对某些当事方来说是次优的。另一方面，如果每一方都在他们的本地数据上训练一个本地模型，他们会在类似于测试时预期的数据分布上进行训练，但由于本地一方可用数据的匮乏，可能无法进行泛化。个性化联合学习的目的是学习一个具有全局模型的泛化能力的模型，但也能在每一方的特定数据分布上表现良好。

为了说明个性化的需要，考虑在联合学习环境中学习的语言模型的情况 [10]：如果我们使用一个全局模型来预测提示的下一个词：” 我住在..... 州”，全局模型将为每一方预测相同的标记（州名），而不考虑他们的本地数据分布。因此，虽然全局模型可能能够很好地学习语言的一般语义，但它却不能对个别当事人进行个性化处理。

除了上述定性的例子外，我们还可以从数量上证明个性化的必要性。我们使用 MNIST 数据集建立了我们的实验，该数据集被划分为 100 方。我们使用具有不同浓度参数（）的 Dirichlet 分布 [64]，以异质的方式在这些当事人之间分配数据。我们在两种情况下训练一个 2 层全连接网络，以衡量各方从参与联合学习中获得的好处。在第一种情况下，我们为 100 个党派中的每一个人训练一个单独的网络，只使用党派自己的数据训练 10 个历时，并测量这些单独网络在各自党派的测试数据（$\mathrm{Acc}_i^{\mathrm{local}}$）上的性能。在第二种情况下，所有 100 个党派都参与到使用联邦平均法（FedAvg）[39] 训练一个全局模型，进行 100 个通信回合，我们测量这个全局模型在每个党派的测试数据上的性能（$\mathrm{Acc}_i^{\mathrm{global}}$）。图 4.1 显示了在不同的数据异质性水平下，每一方的全局模型和局部模型（$\mathrm{Acc}_i^{\mathrm{global}} - \mathrm{Acc}_i^{\mathrm{local}}$）的性能差异柱状图。从图中可以看出，全局模型对参与其训练的每一方都没有好处，当数据的非 IID 特征更严重（$\alpha$ 值更小）时，这种现象就更明显了。这个实验强调了在各方的本地数据分布上对全局模型进行个性化处理的必要性，以确保每一方都能从其参与的学习设置中获益。

在这一章中，我们回顾了在联合学习文献中提出的不同的个性化技术，并讨论了联合学习和一阶元学习 [18] 之间的联系。我们还研究了联合学习中个性化的统计限制。特别是，我们表明个性化在一定程度上提高了特定方的性能。在这一点之后，在问题中增加更多的当事方并不能导致性能的改善。

## 1.2 迈向个性化的第一步

在这一节中，我们将考察一种结合了联合学习和个性化的基本技术，并探讨为什么这种技术是个性化任务的一个强有力的基线。

### 1.2.1 微调全局模型实现个性化

使用联合学习学到的全局模型进行个性化的一个直接方法是在本地数据上进一步训练它。这种方法允许我们通过对全局模型进行本地更新的数量来控制通过对全局模型进行本地更新的数量来控制个性化程度–零本地更新可以保留全局模型，而随着本地更新数量的增加，模型对本地数据变得更加个性化。

虽然这种技术可能看起来很简单，但它是人称化任务的一个强有力的基线。我们在第 4 节介绍的实验中研究了这种微调方法。4.1 和图 4.1。我们通过在本地数据上进行 1 次微调，将在 100 个当事人身上学到的全局模型个性化，然后在当事人的本地测试数据上测量这个微调模型的性能。通过这个实验的结果可以看出，与本地模型相比，这种简单的微调技术大大地提高了全局模型的性能。对于异质性的极端情况，这种方法提高了相当多的当事方的性能，也没有对异质性不太严重的情况下的性能产生负面影响。我们现在旨在了解这种微调方法的强大性能背后的原因。

### 1.2.2 作为一阶元学习方法的联合平均法

在这一节中，我们试图理解使用联合平均法学习的全局模型进行微调的有效性背后的原因。我们复制了 Jiang 等人 [29] 的推导，表明联合平均法的更新是联合 SGD 更新和一阶 MAML（FOMAML）更新的组合。

什么是元学习和 MAML？传统的机器学习方法旨在学习在给定任务上表现最好的参数，而元学习或学习学习 [55- 57, 59] 旨在学习可以快速适应新任务的参数。模型不可知元学习（MAML）[18] 是最流行的元学习方法之一：它的目标是找到可以在少数梯度更新中适应新任务的模型参数。然而，为了实现这一目标，MAML 的目标需要计算二阶导数，这在计算上是很昂贵的。一阶 MAML（FOMAML）[43] 通过只考虑一阶导数来接近 MAML 的目标，从而减少 MAML 的计算需求。参见第 4.3.5 节。4.3.5 节进一步讨论了元学习和相关的联合学习个性化策略。

我们通过定义 FedSGD 的更新来开始分析（公式（1.1））。FedSGD 的操作方式是对 $N$ 个缔约方中的每一个采取单一的梯度步骤，将这些梯度传回给聚合器，然后聚合这些梯度来更新全局模型。我们用 $\nabla_k^i$ 来表示第 $i$ 方

的第 $k$ 步梯度。

$$\nabla_{\text{FedSGD}} = \frac{1}{N}\sum_{i=1}^{N}\frac{\partial \mathcal{L}_i(\theta)}{\partial \theta} = \frac{1}{N}\sum_{i=1}^{N}\nabla_1^i \tag{1.1}$$

其中 $\theta$ 为模型参数（如神经网络权重），$\mathcal{L}_i(\theta)$ 为 $i$ 方的损失。接下来，我们以类似的方式推导出 MAML 和一阶 MAML[18] 的更新。假设 $\theta_K^i$ 是第 $i$ 方的个性化模型，是在以 $\beta$ 为学习率的损失梯度上走了 $K$ 步后得到的：

$$\theta_K^i = \theta - \beta\sum_{j=1}^{K}\frac{\partial \mathcal{L}_i(\theta_j^i)}{\partial \theta} \tag{1.2}$$

然后，MAML 更新被定义为个性化模型 $\theta_K^i$ 相对于初始参数 $\theta$ 的梯度，在 $N$ 个当事方中平均。不幸的是，这种计算需要高阶导数，即使对 $K=1$ 来说也很昂贵。FOMAML 忽略了高阶导数，只使用一阶梯度：

$$\nabla_{\text{FOMAML}}(K) = \frac{1}{N}\sum_{i=1}^{N}\frac{\partial \mathcal{L}_i(\theta_K^i)}{\partial \theta} = \frac{1}{N}\sum_{i=1}^{N}\nabla_{k+1}^i \tag{1.3}$$

在计算了 FedSGD 和 FOMAML 的更新后，我们现在来看看 Federated Averaging（FedAvg）的更新。FedAvg 的更新是各方更新的平均值，是局部梯度更新 $\nabla_j^i$ 的总和：

重新排列这些条款，我们可以得出 FedAvg、FedSGD 和 FOMAML 的更新之间的关系：

$$\nabla_{\text{FedAvg}} = \nabla_{\text{FedSGD}} + \sum_{j=1}^{K-1}\nabla_{\text{FOMAML}}(j) \tag{1.4}$$

在每次通信之前，FedAvg 中 1 个梯度更新（$K=1$）的 FedAvg 更新根据公式（1.4）减少到 FedSGD 设置。增加梯度更新的数量会逐步增加更新中的 FOMAML 部分。根据 Jiang 等人 [29] 的研究，用 $K=1$ 训练的模型很难个性化，而增加 $K$ 可以增加模型的个性化能力，直到某一点，超过这一点，初始模型的性能就变得不稳定。

## 1.3 个性化策略

近年来，联合学习环境下的个人化在研究界获得了相当大的兴趣。在这一节中，我们研究了针对这一问题提出的各种技术，并将它们分为 8 大类。

表 4.1 中总结了分类标准以及属于各自标准的方法。在下面的小节中，我们将深入研究表中定义的每个标准，并研究其优势和劣势。

### 1.3.1 客户（方）集群

联合学习中个性化的核心前提是，由于各方数据的非 IID 异质性分布，一个全局模型可能无法适用于所有各方。用于个性化的客户（当事方）聚类技术在一个共同的假设下运作：在系统中存在的 $N$ 个当事方中，有 $K < N$ 的不同数据分布。这个假设使这些技术能够将各方聚成 $K$ 个集群，以缓解非 IID 数据分布条件，并为 $K$ 个集群中的每一个学习一个共同的全局模型。因此，在这种表述下，个性化问题又被细分为两个子问题：（1）定义一个聚类假设，将各方聚在一起；（2）为每个定义的聚类汇总并学习一个模型。

集群联合学习（CFL）[47] 假设存在一个分区 $\mathcal{C} = \{c_1, \cdots, c_K\}, \cup_{k=1}^{K} c_k = \{1, \cdots, N\}$，这样，每一个缔约方的子集 $c_k \in \mathcal{C}$ 都满足传统的联合学习假设，即全局模型同时对所有缔约方的数据分布进行风险最小化。然而，CFL 并不是一次就能识别出完整的当事方聚类 $\mathcal{C}$，而是递归地将当事方双分到聚类中，直到所有的聚类都被识别出来。该算法通过训练局部模型进行，直到收敛到一定限度。然后，这些单独的当事方模型被汇总，并检查全局模型的一致性，即全局模型在多大程度上使每一方的风险最小。如果全局模型符合各方的某个停止标准，CFL 就会终止。否则，各方被划分为两个子群，CFL 在每个子群上被递归执行。由于双分区方法是递归地工作的、集群的数量 $K$ 不需要事先知道。此外，由于聚类机制是在聚合器上实现的，各方并不承担该方法的计算负担。相反，聚合器凭借其通常比各方更强的计算能力，可以减少聚类的开销。

3S-聚类 [19] 也制定了与 CFL[47] 类似的问题，但它的目的不是递归的双分区方，而是在聚合器上一次性找到 $K$ 个集群。一旦本地模型被训练好并传达给聚合器、3S-聚类执行一种聚类方法–通常情况下，KMeans 可行，但也可以采用其他的聚类方法，如原始论文中所示，他们研究这种方法主要是为了在聚合器上进行 byzantine-robust 分布式优化，以寻找 $K$ 个聚类。然而，这种方法仅限于凸目标，因此不适用于非凸目标，如深度神经网络的情况。

前面提到的两种方法使用聚合器进行聚会聚类，而这一类的另外两种方法，IFCA[20] 和 HypCluster[36] 则利用聚会来确定自己的聚类成员资格（图 4.2）。这两种方法彼此相当相似，通过聚合器维护 K 个集群中心和相关

的模型参数来操作。在每一轮，聚合器将集群参数广播给每一方，而每一方则通过选择实现最低损失值的参数来估计其集群身份。然后，这些集群中心被用作本地模型的初始化器，在本地数据上进行微调，并与集群身份一起发回给聚合器进行聚合。聚合器然后根据模型的集群成员资格聚合模型，整个过程重复进行。

### 1.3.2 客户背景介绍

学习用户特定的情境特征或嵌入已被广泛用于改善与联合学习无关的问题中模型的个性化 [1, 22, 27, 38, 58]。客户端情境化利用了学习用户嵌入的相同方法来完成联合学习中的个性化任务。这种方法背后的原理是，每一方的嵌入捕捉了特定一方的特征，并作为全局模型的指标，利用这种背景来调整它对特定一方的预测。

联合协作过滤（FCF）[2] 提出了一个基于协作过滤 [48] 的推荐系统，以联合的方式学习。协同过滤通过用户-物品交互矩阵 $\boldsymbol{R} \in \mathbb{R}^{N \times M}$ 作为用户因素矩阵 $\boldsymbol{X} \in \mathbb{R}^{K \times N}$ 和物品因素矩阵 $\boldsymbol{Y} \in \mathbb{R}^{K \times M}$ 的线性组合来模拟 $N$ 个用户和 $M$ 个物品之间的交互。

$$\boldsymbol{R} = \boldsymbol{X}^T \boldsymbol{Y} \tag{1.5}$$

在 FCF 算法的每个迭代中，聚合器将物品因素矩阵 $\boldsymbol{Y}$ 发送给每一方，而每一方又使用他们的本地数据来更新用户因素矩阵 $\boldsymbol{X}$ 和物品因素矩阵 $\boldsymbol{Y}$。更新的物品因素矩阵被送回聚合器进行聚合，而用户因素矩阵在每一方都是保密的。这允许每一方学习自己的用户因素矩阵集，同时利用各方的项目因素信息。在比较 FCF 和标准协同过滤的实验中，FCF 被证明在多个推荐性能指标和多个推荐数据集上与标准协同过滤的性能接近。

虽然 FCF 是专门针对协同过滤的，但 FURL[5] 通过 (a) 定义私有和联合参数以及 (b) 指定独立的本地训练约束和独立的聚合约束来概括这种方法。独立的本地训练约束规定，本地各方使用的损失函数与其他各方的私有参数无关，而独立的聚合约束规定，全局聚合步骤与各方的私有参数无关。当这些条件得到满足时，FURL 保证了将参数分成私有参数和联合参数不会造成模型质量损失。

图 4.3a 显示了 FURL 在文档分类任务中的应用。这里，用户嵌入对每一方都是私有的，而 BiLSTM 和 MLP 的参数是所有用户共享的联合参数。每一方联合训练私有参数和联合参数，但只与聚合器分享联合参数以进行

聚合。在实验中，通过 FURL 实现的个性化被证明可以显著提高文档分类任务的性能。

然而，通过 FURL 实现个性化，有几个缺点。首先，使用 FURL 需要将私人参数纳入建模，这可能需要对网络结构进行修改。随后，将私人参数纳入模型增加了需要学习的参数数量，鉴于当事方数据的匮乏，这可能会使任务更加困难。最后，FURL 技术对于新的政党来说存在一个冷启动问题。由于私有参数是针对每一方的，新加入框架的各方需要首先训练他们的私有参数，然后才能利用个性化的力量，在此之前可能会遭受性能下降的影响。

### 1.3.3 数据增强

数据增强技术已被用于标准的机器学习问题，以缓解类不平衡、非 IID 数据集的问题，或人为地增加其他较低大小的数据集。这些技术包括从过度采样代表性不足的类样本 [8] 到训练 GANs 来生成增强的数据样本 [37]。对这一领域感兴趣的读者应该参考关于数据增强技术的调查，以了解这一领域的情况 [40, 52]。

由于联合学习也受到各方数据匮乏的影响，而全球范围内有大量的数据，因此很自然地会问，全球数据（跨越所有各方的数据）是否可以用来提高某一方的性能。本着同样的精神，人们提出了一些方法，要么在全球范围内共享少量的数据，以帮助提高各方的性能 [66]，要么在本地模型之外训练生成对抗网络（GAN）以增强数据样本 [28]。

一种直接的增强数据的方法是收集所有各方的数据子集，创建一个全球共享的数据集，每一方都可以用它来增强他们的本地数据集。DAPPER[36] 和全球数据共享 [66] 方法就属于这个类别。这两种方法都利用一个表明全球数据分布的全局数据集 DG。全局数据共享方法建议通过共享一个在全局数据集上训练的热身模型，以及数据集的一个随机子集（ DG）给每一方来初始化联合学习过程。每一方用聚合器提供的数据集增加其本地数据集，以训练其本地模型，然后将其传送回聚合器进行聚合。图 4.3b 显示了这个过程的说明。

另一方面，DAPPER[36] 不是直接用全局数据集来增加本地数据集，而是优化目标：

$$\lambda D_{\text{party}} + (1 - \lambda) D_G \tag{1.6}$$

每一方在每个优化步骤中，都以 $\lambda$ 的概率选择本地数据集 $D_{\text{party}}$，以

$(1-\lambda)$ 的概率选择全局数据集 $D_G$ 进行优化。其余的优化和聚合步骤保持不变。

DAPPER 和全球数据共享方法都显示出比没有个性化训练的模型有明显的改进，但它们需要将各方的数据转移到全球聚合器和其他各方。将当事人的数据转移到他们的机器之外，违反了联合学习的隐私保证，因此这些方法在实践中可能无法直接实现。

XorMixup[51] 旨在规避与转环方数据有关的隐私问题，同时仍然利用数据增强的个性化能力。它建议使用一个 XOR 编码方案来混淆上传到聚合器的数据样本。实施 XorMixup 的每一方首先从两个不同的类别标签中选择数据样本，然后使用两者的 XOR 创建一个编码的数据样本。每一方都将这种编码样本上传到聚合器，聚合器使用来自指定类标签的单独数据样本对其进行解码，并使用这些解码样本来训练模型。这些编码样本被证明与原始数据样本有很高的不相似性，而且在非 IID 条件下，模型的性能也有改善。

### 1.3.4 蒸馏

在联合学习的一般表述下，当聚合器将一个模型发送给一方时，它将该模型作为起点，在本地数据上进行训练。通过蒸馏实现的个性化采取了不同的方法来解决这个问题。基于蒸馏的方法不是使用中心模型参数作为起点，而是使用知识蒸馏 [21,24] 在其他模型之间转移知识，而不明确复制参数。利用蒸馏而不是复制模型参数的一个关键优势是，模型架构不需要在各方和聚合器之间相同，框架就能发挥作用。例如，各方可以选择更适合他们的数据的模型架构和硬件约束。联合相互学习（FML）[50] 和 FedMD[34] 是遵循这种方法的两个主要方法。

什么是知识蒸馏？模型压缩 [11] 是指减少模型大小的任务，从而减少存储模型所需的内存，提高推理速度，同时保留原始神经网络中的信息。知识提炼 [21, 24] 是一种模型压缩技术，旨在将信息或知识从一个较大的网络有效地转移到一个较小的网络。在知识提炼中，有 3 个主要组成部分–教师网络、学生网络和知识。教师网络是编码知识的较大的模型，这些知识需要转移到通常规模较小的学生网络中。有几种方法可以定义要提炼的知识[21]–它可以是网络中某些层的输出（如基于反应和特征的知识），也可以是不同层或数据样本之间的关系（如基于关系的知识）。

FML 在本地模型和全局模型之间采用双向提炼的方式。实施 FML 的各方维护一个本地模型，该模型在他们的数据上不断训练，而不与聚合器共

享数据。在每一轮通信中，聚合器向每一方发送全局模型，该模型由该方通过全局和局部模型之间的双向知识蒸馏进行更新。相应的目标函数如下：

$$\mathcal{L}_{\text{local}} = \alpha\mathcal{L}_{\text{local}} + (1-\alpha)D_{KL}(p_{\text{global}}||p_{\text{local}}) \tag{1.7}$$

$$\mathcal{L}_{\text{global}} = \beta\mathcal{L}_{\text{global}} + (1-\beta)D_{KL}(p_{\text{local}}||p_{\text{global}}) \tag{1.8}$$

由于 FML 中局部和全局模型之间的连接是通过输出概率的 KL 发散，而不像其他联合学习方法中通过参数复制，所以局部和全局模型的架构可以是不同的。FML 的原始工作 [50] 进行了实验来证明这种效果。在不同的一方使用不同的网络架构，作者显示在这种设置下，比独立训练一个全局模型也有改善。

FedMD[34] 也提出了一个类似于 FML 的使用 distilla- tion 的个性化表述。FedMD 框架需要一个公共数据集，该数据集在各方和聚合器之间共享，同时还有各方维护的私人数据集。该框架通过各方首先在公共数据集上训练模型，然后在各自的私有数据集上进行训练，然后将公共数据集中每个样本的等级分数传达给中央聚合器。所有各方的这些等级分数的汇总被用作目标分布，每一方使用蒸馏法学习。与 FML 类似，FedMD 具有支持各方不同模型架构的优势。然而，FedMD 需要一个大型的公共数据集，需要在各方之间共享，从而增加了各方和聚合器之间的通信成本。

### 1.3.5 元学习方法

当代机器学习模型被训练成在单一任务上表现良好。另一方面，元学习或" 学会学习"[25, 57, 59] 的目的是学习可以快速适应新任务的模型，只需要少量的例子。有多种方法可以实现这一点–基于指标、基于模型和基于优化的方法 [59]。在本节中，我们重点讨论基于优化的方法，它更适合我们的目的。基于优化的元学习技术旨在学习模型参数，这些参数可以在给定的少量例子和几个梯度更新内快速修改以适应新任务。模型不可知元学习（MAML）[18] 是一种相当流行的方法，适用于任何可以用基于梯度的方法学习的模型。MAML 不是训练模型参数以最小化给定任务上的损失，而是训练模型参数以最小化几个参数适应步骤后的任务上的损失。如果我们把每个任务看作是联合学习设置中的一方，我们可以在个性化的联合学习和元学习之间得出一个平行的结论。我们希望训练一个全局模型，作为聚会模型的一个很好的初始化器，使其能够快速适应，即个性化，以适应聚会数据的分布。在第 4.2 节中，我们回顾了在全球范围内的联系。4.2 节中，我们

回顾了天真的个性化基线（即 FedAvg 的微调）与元学习之间的联系。现在我们回顾一下最近的其他基于元学习的方法。

ARUBA[31] 是一个将元学习与多任务学习技术相结合的框架，使元学习方法能够学习并利用任务的相似性来提高其性能。ARUBA 背后的动机之一是，在元学习模型中，某些模型的权重作为特征提取器，不需要太多修改就可以在不同的任务中转移，而其他权重则变化很大。有了每个坐标的学习率，就可以根据参数在不同任务中的可转移性，以不同的速度进行调整。ARUBA 在联合学习环境下对下一个字符预测任务进行测试时，与微调的 FedAvg 基线的性能相匹配，但没有对微调学习率进行额外的超参数优化。

FedMeta[9]–与 ARUBA 同时提出–将标准的元学习算法纳入联合学习设置。在这种设置下，聚合器的目的是保持一个初始化，使一方能够迅速适应其本地数据分布。当事人通过在本地执行元学习算法的内循环（支持数据的适应步骤）进行训练，并将外循环的梯度（查询数据）返回给聚合器，聚合器使用这些信息来更新其初始化。虽然 FedMeta 通过在当事人身上运行元学习步骤，同时在聚合器上聚合模型初始化，将元学习纳入了个性化，但 Per-FedAvg[17] 显示，这种表述在某些情况下可能表现不佳。相反，Per-FedAvg 假设每一方将全局模型作为初始化，并就其自身的损失函数更新一个梯度步骤，将问题表述改变为如下（4.11）：

$$\min_{\theta} := \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_i(\theta - \alpha \nabla \mathcal{L}_i(\theta)) \tag{1.9}$$

最后，FedPer[3] 提出将全局模型分离成一个作为特征提取器的基础网络和一个个性化层。各方共同训练基础层和个性化层，但只与聚合器共享基础网络进行聚合。这允许系统使用来自多方的信息学习表征提取网络，同时学习针对每一方的个性化层。原文中没有明确探讨这种方法和元学习之间的联系，元学习文献中也有相关的工作，几乎没有内循环（ANIL）[45]，它提出将网络分离成主体和头部网络，在元学习的内循环中只让头部适应新任务。

### 1.3.6 模型的混合

在联合学习的标准表述中，本地模型是在本地数据上训练的，而全局模型则是将各方的信息汇总起来，建立一个全局模型。一方参与联合学习的主要动机是利用其他方的信息，与在本地训练模型相比，减少其概括误差。然

而，在某些情况下，全局模型对联合学习系统中的某些当事方的表现比这些当事方在本地训练的单个模型要差 [62]，例如，见图 4.1 的实验。这就促使了通过学习一个参数来混合全局和局部模型的想法，以最佳方式结合这两个模型。

FL+DE[44] 使用专家混合技术 [63] 学习结合本地和全局模型的预测类别概率。每一方都保持一个在本地数据上训练的本地领域专家（DE）模型，同时也与其他各方合作建立一个全局模型。门控函数（$\alpha_i(x)$）–在最初的工作中被参数化为逻辑回归模型 [44]–与联合学习设置一起学习，以最佳方式结合全局模型（$\hat{y}_G$）和本地领域专家（$\hat{y}_i$）的预测类概率。因此，门控函数在输入的条件下，学习两个模型之间的偏好区域。对一个给定的数据样本 $x$ 的最终预测是两个模型的预测类别概率的凸组合：

$$\hat{y}_i = \alpha_i(x)\hat{y}_G(x) + (1-\alpha_i(x))\hat{y}_{\text{local}}(x) \tag{1.10}$$

$$\alpha_i(x) = \sigma(w_i^T x + b_i) \tag{1.11}$$

MAPPER[36] 和 APFL[15] 方法没有使用专家混合技术来组合输出概率，而是学习一个混合参数（$\alpha$）来优化组合局部和全局模型。在 APFL 中，虽然全局模型仍然像传统的联合学习那样被训练成在集合域上的经验风险最小化，但本地模型（$h_{\text{local}}$）被训练成也使用 $\alpha$ 来纳入全局模型（$h_g$）的一部分（公式（4.14））。第 $i$ 方的个性化模型是全局模型（$h_g$）和局部模型（$h_{\text{local}}$）的凸组合（公式（1.13））。

$$h_{\text{local}} = \arg\min_h \hat{\mathcal{L}}_{D_i}(\alpha_i h + (1-\alpha_i)h_g) \tag{1.12}$$

$$h_{\alpha_i} = \alpha_i h_{\text{local}} + (1-\alpha_i)h_g \tag{1.13}$$

到目前为止，我们所研究的三种方法都保持了局部和全局模型。然而，这些模型都是为同一任务而训练的。LG-FedAvg[35] 则提议将学习任务在本地和全局模型之间进行分叉–每一方都学习从原始数据中提取高级表征，而全局模型则在所有设备的这些表征（而不是原始数据）上操作。我们在图 4.4 中描述了一个图像分类任务的 LG-FedAvg 过程。在这里，局部模型被训练成从原始图像中提取高级表征，虽然全局模型是使用监督学习训练的，但局部模型可以自由选择技术来学习这些表征。如图 4.4 所示，可以使用监督预测任务（使用辅助模型将表征映射到预测）或无监督或半监督技术（子图（a）至（c））来学习这些表征。本地模型也可以通过针对受保护属性的对抗性训练来学习公平表征 (子图 d)。这种分叉有几个好处：(a) 在表征而

不是原始数据上操作全局模型，减少了全局模型的大小，从而减少了需要在聚合器和各方之间交流的参数和更新的数量，(b) 它允许本地各方根据其本地数据集的特点选择专门的编码器来提取表征，而不是使用一个通用的全局模型，(c) 它允许本地模型学习混淆受保护属性的公平表征，从而增强本地数据的隐私性。

### 1.3.7 模型正则化

在传统的监督式联合学习中，系统被优化为：

$$\min_{\theta}\left\{\mathcal{L}(\theta):=\frac{1}{N}\sum_{i=1}^{N}\mathcal{L}_i(\theta)\right\} \tag{1.14}$$

这里，$N$ 是当事方的数量，$\theta$ 是模型参数，$\mathcal{L}_i(\theta)$ 表示第 $i$ 方数据分布的损失。

另一方面，基于规则化的个性化技术对标准目标的规则化版本进行优化。Loopless Local Gradient Descent（L2GD）[23]、Federated Attentive Message Passing（FedAMP）[26]、pFedME[16] 和 Fed+[61] 都是正则化技术的实例，主要区别在于它们对正则化目标的定义。

L2GD[23] 将正则化目标定义为局部模型参数（$\theta_i$）与各方平均参数（$\bar{\theta}$）之差的 L2 准则，整个系统优化方程（1.15）和（1.16）中定义的目标。为了优化这一目标，L2GD 提出了一种非均匀 SGD 方法，并对各方与聚合器之间所需的通信回合数进行收敛分析。该方法将目标视为一个 2 和问题，对 $\nabla\mathcal{L}$ 或 $\nabla\psi$ 进行采样以估计 $\nabla F$，并定义了一个梯度的无偏估计器，如公式（1.17）。在每个时间步骤中，要么局部模型以 $1-p$ 的概率采取局部梯度步骤，要么聚集器以 $p$ 的概率将局部模型向平均方向移动。

$$\text{L2GD}:\min_{\theta_1,\cdots,\theta_N}\{F(w):=\mathcal{L}(\theta)+\lambda\psi(\theta)\} \tag{1.15}$$

$$\mathcal{L}(\theta):=\frac{1}{N}\sum_{i=1}^{N}\mathcal{L}_i(\theta_i),\psi(\theta):=\frac{1}{2N}\sum_{i=1}^{N}\|\theta_i-\bar{\theta}\|^2 \tag{1.16}$$

$$G(\theta):=\begin{cases}\frac{\nabla\mathcal{L}(\theta)}{1-p} & \text{以}1-p\text{ 的概率}\\ \frac{\lambda\nabla\psi(\theta)}{p} & \text{以}p\text{ 的概率}\end{cases} \tag{1.17}$$

L2GD 适用于凸型损失函数并提供保证，因此不能直接适用于神经网络模型中通常遇到的非凸型损失函数。

pFedMe[16] 通过定义如下的目标，将个性化问题建模为一个双级优化问题：

$$\text{pFedMe} : \min_{\theta} \left\{ F(\theta) = \frac{1}{N} \sum_{i=1}^{N} F_i(\theta) \right\} \tag{1.18}$$

$$F_i(\theta) = \min_{\theta_i} \left\{ \mathcal{L}_i(\theta_i) + \frac{\lambda}{2} \|\theta_i - \theta\|^2 \right\} \tag{1.19}$$

这里，$\theta_i$ 是第 $i$ 方的个性化模型，根据其本地数据分布进行训练，同时与内部层面的全局模型参数 $\theta$ 保持一定的距离。然后，一方的最佳个性化模型被定义为

$$\hat{\theta}_i(\theta) := \text{prox}_{\mathcal{L}_i/\lambda}(\theta) = \arg\min_{\theta_i} \left\{ \mathcal{L}_i(\theta_i) + \frac{\lambda}{2} \|\theta_i - \theta\|^2 \right\} \tag{1.20}$$

与 FedAvg 类似，实施 pFedMe 的系统在每一轮通信中向各方发送全局模型权重，并在一定数量的局部回合后利用各方返回的权重进行模型聚合。与 FedAvg 不同，当事人在本地最小化方程（1.19），这是一个双级优化问题。在每个局部回合，党首先解决方程（1.20），以找到最佳的个性化党的参数 $\hat{\theta}_i(\theta_{i,r}^t)$。这里，$\theta_{i,r}^t$ 是 $i$ 方在全局回合 $t$ 和局部回合 $r$ 的局部模型，其中 $\theta_{i,0}^t = \theta^t$。此后，在外部层面上，一方利用梯度更新本地模型 $\theta_{i,r}^t$，相对于 $F_i$ 方程（1.19）。

FedAMP[26] 为个性化提出了以下目标：

$$\min_{\theta} \left\{ \sum_{i=1}^{N} \mathcal{L}_i(\theta_i) + \lambda \sum_{i<j}^{N} A(\|\theta_i - \theta_j\|^2) \right\} \tag{1.21}$$

目标的第二部分定义了一个注意力诱导函数 $A(\|\theta_i - \theta_j\|^2)$，它以非线性的方式衡量各方参数之间的相似性，旨在改善各方之间的合作。注意力诱导函数可以采取任何形式；然而，在拟议的工作中，作者使用了负指数函数 $A(\|\theta_i - \theta_j\|^2) = 1 - e^{-\|\theta_i - \theta_j\|^2/\sigma}$。为了优化目标，FedAMP 采用了一种交替优化策略，首先通过从各方收集的权重在聚合器上优化 $\sum_{i<j}^{N} A(\|\theta_i - \theta_j\|^2)$，然后使用其本地数据集在相应各方优化 $\mathcal{L}_i(\theta_i)$。

Fed+[61] 认为，稳健聚合可以更好地处理各方数据的异质性，并通过模型正则化的方法来适应个性化。Fed+ 引入了一个凸的惩罚函数 $\phi$ 和常数 $\alpha, \nu$，如下所示：

$$\min_{\theta, z, \bar{\theta}} \frac{1}{N} \left\{ F_{\nu,\alpha}(\theta, z, \bar{\theta}) = \sum_{i=1}^{N} \mathcal{L}_i(\theta_i) + \frac{\alpha}{2} \|\theta_i - z_i\|_2^2 + \nu\phi(z_i - \bar{\theta}) \right\} \tag{1.22}$$

并提出了一个当前局部和全局模型的稳健组合，通过对 $z_i$ 进行最小化（1.22），保持 $\theta_i$ 和 $\bar{\theta}$ 固定而得到。设置 $\phi(\cdot) = \|\cdot\|_2$ 可以得到几何中位数的 $\rho$-平滑近似值，这是一种稳健的聚合形式：

$$z_i \leftarrow \bar{\theta} + \text{prox}_{\phi/\rho}(\theta_i - \bar{\theta}), \rho := \nu/\alpha$$
$$z_i = (1-\lambda_i)\theta_i + \lambda_i\bar{\theta}, \lambda_i := \min\left\{1, \rho/\|\theta_i - \bar{\theta}\|_2\right\}$$

为了从 $\{\theta_i\}$ 中计算出稳健的 $\bar{\theta}$，聚合器运行以下两步迭代程序，初始化为 $\bar{\theta} = \theta_{\text{mean}} := \text{Mean}\{\theta_i\}$，直到 $\bar{\theta}$ 收敛：

$$v_i \leftarrow \max\left\{0, 1 - (\rho/\|\theta_i - \bar{\theta}\|_2)\right\}(\theta_i - \bar{\theta})$$
$$\bar{\theta} \leftarrow \theta_{\text{mean}} - \text{Mean}\{v_i\}$$

### 1.3.8 多任务学习

传统的机器学习方法通常为单一任务优化一个模型。多任务学习（MTL）[4, 7, 54] 扩展了这种传统的方法，以联合学习多个任务，从而利用任务之间的共同点和差异来潜在地提高单个任务的性能。由于这些方法可以学习非 IID 和不平衡数据集之间的关系，它们很适合应用于联合学习的环境 [53]。对多任务学习感兴趣的读者应该参考以下调查论文 [46, 65]，以获得该领域的概况。

虽然 MTL 方法在联合学习的背景下很有吸引力，但它们并没有考虑到框架中的通信挑战，如容错和散兵游勇。MOCHA[53] 是第一个使用多任务学习的联合学习框架，它在训练中考虑了容错和散兵游勇。多任务学习方法通常将问题表述如下：

$$\min_{w,\Omega}\left\{\sum_{i=1}^{N}\mathcal{L}_i(\theta_i) + R(\Omega)\right\} \tag{1.23}$$

这里，$N$ 是任务的总数，$\mathcal{L}_i(\theta_i)$ 和 $\theta_i$ 是任务 $i$ 的损失函数和参数。矩阵 $\Omega \in \mathbb{R}^{N\times N}$ 表示任务之间的关系，它要么是事先已知的，要么是在同时学习任务模型时估计的。MTL 方法的不同之处在于他们对 $R$ 的表述，即通过 $\Omega$ 矩阵促进任务之间的适当结构。MOCHA 在联合学习环境中使用目标的分布式原始-双重优化表述来优化这一目标。这使得它可以通过只要求一方的可用数据来更新本地模型参数，从而将各节点的计算分开。MOCHA 显示了多任务学习方法在联合学习环境中的适用性，并且与在实验数据集

上训练的全局和局部模型相比，表现出更好的性能。然而，它只为凸型模型设计，因此不适用于非凸型深度学习模型。

VIRTUAL（变异联合多任务学习）[14] 通过使用变异推理方法将多任务学习框架扩展到非凸模型。给定 $N$ 个当事方，每个当事方都有数据集 $D_i$、局部模型参数 $\theta_i$ 和中心模型参数 $\theta$，VIRTUAL 计算出后验分布

$$p(\theta, \theta_1, \cdots, \theta_N | D_{1:N}) \propto \frac{\prod_{i=1}^{N} p(\theta, \theta_i | D_i)}{p(\theta)^{N-1}} \tag{1.24}$$

这种后验分布假定：（1）在给定聚合器和政党参数的情况下，政党数据是有条件独立的，$p(D_{1:N}|\theta, \theta_1, \cdots, \theta_N) = \prod_{i=1}^{N} p(D_i|\theta, \theta_i)$ 和（2）先验的因子化为 $p(\theta, \theta_1, \cdots, \theta_N) = p(\theta) \prod_{i=1}^{N} p(D_i|\theta, \theta_i)$。由于方程（1.24）中定义的后验分布是难以处理的，该算法提出了一种类似期望传播的算法 [41] 来近似后验。

## 1.4 个性化技术的基准

在这一节中，我们回顾了适合作为联合学习中每个人标准化方法的基准的数据集。我们考虑的是具有非 IID 方数据分布的数据集，即每一方的数据是从不同的分布中取样的。我们回顾了先前工作中使用的数据集，以及其他可能适合个性化问题设置的数据集。

大体上分类，该领域的先前工作使用了以下类型的数据集之一：（a）合成数据集，其中定义了数据集的生成过程，以生成各方的数据样本；（b）通过根据一些假设对常用的数据集（如 MNIST 或 CIFAR-10[32]）进行分区，模拟联合数据集；以及（c）利用具有自然分区的数据集，如从多方收集的数据。我们现在详细研究一下这些类型中的每一种。

### 1.4.1 合成的联合数据集

一个相当常见的生成合成联合数据集的方法是遵循 Shamir 等人 [49] 提出的方法，并添加了一些修改以注入各方的异质性。虽然所提出的方法之间生成数据集的确切方式不同，但基本过程如下：对于每个设备 $k$，根据模型 $y = \arg\max(\text{softmax}(\boldsymbol{W}x + b))$ 生成样本（$\boldsymbol{X}_k, \boldsymbol{Y}_k$）。模型参数 $\boldsymbol{W}_k$ 和 $b_k$ 由参数 $\alpha$ 控制，采样方式为：$u_k \sim \mathcal{N}(0, \alpha), \boldsymbol{W}_k \sim \mathcal{N}(u_k, 1), b_k \sim \mathcal{N}(u_k, 1)$。$\boldsymbol{X}_k$ 的生成由第二个参数 $\beta$ 控制，其采样方式为：$\boldsymbol{B}_k \sim \mathcal{N}(0, \beta), v_k \sim \mathcal{C}(\boldsymbol{B}_k, 1)$, $\sum$ 是一个对角线协方差矩阵，$\sum_{j,j} = j^{-1,2}$, 和 $x_k \sim \mathcal{N}(v_k, \sum)$。

这个合成数据集 Synthetic$(\alpha, \beta)$ 有两个参数：$\alpha$ 和 $\beta$。这里，$\alpha$ 控制本地模型之间的差异程度，$\beta$ 控制每个设备上的本地数据与其他方的数据的差异程度。

### 1.4.2 模拟联合数据集

模拟联合数据集的一个常见方法是使用常用的数据集，并根据一个假设在各方之间进行划分。之前的工作一般利用 MNIST,2CIFAR-10 和 CIFAR-100[32] 等数据集来完成这一任务。由于这些数据集没有特定的自然特征可用于在各方之间进行划分，我们需要根据假设对其进行划分。一种分区方式是对每一方的数据点从特定的类子集中抽样，以确保各方不会看到来自所有类的数据，从而没有代表数据集中所有类的特征。另一种分区方式包括使用数据样本在各方之间的概率分配–如抽样 $p_k \sim \mathrm{Dir}_N(\alpha)$，并将 $p_{k,i}$ 比例的 $k$ 类实例分配给 $i$ 方 [64]。

合成的联合数据集的优点是可以控制数据集中的异质性数量；但是，它们受限于所能支持的各方数量。以前的工作都是以几十人的数量来进行实验的。虽然这适合于企业环境中的联合学习，因为在企业环境中，各方的数量通常不会有太大的变化，但这种设置并没有考虑到在智能手机或物联网类型的应用中通常遇到的规模。

### 1.4.3 公共联合数据集

除了合成和模拟的联合数据集，还有一些数据集，它们支持各方之间的数据自然分割。LEAF[6] 是一个流行的联合学习方法的基准，它为图像分类、语言建模和情感分析任务提供多个数据集。这些数据集由于接近现实生活中的非 IID 数据特征，以及它们所支持的各方规模，因此是衡量个性化技术的首选数据集。LEAF 中包含的一些数据集是：

- **FEMNIST**：Extended MNIST（EMNIST）[13] 是一个包含数字、大写和小写字符的手写样本的数据集，总共有 62 个类标签。EMNIST 数据集被手写样本的原始写作者划分为 3,500 多方，以创建 FEMNIST 数据集。

- **Shakespeare 数据集**：该数据集用于语言建模任务，是根据《威廉-莎士比亚全集》3 建立的，将每部戏剧中的每个说话的角色视为一个单独的当事人。

关于 LEAF 中可用的数据集及其总量和政党层面的统计数据的详情，可在表 4.2 中找到。

除了 LEAF 中提供的数据集，还有其他一些数据集也具有联合学习任务中个性化所需的特征。这些数据集中的一些是：

- **Google Landmarks Dataset v2 (GLDv2)**：GLDv24 是一个大规模的细粒度数据集，用于实例识别和图像检索任务 [60]。它由 246 个国家的大约 500 万张人造和自然地标的图像组成，这些图像有大约 20 万个不同的标签。这个数据集可以通过根据地标的地理位置或者是地标的位置来划分，从而在联合学习环境中得到利用，或地标类别，或作者。鉴于其规模和多样性，这个数据集是个性化任务的一个强有力的测试平台。

- **MIMIC-III:**Medical Information Mart for Intensive Care III (MIMIC-III) [30] 是一个大规模的去识别的健康相关数据，包括 2001 年至 2012 年期间在马萨诸塞州波士顿一家医院的重症监护室住院的 4 万多名患者。它包括超过 60,000 名重症监护室住院患者的人口统计学、生命体征测量、实验室测试结果、程序、药物、护理人员记录、成像报告和死亡率（包括院内和院外）等信息。虽然这个数据集的规模与 GLDv2 相比是有限的，但它是最大的可用医疗数据集之一，因此为评估联合学习的个性化提供了一个重要的基准。

## 1.5 个性化是附带的参数问题

对于联合学习中的个性化限制，有一个可能的理论解释：附带参数问题。我们考虑联合学习中个性化的一般模型：聚合者和各方的目标是解决一个优化问题，其形式为

$$\min_{\theta,\theta_1,\cdots,\theta_N} \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_i(\theta,\theta_i) \tag{1.25}$$

其中， 是共享的模型参数， i 是特定政党的参数，Li 是第 i 个政党的经验风险。在大多数联合学习的设置中，每一方的样本量是有限的，所以对特定政党的参数 ia 的估计只能达到一定的精度，而这个精度取决于每一方的样本量。我们可能希望有可能更准确地估计共享参数 ，但这只在一定程度上是可能的。

该问题的人口版本是

$$\min_{\theta,\theta_1,\cdots,\theta_N} \frac{1}{N}\sum_{i=1}^{N} \mathcal{R}_i(\theta,\theta_i) \tag{1.26}$$

其中 $\mathcal{R}_i$ 是第 $i$ 方的（人口）风险：$\mathcal{R}_i(\cdot) = E[\mathcal{L}_i(\cdot)]$。让 $(\hat{\theta},\hat{\theta}_1,\cdots,\hat{\theta}_N)$ 和 $(\theta^*,\theta_1^*,\cdots,\theta_N^*)$ 分别为 (4.27) 和 (4.28) 的 argmin。共享参数的估计值满足得分方程：

$$0 = \frac{1}{N}\sum_{i=1}^{N} \partial\theta\mathcal{L}_i(\hat{\theta},\hat{\theta}_1,\cdots,\hat{\theta}_N) \tag{1.27}$$

围绕 $(\theta^*,\theta_1^*,\cdots,\theta_N^*)$ 展开得分方程（并放弃高阶项），我们有

$$\begin{aligned} 0 = \frac{1}{N}\sum_{i=1}^{N} &\partial_\theta\mathcal{L}_i(\theta^*,\theta_1^*,\cdots,\theta_N^*) + \partial_\theta^2\mathcal{L}_i(\theta^*,\theta_1^*,\cdots,\theta_N^*)(\hat{\theta}-\theta^*) \\ &+\partial_{\theta_i}\partial_\theta\mathcal{L}_i(\theta^*,\theta_1^*,\cdots,\theta_N^*)(\hat{\theta}-\theta^*) \end{aligned}$$

我们重新排列以隔离共享参数的估计误差

$$\begin{aligned} \hat{\theta}-\theta^* = (\frac{1}{N}\sum_{i=1}^{n} &\partial_\theta^2\mathcal{L}_i(\theta^*,\theta_1^*,\cdots,\theta_N^*))^{-1}(\frac{1}{N}\sum_{i=1}^{N}\partial_\theta\mathcal{L}_i(\theta^*,\theta_1^*,\cdots,\theta_N^*) \\ &+\partial_{\theta_i}\partial_\theta\mathcal{L}_i(\theta^*,\theta_1^*,\cdots,\theta_N^*)(\hat{\theta}-\theta^*)) \end{aligned}$$

我们看到，政党特定参数的估计误差通过 $\partial_{\theta_i}\partial_\theta\mathcal{L}_i(\theta^*,\theta_1^*,\cdots,\theta_N^*)(\hat{\theta}_i-\theta_i^*)$ 的平均值影响共享参数的估计误差。这个平均数通常不是平均值为零，因此，即使各方的数量增加，它也不会收敛为零。要使这个平均值收敛为零，必须发生以下两种情况之一：

1. $\hat{\theta}_i - \theta_i^* \to^p 0$：个性化参数的估计误差收敛为零。只有当每一方的样本量增加时，这才有可能。不幸的是，在大多数联合学习问题中，各方的计算和存储限制排除了这种情况。

2. $\partial_{\theta_i}\partial_\theta\mathcal{L}_i(\theta^*,\theta_1^*,\cdots,\theta_N^*)$ 为均值零。这相当于分数方程（4.29）满足某种正交性属性 [12，42]。如果分数方程满足这一属性，那么特定政党参数的估计误差就不会影响（一阶）共享参数的估计。尽管这是非常理想的，但正交性只发生在某些特殊情况下。

综上所述，共享参数的估计误差通常受到党派特定参数的估计误差的影响，而且在现实的联合学习环境中，它不会收敛为零，在这种情况下，党派

的数量会增加，但每个党派的样本量仍然是有限制的。退一步讲，从自由度的角度来看，这也是可以预期的。随着各方数量的增加，虽然总样本量增加，但我们必须学习的参数总数也会增加。换句话说，联合学习中的个性化是一个高维度的问题。众所周知，这样的问题是具有挑战性的，除非参数表现出特殊的结构（如稀疏性、低等级），否则估计误差一般不会收敛为零。不幸的是，在联合学习中通常不是这种情况。

实际上，这意味着如果我们在联合学习问题中加入了个性化，那么在不增加每一方的样本量的情况下，增加一方的数量超过一定程度就是暴殄天物。我们是否超过了这一点，可以通过检查共享参数估计的质量是否随着更多的缔约方的加入而提高来确定。如果我们超过了这一点，那么特定政党参数的估计误差就会支配共享参数的估计误差，所以参数共享就没有好处。这就是所谓的附带参数问题，它在统计学中有着悠久的历史。我们可以参考 [33] 对该问题的回顾。

## 1.6 总结

在这一章中，我们通过证明原生的联合学习与他们在本地训练模型相比，不一定能帮助所有各方训练出更好的模型来激发个性化的需求。为了缓解这个问题，人们提出了不同的个性化技术。我们根据这些技术所采用的个性化策略的类型，将其分为八大类。除了对个性化策略的回顾之外，我们还对联合学习中的个性化的统计挑战进行了概述。在本章的最后，我们提供了在实施或利用个性化策略时需要考虑的实际问题的建议，以及未来的研究方向和对联合学习中的个性化理论理解的开放问题。

**实际考虑。**为你的应用选择一个个性化的策略，与参与联合学习设置的各方和聚合器的属性密切相关。具体来说，有助于做出明智选择的问题是：(1) 所有各方都有相同的模型架构吗？(2) 是否有一个数据共享机制可以用来增加本地数据？(3) 参与方和聚合器的计算能力如何？(4) 你希望每一方有多少数据？

如果不是所有各方都有相同的模型架构，或者最好是不同各方有不同的架构，那么应该探索基于蒸馏的方法（第 4.3.4 节）或 LG-FedAvg [35] 方法。这些技术支持并已被实验证明可以与不同的当事方和全局模型架构一起工作。另一个重要的考虑因素是是否有可用的全局数据来增强本地数据。虽然共享聚会数据违反了联合学习的核心原则，但如果有可能收集到一个

共享的数据集，用于个性化的数据增强技术（第 4.3.3 节）可以成为这些场景中强有力的候选者。

当事人和聚合者的计算能力在选择个性化策略方面也起着重要作用。具体到当事人，如果计算能力和内存能力充足，那么可以探索混合模型的方法（第 4.3.6 节）。由于混合模型的方法在各方身上保持了一个局部和一个全局模型，并使用两者的组合进行推理，它大大增加了参与各方的内存和计算要求。按照类似的思路，如果聚合器有足够的内存容量，允许维护多个模型参数，那么客户（方）聚类方法（第 4.3.1 节）可能会有帮助。

有助于做出明智决定的最后一个问题是关于每一方的可用数据量。这是应用情境化方法的一个重要考虑。客户端（当事方）情境化（第 4.3.2 节）增加了从本地数据中学习的参数数量，如果当事方有足够的数据，那么就有可能学习这些情境参数来帮助实现个性化。

最后，无论上述条件是否满足，元学习（第 4.3.5 节）和模型正则化（第 4.3.7 节）方法都适用，在选择个性化策略时应始终考虑。

**推进联合学习中的个性化。**随着文献中提出的个性化算法越来越多，我们认为下一个重要的步骤是建立基准和性能指标，以有效和可靠地衡量所提出的技术的性能。这需要有模仿实际部署中通常遇到的情况的数据集。虽然有一些数据集可用于这一目的，但需要在更广泛的应用领域提供更多的数据集。在标准化的数据集上进行基准测试，可以更好地解释所提出的技术的能力和局限性，也可以方便地比较各种技术。除了数据集之外，还需要一个标准化的个性化评估设置。评价联合学习技术的典型方法是测量全局模型的性能，这种技术也被移植到了个性化问题上。然而，正如我们在个性化的激励性例子中所看到的，测量全局模型的准确性并不一定能提供对每一方性能的完整描述。因此，定义一个考虑到每一方性能的评估设置将成为有效评估个性化技术的一个重要贡献。

**对个性化的理论理解。**正如我们所看到的，由于偶然的参数问题，个性化存在着一个可扩展性问题。这个问题因其统计学性质而与机器学习中的大多数可扩展性问题相区别。克服这个问题是将个性化扩展到大型聚会云的一个要求。不幸的是，一个世纪以来，统计学界一直没有找到解决底层偶然参数问题的一般方法，所以不太可能有一般的方法来进行大规模的个性化定制。然而，也许有可能开发出针对特定模型的解决方案

# 第二章　通信高效的分布式优化算法

Gauri Joshi and Shiqiang Wang

## 摘要

在联邦学习中，连接边缘参与者和中央聚合器的通信链路有时是有带宽限制的，而且会有很高的网络延迟。因此，设计和部署具有通信高效的分布式训练算法是急需的。在本章中，我们将回顾两种不同的具有通信高效的分布式随机梯度下降（SGD）方法：（1）本地更新随机梯度下降（SGD），客户端进行多次本地模型更新，并周期性的进行聚合；（2）梯度压缩和稀疏化方法，以减少每次更新传输的比特数。在这两种方法中，误差收敛与迭代次数和通信效率之间存在着权衡关系。

## 2.1　引言

**ML 训练中的随机梯度下降。**大多数监督学习问题都是使用经验风险最小化框架来解决的 [1, 2]，其目标是最小化经验风险目标函数 $F(\boldsymbol{x}) = \sum_{j=1}^{n} f(\boldsymbol{x}, \xi)/n$。其中，$n$ 是训练数据集的大小，$\xi_j$ 是第 $j$ 个已标注的训练样本，$f(\boldsymbol{x}, \xi_n)$ 是（通常是非凸的）损失函数。一种普遍优化 $F(\boldsymbol{x})$ 的算法是随机梯度下降 (SGD)。在这种算法中，我们计算 $f(\boldsymbol{x}, \xi_n)$ 在小的、随机选择的子集 $\mathcal{B}$（称为迷你批次）上的梯度，每个子集有 $b$ 个样本 [3, 4, 5, 6, 7, 8]，并根据 $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \eta \sum_{i \in \mathcal{B}} \nabla f(\boldsymbol{x}_k; \xi_i)/b$ 更新 $\boldsymbol{x}$，其中 $\eta$ 被称为学习率或步长。虽然算法是为凸目标设计的，但由于小型批量 SGD 有能力摆脱鞍点

和局部最小值 [9, 10, 11, 12]，因此即使在非凸损失表面也有良好的表现。因此，它是最先进的机器学习中的主导训练算法。

对于像 Imagenet[13] 这样的大规模数据集，在单个节点上运行小批量的 SGD 可能会非常慢。进行梯度计算并行化的一个标准方法是参数服务器（PS）框架 [14]，由一个中央服务器和多个工作节点组成。拖拽的工人节点和通信延迟会成为将这个框架扩展到大量工人节点的瓶颈。一些方法，如异步 [15, 16, 17, 18] 和周期性梯度聚合 [19, 20, 21] 已被提出，以提高基于数据中心的 ML 训练的可扩展性。

**联邦学习的动机。**尽管算法和系统的进步提高了效率和可扩展性，但基于数据中心的训练仍有一个主要局限。它要求将训练数据集集中在参数服务器上，由它在工作节点上进行打乱和拆分。手机、物联网传感器和具有设备上计算能力的相机等边缘方的迅速扩散，导致了这种数据分区模式的重大转变。边缘方从它们的环境中收集丰富的信息，这些信息可用于数据驱动的决策。由于有限的通信能力以及隐私问题，这些数据不能直接发送到云端进行集中处理或与其他节点共享。联邦学习框架建议将数据留在边缘方，而将模型训练放在边缘。在联邦学习中，数据被保存在边缘方，模型以分布式的方式被训练。只有梯度或模型更新在边缘方和聚合器之间进行交换。

**系统模型和符号。**如图 2.1所示，一个典型的联邦学习设置包括一个连接到 $K$ 个边缘方的中央聚合器，其中 $K$ 可能是数千甚至数百万的量级。每一方 $i$ 都有一个由 $n_i$ 个样本组成的本地数据集 $\mathcal{D}_i$，它不能被传输到中央聚合器或与其他边缘方共享。我们用 $p_i = n_i/n$ 来表示第 $i$ 方所占数据比例，其中 $n = \sum_{i=1}^{K} n_i$。聚合器试图用本地数据集的并集 $\mathcal{D} = \cup_{i=1}^{K} \mathcal{D}_i$ 来训练一个机器学习模型 $\boldsymbol{x} \in \mathbb{R}^d$。模型向量 $x$ 包含模型的参数，例如，神经网络的权重和偏差。为了训练模型 $x$，聚合器试图最小化以下经验风险目标函数：

$$F(\boldsymbol{x}) := \sum_{i=1}^{K} p_i F_i(\boldsymbol{x}) \tag{2.1}$$

其中 $F_i(\boldsymbol{x}) = \frac{1}{n_i} \sum_{\xi \in \mathcal{D}_i} f(\boldsymbol{x}; \xi)$ 是第 $i$ 方的本地目标函数。$f$ 是由模型 $\boldsymbol{x}$ 定义的损失函数（可能是非凸的），$\xi$ 代表本地数据集 $\mathcal{D}_i$ 的一个数据样本。请注意，我们分配的权重与第 $i$ 方的数据比例成正比。这是因为我们想模拟一个集中式的训练场景，将所有的训练数据传输到一个中央参数服务器。因此，拥有更多数据的一方将在全局目标函数中获得更高的权重。

由于边缘方的资源限制和大量参与方，联邦训练算法必须在严格的通信限制下运行，并应对数据和计算的异质性。例如，连接每个边缘方和中央聚

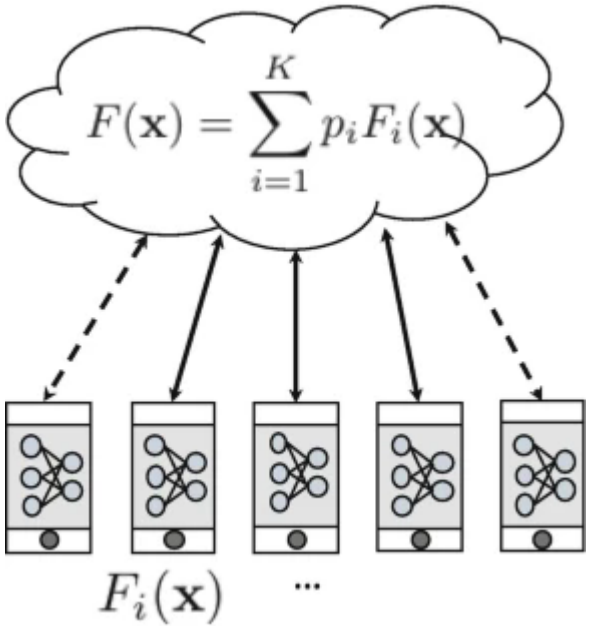



图 2.1: 在联邦优化中，聚合器的目标是最小化边缘方局部目标函数 $F_i(\boldsymbol{x})$ 的加权平均值

合器的无线通信链路可能有带宽限制，并且有很高的网络延时。另外，由于网络连接有限和电池的限制，边缘方可能只是间歇性地可用。因此，在给定的时间内，只有 $K$ 个边缘方中只有 $m$ 个子集可以参与训练模型 $x$。为了在这些通信约束条件下运行，联邦学习框架需要新的分布式训练算法，超越在数据中心环境中使用的算法。在第 6.2 节中，我们回顾了 local-update SGD 算法及其变体，这些算法减少了边缘各方与聚合器的通信频率。在第 6.3 节中，我们回顾了压缩和量化的分布式训练算法，这些算法减少了边缘各方发送给聚合器的每次更新的比特数量。

## 2.2 Local-Update SGD 和 FedAvg

在本节中，我们首先讨论 local-update SGD 及其变体。FedAvg 算法是联邦学习的核心，它是 local-update SGD 的扩展。我们将讨论 FedAvg 如何建立在 local-update SGD 之上，以及在联邦学习中用来处理数据和计算异质性的各种策略。

### 2.2.1 Local-Update SGD 及其变体

**同步分布式 SGD。**在数据中心的设置中，训练数据集 $\mathcal{D}$ 被打乱并平均分配到 $m$ 个工作者节点中。训练机器学习模型的标准方法是使用同步分布式 SGD，其中梯度由工作者节点计算，然后由中央参数服务器汇总。在同步 SGD 的第 t 次迭代中，工作者从参数服务器中拉取模型的最新版本 $x_t$。每个工作者 $i$ 使用从本地数据集 $\mathcal{D}_i$ 中抽取的 B 个样本的小批量数据 $\mathcal{B}$ 来计算一个小批量随机梯度 $g_i(x) = \sum_{\xi \in \mathcal{B}} f(x;\xi)$。然后，参数服务器收集来自所有工作者的梯度，并按以下方式更新模型参数

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \frac{\eta}{m} \sum_{i=1}^{m} g_i(\boldsymbol{x}) \tag{2.2}$$

随着工作者数量 $m$ 的增加，同步 SGD 的误差与迭代收敛性得到改善。然而，由于工作者的局部梯度计算时间存在差异，等待所有工作者完成梯度计算的时间也会增加。为了提高工作者数量的可扩展性，中提出了同步 SGD 的 <mark> 拖拽者弹性变体 </mark>，进行异步梯度聚合。

**Local-Update SGD。**尽管异步聚合方法对提高分布式 SGD 的可扩展性很有效，但在许多分布式系统中，工作者和参数服务器之间交换梯度和模型更新的通信时间会主导本地梯度计算时间的变化。因此，每次迭代后的节点间的持续通信可能是过于昂贵和缓慢的。Local-Update SGD 是一种通信效率高的分布式 SGD 算法，它通过让工作者节点执行多次本地 SGD 更新而不是仅仅计算一个小批量梯度来克服这个问题。

如图 6.2 所示，Local-Update SGD 将训练分为几轮通信。在一个通信轮中，每个工作者使用 SGD 对其目标函数 $F_i(x)$ 进行局部优化。每个工作者 $i$ 从当前的全局模型开始，用 $x_t$ 表示，并执行 $\tau$ 次 SGD 迭代，以获得模型 $x_{t+\tau}^{(i)}$。然后，所得到的模型由 $m$ 个工作者发送到参数服务器，服务器对其进行平均，以更新全局模型，如下所示:

$$x_{t+\tau} = \frac{1}{m} \sum_{i=1}^{m} x_{t+\tau}^{(i)}$$

**Local-Update SGD 每一次迭代的运行时间。**通过在与参数服务器通信之前在每个工作器上执行 个本地更新，本地更新 SGD 减少了每次迭代的预期运行时间。让我们通过考虑以下延迟模型来量化这种运行时间的节省。第 i 个工作者在第 k 个局部步骤计算小批梯度的时间被建模为随机变量 Yi,k FY，假定在工作者和小批之间是独立和相同的分布（i.i.d）。通信延

迟用一个常数 D 表示，它包括将本地模型发送到参数服务器和从参数服务器接收平均的全局模型所需的时间。由于每个工人 i 进行了 次本地更新，其平均本地计算时间（完成图 6.3 中 3 个蓝色箭头的一个序列所需的时间）由以下公式给出

$$\bar{Y} = \frac{Y_{i,1} + Y_{i,2} + \cdots + Y_{i,\tau}}{\tau}$$

如果 =1，在这种情况下，本地更新 SGD 会简化为同步 SGD，那么随机变量 Y 与 Y 是相同的。

$$\mathbb{E}[T_{Local-update}] = \mathbb{E}[max(\bar{Y}_1, \bar{Y}_1, \cdots, \bar{Y}_m] + \frac{D}{\tau} = \mathbb{E}[\bar{Y}_{m:m}] + \frac{D}{\tau}$$

术语 $Y_{m:m}$ 表示具有概率分布 $Y \sim F_Y$ 的 $m$ 个 i.i.d. 随机变量的最大顺序统计。从（6.6）中我们可以看出，执行更多的局部更新可以通过两种方式减少每次迭代的运行时间。首先，通信延迟在 次迭代中得到摊销，并减少了一个系数 。其次，执行局部更新也提供了一个减少散兵游勇的好处，因为 $\bar{Y}_{m:m}$ 的尾部比 $Y_{m:m}$ 轻，因此（6.6）中的第一个项随着 $\tau$ 而减少。

** 本地更新 SGD 的错误收敛 **。正如我们在上面看到的，将工作者和参数服务器之间的通信频率降低到在 迭代中只有一次，可以使每次迭代的运行时间大大减少。然而，设定一个大的 $\tau$ 值，局部更新的数量会导致较差的误差收敛。这是因为，随着 $\tau$ 的增加，工作节点的模型 $x_{t+\tau}^{(i)}$ 会相互偏离。论文给出了局部更新 SGD 在局部更新数量 $\tau$ 方面的错误收敛分析。假设目标函数 $F(x)$ 是 $L$-Lipschitz 光滑的，学习率 $\eta$ 满足 $\eta L + \eta^2 L^2 \tau(\tau - 1)$ 1。随机梯度 $g(x;\xi)$ 是 $\nabla F(x)$ 的无偏估计，即 $\mathbb{E}_\xi[g(x;\xi)] = \nabla F(x)$。随机梯度 $g(x;\xi)$ 被假定为有边界的方差，即 $Var(g(x;\xi))$ $\sigma^2$。如果起点是 $x_1$，那么经过局部更新 SGD 的 $T$ 次迭代后，$F(x_T)$ 被约束为

$$\mathbb{E}[\frac{1}{T}\sum_{t=1}^{T} \|\nabla F(x_t)\|^2] \leq \frac{2[F(x_1 - F_{\text{inf}})]}{\eta t} + \frac{\eta L \sigma^2}{m} + \eta^2 L^2 \sigma^2 (\tau - 1)$$

其中 $x_t$ 表示第 $t$ 次迭代时的平均模型。设置 $\tau = 1$ 使得本地更新 SGD 及其误差收敛边界与同步分布式 SGD 相同。随着 $\tau$ 的增加，约束的最后一项会增加，从而增加收敛时的误差底线。

** 适应性沟通策略 **。从上面的运行时间和误差分析中，我们可以看到，当我们改变 时，每个迭代的误差和通信延迟之间存在着权衡。较大的 $\tau$ 可以减少预期的通信延迟，但是产生更差的误差收敛。为了获得快速收敛和低误差底线，提出了一个在训练过程中适应 $\tau$ 的策略。对于一个固定的

学习率 $n$，中的以下策略会逐渐减少 $\tau$:

$$\tau_\ell = \left\lceil \sqrt{\frac{F(x_{t=\ell T_0})}{F(x_{t=0})}} \tau_0 \right\rceil \tag{2.3}$$

其中，$\tau_\ell$ 是训练中 $T_0$ 秒的第 $\ell$ 个区间内的局部更新次数。这个更新规则也可以被修改，以考虑到基本的可变学习率时间表（图 6.4）。

** 弹性平均法和重叠 SGD**。在本地更新的 SGD 中，在下一组 更新开始之前，需要将更新的全局模型传达给各节点。此外，在 m 个节点中最慢的节点完成其 个本地更新之前，全局模型不能被更新。这种通信障碍会成为全局模型更新的瓶颈，并增加每轮训练的预期运行时间。由于这种通信障碍是由算法而不是系统实现强加的，我们需要一种算法方法来消除它，并允许通信与本地计算重叠。诸如等作品使用异步梯度聚合来消除同步障碍。然而，异步聚合会导致模型僵化，也就是说，慢速节点会有任意过时的全局模型版本。最近的一些工作提出了本地更新 SGD 的变种，允许通信和计算的重叠。在这些算法中，工作节点从一个锚模型开始他们的本地更新，该模型甚至在最慢的节点完成上一轮本地更新之前就可以使用。这种方法受到提出的弹性平均 SGD（EASGD）算法的启发，该算法在目标函数中增加了一个近似项。近似方法，如，虽然不是为此目的而设计的，但自然允许通信和计算的重叠。

6.2.2 联合平均法（FedAvg）算法及其变体

**FedAvg 算法。** 由于在联合学习中，边缘伙伴的通信能力有限，本地更新的 SGD 特别适合于联合学习的 132G。Joshi 和 S. Wang 的学习框架，在这里它被称为 FedAvg 算法。其主要区别如下。首先，作为云中服务器的工作节点被移动和物联网设备等边缘方所取代。由于边缘方的间歇性可用性，与数据中心的设置不同，每轮训练中只有 K 方中的 m 个子集参与。其次，数据集 Dican 的大小和组成在边缘方之间都是高度异质的，不像数据中心的设置，数据集 D 被洗牌并均匀地划分到工人节点。

联合平均算法（FedAvg）也将训练分为通信轮。在一个通信轮中，聚合器从可用的各方中均匀地随机选择 m 个边缘方。每个边缘方使用类似于局部更新 SGD 的 SGD 对其目标函数 Fi(x) 进行局部优化。与基本的本地更新 SGD 不同的是，每个工作者执行相同数量的本地更新 ，在 FedAvg 中，本地更新的数量 可能在不同的边缘方和通信回合中有所不同。一个常见的实施做法是，各方运行的本地纪元 E 是相同的。因此， i= Eni 其中 B 是小批量的大小。另外，如果每个通信轮次在壁钟时间上有一个固定的长度，

那么 irep 代表 i 方在时间窗口内完成的局部迭代，并且可能在不同的客户（取决于他们的计算速度和可用性）和不同的通信轮次中变化。在第 r 轮通信中，边缘各方从全局模型 xr,0 开始，各自进行 i 个局部更新。假设他们得到的模型用 x(i) r, i 表示。共享的全局模型 xris 的更新方式如下。其中 pi=|Di||D|，第 i 个边缘方的数据部分。

** 处理数据异质性的策略。** 由于数据集在各节点间高度异质化，边缘方的本地训练模型可能彼此有很大的不同。而且随着本地更新数量的增加，模型可能会变得对本地数据集过度拟合。因此，FedAvg 算法可能会收敛到一个不正确的点，而这个点不是全局目标函数 F（x）的静止点。例如，假设每个边缘方执行了大量的局部更新，第 i 方的局部模型收敛到 x(i)∗= min Fi(x)。那么这些局部模型的加权平均将收敛于 x= Ki=1pix(i)∗，这可能与真正的全局最小值 x∗=min F(x) 有任意的不同。为了减少这种由数据异质性引起的求解偏差，一个解决方案是选择一个小的或衰减的学习率 和或保持小的局部更新数量 。其他用于克服解决方案偏差的技术包括近似的局部更新方法，如，该方法为全局目标添加了一个正则化项，以及旨在最小化跨方模型的方法。通过交换控制变量来实现漂移。在高层次上，这些技术阻止了边缘方的模型偏离全局模型的情况。

** 处理计算异质性的策略 ** 数据异质性的影响会因边缘各方的计算异质性而加剧。即使边缘各方进行不同数量的局部更新 i，标准的 FedAvg 算法建议将所产生的模型按照数据比例 pi 进行简单的聚合。然而，这可能会导致一个不一致的解决方案，与预期的全局目标不匹配，如所示，并在图 6.5 中说明。最终的解决方案变得偏向于局部最优 x(i)∗= min Fi(x)，而且它可能离全局最小 x(i)∗= min F(x) 有任意的距离。论文通过将累积的局部更新（x(i)r, i- x(i)r,0）按局部更新的数量 i 进行归一化，然后再将其发送到中央聚合器，从而解决了这种不一致的情况。这种被称为 FedNova 的规范化联合平均算法的结果是一致的解决方案，同时保留了快速收敛率。

除了局部更新数量 i 的变化，由于边缘方使用局部动量、自适应局部优化器（如 AdaGrad）或不同的学习率计划，也可能出现计算异质性和解决方案不一致的情况。在这些情况下，需要一个通用的 FedNova 版本来解决不一致的问题。

** 处理边缘方间歇性可用性的策略 ** 在一个联合学习设置中，边缘方的总数可以达到数千甚至数百万台设备的数量。由于本地计算资源的限制和带宽的限制，边缘方只能间歇性地参与训练。例如，目前手机只有在插电

充电时才会被用于联合训练，以节省电池。因此，在每一轮通信中，只有一小部分边缘方参与到 FedAvg 算法中。大多数关于设计和分析联合学习算法的工作都假设边缘方的子集是从整个边缘方集合中均匀地随机选择的。这种部分和间歇性的参与通过给误差增加一个方差项而放大了数据异质性的不利影响。最近一些 134G. Joshi 和 S. Wang 的作品提出了应对这种异质性并提高收敛速度的客户端选择方法。这些策略将更高的选择概率分配给具有较高局部损失的边缘方，并表明它可以加速全局模型的进展。然而，这种加速是以较高的非消失偏差为代价的，这种偏差随着数据异质性程度的增加而增加。论文提出了一种自适应策略，逐渐减少选择倾斜，以实现收敛速度和误差底限之间的最佳权衡。

## 2.3 模型压缩

除了执行多次本地更新外，模型在通信和计算过程中也可以被压缩。一种方法是使用标准的无损压缩技术，然而这只能在有限的程度上减少模型的大小，并且需要在接收方进行解压。在本节中，我们将讨论一类特殊的有损压缩技术，该技术通常用于提高联邦学习和分布式 SGD 中的通信效率。这些技术不需要在接收方进行解压，并且可以保证训练收敛。我们在第 6.3.1 和 6.3.2 节中重点介绍了提高通信效率的方法，在 6.3.3 节中重点介绍了提高通信和计算效率的方法。

### 2.3.1 有压缩更新的 SGD

一个广泛使用的方法是压缩各参与方和聚合器之间传输的模型更新 [22, 23]。特别是，我们定义了一个压缩器 $\mathcal{C}(\boldsymbol{z})$，它产生任意向量 $\boldsymbol{z}$ 的压缩版本。流行的压缩器包括那些实现量化 [22] 和稀疏化 [24] 的压缩器。根据它们的特点，压缩器可以分为无偏和一般（即可能有偏）。我们在下文中讨论这两种压缩器的变体，其中我们考虑一种用于一般压缩器的误差反馈技术，这种技术对于避免方差爆炸和保证收敛是很有用的。请注意，我们在本节中的偏差概念是在概率建模的背景下，无偏的压缩器意味着压缩向量的期望值（从该压缩器得到）等于原始向量。

#### 2.3.1.1 没有误差反馈的无偏压制器

一个无偏的压缩器 $\mathcal{C}(\boldsymbol{z})$ 满足以下两个特征:

$$\mathbb{E}[\mathcal{C}(\boldsymbol{z})|\boldsymbol{z}] = \boldsymbol{z} \tag{2.4}$$

$$\mathbb{E}[\|\mathcal{C}(\boldsymbol{z}) - \boldsymbol{z}\|^2|\boldsymbol{z}] \leq q\|z\|^2 \tag{2.5}$$

其中，$q \geq 0$ 是一个常数，用于捕获压缩器实现的相对近似间隙。直观地说，相对近似间隙指的是压缩后的向量与原始向量相比的相对误差。我们很容易看到，$q = 0$ 是 $\mathcal{C}(\boldsymbol{z}) = \boldsymbol{z}$（即不压缩）的必要条件，$q = 1$ 是 $\mathcal{C}(\boldsymbol{z}) = 0$（即不传输）的必要条件。一般来说，较大的 $q$ 对应于由 $\mathcal{C}(\boldsymbol{z})$ 产生的更压缩的向量。正如我们在接下来介绍的“随机-$k$”例子中所看到的，在某些情况下，我们可能会放大压缩结果以保证无偏性，这可能会产生一个大于 1 的 $q$ 值。

*样例:* 无偏压缩器的一个例子是一个随机量化器:

$$[\mathcal{C}(\boldsymbol{z})]_i = \begin{cases} \lfloor z_i \rfloor, \text{以 } \lceil z_i \rceil - z_i \text{ 的概率} \\ \lceil z_i \rceil, \text{以 } \lfloor z_i \rfloor - z_i \text{ 的概率} \end{cases} \tag{2.6}$$

为该向量的第 $i$ 个分量，其中 $\lfloor \cdot \rfloor$ 和 $\lceil \cdot \rceil$ 分别表示下界（向下舍入为整数）和上界（向上舍入为整数）运算符。我们注意到，在浮点表示法的情况下，这里的整数可以是基数。可以很容易地看出，这种量化操作满足无偏性属性 (2.4)。注意到量化操作给出 $q = \max_{y\in[0,1]}(1-y)^2 y + y^2(1-y)$，我们有 $q = 1/4$。

另一个例子是，从原始向量 $\boldsymbol{z}$ 中随机选择 $k$ 个分量，其概率同为 $k/d$，并将结果放大 $d/k$。即，

$$[\mathcal{C}(\boldsymbol{z})]_i = \begin{cases} \frac{d}{k} z_i, \text{以} \frac{k}{d} \text{ 的概率} \\ 0, \text{以} 1 - \frac{k}{d} \text{ 的概率} \end{cases} \tag{2.7}$$

为向量的第 $i$ 个分量。这通常被称为随机-$k$ 稀疏化技术。显然，这种操作也是无偏的。(2.5) 的左边是所有分量 $[(d/k-1)^2 \cdot k/d + 1 \cdot (1-k/d)]z_i^2$ 的总和。因此，我们得到 $q = d/k - 1$。

**带有压缩更新的本地更新 SGD。**当使用压缩和本地更新 SGD 时，每一方都像往常一样计算其本地更新。这些更新在发送到聚合器之前被压缩，然后聚合器对压缩的更新进行平均，以获得下一个全局模型参数。假设一个有 $\tau$ 个迭代的回合从迭代 $t$ 开始，这就给出了以下递推关系。

$$\boldsymbol{x}_{t+\tau} = \boldsymbol{x}_t + \frac{1}{m}\mathcal{C}(\boldsymbol{x}_{t+\tau}^{(i)} - \boldsymbol{x}_t) \tag{2.8}$$

在不同的实现中，可以在服务器上应用另一种压缩操作，以保持相同的压缩水平（例如，量化精度或要传输的组件数量）。这样就可以得到

$$\boldsymbol{x}_{t+\tau} = \boldsymbol{x}_t + \mathcal{C}(\frac{1}{m}\mathcal{C}(\boldsymbol{x}_{t+\tau}^{(i)} - \boldsymbol{x}_t)) \tag{2.9}$$

2.8和 2.9中的操作是相似的，可能是整体近似间隙 $q$ 不同。

**收敛的边界。**在适当选择学习率的情况下，使用 2.8进行 $T$ 次迭代后的最优性（以梯度的平方准则表示）可以被约束为。

$$\mathbb{E}[\frac{1}{T}\sum_{t=1}^{T}\|\nabla F(\boldsymbol{x}_t)\|^2] = \mathcal{O}(\frac{1+q}{\sqrt{T}} + \frac{\tau}{T}) \tag{2.10}$$

其中 $\boldsymbol{x}_t := \frac{1}{m}\sum_{i=1}^{m}\boldsymbol{x}_t^{(i)}$ 对于所有 $t$，即使没有压缩或聚合发生在迭代 $t$ 中，除 $q$, 和 $T$ 之外的常数被吸收在 $\mathcal{O}(\cdot)$ 中，其中 $\mathcal{O}(\cdot)$ 是大 $\mathcal{O}$ 的符号，代表了一个上限，同时忽略了常数。

**方差放大。**从 2.10中，我们可以看到，当 $T$ 足够大时，误差由第一项 $\mathcal{O}(\frac{1+q}{\sqrt{T}})$ 主导。当 $q$ 很大时，我们需要将迭代次数 $T$ 增加 $q^2$ 倍才能消除 $q$ 的影响并达到相同的误差，这是有问题的，因为压缩的优势会被增加的计算量所抵消，特别是对于 random-$k$ 这样的压缩器，$1+q$ 与 $k$ 成反比，正如我们前面讨论的那样。由于 2.10的第一项也与随机梯度的方差成正比，为了简单起见，我们将其吸收到 $\mathcal{O}(\cdot)$ 的符号中，这种现象在文献中也被称为方差吹胀。

接下来，我们将看到，误差反馈可以通过在本地积累压缩参数向量和实际参数向量之间的差异来解决方差吹大的问题，这样就可以在未来的通信回合中传输。

#### 2.3.1.2 带有误差反馈的普通压缩机

我们首先介绍一个一般的（可能是有偏见的）压缩器。一般的压缩器 $\mathcal{C}(\boldsymbol{z})$ 满足以下属性。

$$\mathbb{E}[\|\mathcal{C}(\boldsymbol{z}) - \boldsymbol{z}\|^2|\boldsymbol{z}] \leq \alpha\|\boldsymbol{z}\|^2 \tag{2.11}$$

其中，$\alpha$ 是一个常数，$0 \leq \alpha < 1$，表示压缩机实现的相对近似差距。与 2.4和 2.5中无偏压缩器的特性相比，关键的区别是一般压缩器不保证无偏性。当我们让 $\alpha = q$ 时，方程 2.5和 2.11基本上是相同的，只是为了收敛分析的目的，我们要求 $\alpha < 1$。保持 与 $q$ 不同的另一个原因是为了区分两种类型

的压缩机。满足 2.11的压缩机也被称为 -收缩性压缩机。还有一个更严格的2.11版本，其中不等式在没有期望的情况下成立。

例子一般压缩器的一个典型例子是 top-$k$ 稀疏化技术，它选择幅度最大的 $k$ 个分量。这可以表示为：

$$[\mathcal{C}(\boldsymbol{z})]_i = \begin{cases} z_i, \text{如果}|z_i|\text{在}\{|z_j| : \forall j \in \{1,2,\cdots,d\}\}\text{的最大}k\text{个元素中} \\ 0, \end{cases} \tag{2.12}$$

为矢量的第 $i$ 个分量。由于这个操作对给定的 $\boldsymbol{z}$ 来说是确定的，所以它是有偏差的。我们可以得到 $\alpha = 1 - k/d$，因为 $\boldsymbol{z}$ 中其余分量的平方不能大于幅度最大的 $k$ 个分量。

**具有压缩更新和错误反馈的本地更新 SGD。**当使用错误反馈时，除了在客户端和服务器之间交换压缩的更新外，未被传达的部分（这里称为" 错误"）将在本地累积。在下一轮中，累积的误差将被添加到该轮的最新更新中，这个和向量将被压缩器用来计算压缩向量。每一方 $i$ 保留一个误差向量 $\boldsymbol{e}^{(i)}$，初始化为 $\boldsymbol{e}_0^{(i)} = \mathbf{0}$。在每一轮 $r$ 中，执行以下步骤。

1. 对于每一方 $i \in \{1,2,\cdots,m\}$ 并行：
   a. 从全局参数 $\boldsymbol{x}_r$ 开始，计算局部梯度下降的 $\tau$ 步，以获得 $\boldsymbol{x}_{r,\tau}^{(i)}$。
   b. 将累积误差与当前更新相加：$z_r^{(i)} := \boldsymbol{e}_r^{(i)} + \boldsymbol{x}_{r,\tau}^{(i)} - \boldsymbol{x}_r$。
   c. 计算压缩结果 $\triangle_r^{(i)} := \mathcal{C}(z_r^{(i)})$(这就是将被发送给聚合器的结果)。
   d. 减去压缩结果，得到下一轮的剩余误差 $\boldsymbol{e}_{r+1}^{(i)} = z_r^{(i)} - \triangle_r^{(i)}$。
2. 聚合器根据从各方收到的压缩更新来更新下一轮的全局参数，即

$$\boldsymbol{x}_{r+1} = \boldsymbol{x}_r + \frac{1}{m}\sum_{i=1}^{m} \triangle_r^{(i)} = \boldsymbol{x}_r + \frac{1}{m}\sum_{i=1}^{m} \mathcal{C}(\boldsymbol{z}_r^{(i)}) \tag{2.13}$$

我们可以看到 2.8和 2.13的唯一区别是，我们现在对 $\boldsymbol{z}_r^{(i)}$ 进行压缩，这包括前几轮的累积误差。注意，为了方便起见，我们在这里使用 $r$ 轮索引，而不是 2.8中的迭代索引 $t$。与 2.9类似，上述程序也可以扩展到压缩和累积双方和聚合器的误差。

**收敛的边界。**与 2.10类似，我们提出错误反馈机制的最优性约束。在适当选择学习率的情况下，我们有

$$\mathbb{E}[\frac{1}{Tm}\sum_{t=1}^{T}\sum_{i=1}^{m} \|\nabla F(\boldsymbol{x}_t^{(i)})\|^2] = \mathcal{O}(\frac{1}{\sqrt{T}} + \frac{\tau^2}{(1-\alpha)^2 T}) \tag{2.14}$$

我们注意到，尽管 2.10和 2.14的左手边略有不同，但它们的物理含义是相同的。这种微小的差异是由于在推导这些界限时使用了不同的技术。与 2.10相比，我们看到由于压缩而产生的近似差距，由 $\alpha$ 捕获，现在 2.14的第二项中。当 $T$ 足够大时，我们现在的收敛率为 $\mathcal{O}(\frac{1}{\sqrt{T}})$，这就避免了方差爆炸的问题。

请注意，由于我们要求 $0 \leq \alpha < 1$，我们这里的分析对 2.7中的随机-$k$ 压缩器不成立。2.7中的随机-$k$ 压缩器，但我们可以修改 2.7，去掉放大系数 $d/k$，因为我们不再要求无偏性了。所得的得到的压缩器满足 $\alpha = 1 - k/d$，这与 top-$k$ 的压缩器相同。然而，在实践中，top-$k$ 通常比 random-$k$ 更有效，因为它的实际逼近的差距通常比 $1 - k/d$ 的上界小得多。

这些结果表明，错误反馈机制通常比非错误反馈机制表现更好。然而，最近有工作表明，通过以系统的方式将有偏见的压缩器转化为无偏见的压缩器，我们实际上可能获得更好的性能。这是一个活跃的研究领域，从业者可能需要试验不同的压缩技术，以了解哪种技术对手头的问题效果最好。

### 2.3.2 自适应压缩率

压缩更新的 SGD 中的一个问题是如何确定压缩率即 2.5和 2.11中的 $q$ 和 $\alpha$)，以最小化达到目标函数的某个目标值的训练时间。在这种情况下，最佳压缩率取决于每个迭代中的计算和每个回合中的通信所产生的物理时间。这个问题类似于第 6.2.1 节中讨论的确定最优局部更新数 $\tau$，但这里的控制变量是压缩率。可以采用第 6.2.1 节类似的方法来解决这个问题，即压缩率适应方法来自收敛边界。为了克服在收敛边界中估计或消除未知参数的困难，也可以使用无模型的方法，如基于在线学习的方法 [**?** ]。实质上，基于在线学习的方法采用探索-利用的方式，在最初几轮探索不同的压缩率选择，并逐渐切换到利用那些之前已经有利的压缩率。一个挑战是，探索需要有最小的开销，因为否则，即使与没有优化的情况相比，它也会延长训练时间。

为了促进有效的探索，可以制定一个问题来寻找最佳的压缩率，使训练时间最小化，以减少单位数量的经验风险 [**?** ]。这个问题的确切目标是未知的，因为很难预测在使用不同的压缩率时训练将如何进展。然而，经验证据表明，对于一个给定的（当前）经验风险，我们可以假设之前使用的压缩率与未来经验风险的进展无关。再加上其他一些假设，我们可以把这个问题放在一个在线凸优化（OCO）框架 [**?** ] 中，它可以用在线梯度下降法解决，梯度是单位风险降低的训练时间相对于压缩率的导数。注意，这里的这个梯

度与学习问题的梯度不同。然后，在线梯度下降程序是在每一轮的训练时间目标上使用梯度下降来更新压缩率，不同的轮次可以有不同的目标，这些目标是事先未知的。理论上可以证明，尽管我们只对每一轮的目标进行梯度下降，但累积的最优性差距（称为遗憾）在时间上呈亚线性增长，因此，当时间变为无穷大时，时间平均的遗憾会归于零。然而，这种方法需要一个梯度神谕，以每轮选择的压缩率给出准确的导数，这在实践中是很难得到的。

为了克服这个问题，中使用了一种基于符号的在线梯度下降方法 [? ]，它只根据导数的符号而不是实际值来更新压缩率。估计导数的符号相对容易，只要估计正确符号的概率高于估计错误符号的概率，就可以保证有类似的亚线性遗憾。经验结果表明，这种算法能迅速收敛到一个接近最佳的压缩率，并比选择一个任意固定的压缩率提高性能。

### 2.3.3 模型剪枝

除了压缩参数更新外，模型本身也可以通过剪枝（去除）神经网络中一些不重要的权重来压缩。这既加快了计算和通信的速度，又保持了最终模型的类似精度 [? ? ]。图 2.2显示了剪枝的一个例子。一个著名的剪枝方法是迭代训练和修剪模型，在多次 SGD 迭代的时间间隔内，删除一定比例的小幅度权重。当把剪枝和联邦学习结合起来时，可以使用一个两阶段的程序，


图 2.2: 模型剪枝示意图

即在第一阶段对单方进行模型训练和剪枝，然后在涉及多方的常规联合学习过程中进一步剪枝。最初的修剪阶段允许联合学习从一个小的模型开始，与从完整的模型开始相比，可以节省计算和通信，同时随着模型及其权重在进一步修剪阶段的调整，仍然收敛到全局最优。为了确定哪些权重应该被修剪（或在第二阶段加回），可以制定一个目标，使修剪后的模型接近于原始模型，并在未来几轮中保持” 可训练性”。为了接近原始模型，可以采用标准的基于幅度的修剪，并适当选择修剪率，这样只有那些幅度足够小的权重可以被修剪掉。当从修剪后的模型中执行一步 SGD 时，可以使用经验风险降

低的一阶近似值来捕获可训练性。基于这个近似值，我们可以求出应该修剪的权重集（如果之前已经修剪过，则可以加回来）以保持可训练性。总的来说，这种方法随着时间的推移调整模型的大小，以（近似）最大化训练效率。

## 2.4 讨论

在这一章中，我们回顾了在联邦学习中使用的具有通信高效的分布式优化算法，特别是减少通信频率的本地更新 SGD 算法和减少通信比特数的压缩方法。这些方法可以与其他算法相结合，提高联邦学习的收敛速度和效率。例如，边缘方可以使用加速 [**?** ]、方差缩减 [**? ?** ] 或自适应优化方法，而不是使用经典的 SGD 作为本地求解器。

## 参考文献

[1] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[2] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[3] Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM review*, 60(2):223–311, 2018.

[4] Ofer Dekel, Ran Gilad-Bachrach, Ohad Shamir, and Lin Xiao. Optimal distributed online prediction using mini-batches. *Journal of Machine Learning Research*, 13(1), 2012.

[5] Mu Li, Tong Zhang, Yuqiang Chen, and Alexander J Smola. Efficient mini-batch training for stochastic optimization. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 661–670, 2014.

[6] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[7] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[8] Dong Yin, Ashwin Pananjady, Max Lam, Dimitris Papailiopoulos, Kannan Ramchandran, and Peter Bartlett. Gradient diversity: a key ingredient for scalable distributed learning. In Amos Storkey and Fernando Perez-Cruz, editors, *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 1998–2007. PMLR, 09–11 Apr 2018. URL `https://proceedings.mlr.press/v84/yin18a.html`.

[9] Pratik Chaudhari and Stefano Soatto. Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks, 2018.

[10] Behnam Neyshabur, Ryota Tomioka, Ruslan Salakhutdinov, and Nathan Srebro. Geometry of optimization and implicit regularization in deep learning, 2017.

[11] Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information, 2017.

[12] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021.

[13] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

[14] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, et al. Large scale distributed deep networks. *Advances in neural information processing systems*, 25, 2012.

[15] Henggang Cui, James Cipar, Qirong Ho, Jin Kyu Kim, Seunghak Lee, Abhimanu Kumar, Jinliang Wei, Wei Dai, Gregory R Ganger, Phillip B Gibbons, et al. Exploiting bounded staleness to speed up

big data analytics. In *2014 {USENIX} Annual Technical Conference ({USENIX}{ATC} 14)*, pages 37–48, 2014.

[16] Sanghamitra Dutta, Gauri Joshi, Soumyadip Ghosh, Parijat Dube, and Priya Nagpurkar. Slow and stale gradients can win the race: Error-runtime trade-offs in distributed sgd, 2018.

[17] Suyog Gupta, Wei Zhang, and Fei Wang. Model accuracy and runtime tradeoff in distributed deep learning: A systematic study. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 171–180. IEEE, 2016.

[18] Jian Zhang and Ioannis Mitliagkas. Yellowfin and the art of momentum tuning. *arXiv preprint arXiv:1706.03471*, 2017.

[19] Sebastian U Stich. Local sgd converges fast and communicates little. *arXiv preprint arXiv:1805.09767*, 2018.

[20] Jianyu Wang and Gauri Joshi. Cooperative sgd: A unified framework for the design and analysis of communication-efficient sgd algorithms, 2019.

[21] Hao Yu, Sen Yang, and Shenghuo Zhu. Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5693–5700, 2019.

[22] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. Qsgd: Communication-efficient sgd via gradient quantization and encoding. *Advances in neural information processing systems*, 30, 2017.

[23] Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Stich, and Martin Jaggi. Error feedback fixes signsgd and other gradient compression schemes. In *International Conference on Machine Learning*, pages 3252–3261. PMLR, 2019.

[24] Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified sgd with memory. *Advances in Neural Information Processing Systems*, 31, 2018.

# 第三章　高效通信模型融合


## 摘要

我们考虑在通信轮数受到严重限制的情况下，学习一个联合模型的问题。我们讨论了最近关于模型融合的工作，这是联合学习的一个特例，只允许有一个通信回合。这种设置有一个独特的特点，即客户只要有一个预先训练好的模型，但没有数据就足够了。像 GDPR 这样的数据存储规定使这种设置很有吸引力，因为在 FL 开始之前更新本地模型后，数据可以立即被删除。然而，模型融合方法仅限于相对较浅的神经网络架构。我们讨论了适用于深度学习模型的模型融合的扩展，这些模型需要超过一轮的通信，但在通信预算方面仍然非常有效，即通信轮数和客户端与服务器之间交换的消息大小。我们考虑了同质和异质的客户端数据场景，包括由于数据中的偏差，在聚合数据上的训练是次优的场景。除了深度学习方法外，我们还涵盖了无监督的设置，如混合模型、主题模型和隐马尔科夫模型。

我们将模型融合的统计效率与假设的集中式方法进行比较，在这种方法中，一个拥有无限计算和存储能力的学习者简单地聚合所有客户的数据并以非联合的方式训练一个模型。正如我们将看到的，尽管模型融合方法通常与（假设的）集中式方法的收敛率相匹配，但它可能没有同样的效率。此外，当客户数据是异质的时候，集中式和联合式方法之间的这种差异会被放大。


## 3.1　导言

标准的联合学习算法，例如联合平均法 [34]，在聚合来自客户端的模型参数时，依赖于简单的参数平均法。由于其简单性，这种方法与大多数模型和深度学习架构兼容。然而，它也有一些缺点。特别是，在许多应用领域，训练一个高性能模型所需的通信回合数通常是数百个。这种通信成本可能

是令人望而却步的，特别是在通信轮次的开销很高的情况下。例如，在客户端是移动设备的应用中，一轮通信可能对应于每天与服务器的同步任务。在其他应用中，启动一轮通信可能需要人类的批准（例如，医院形成一个数据链）。

上述的挑战促使我们考虑只需要几轮通信的联合学习算法。我们回顾一下最近的模型融合 [48, 49] 技术，它只需要单轮通信。模型融合方法有一个额外的好处，即客户不需要存储数据，也就是说，他们只需要提供他们的本地模型，用于联合学习一个更强大的全局模型。数据存储受到 GDPR[19] 的监管，使得模型融合的这一特点具有实际的吸引力。正如我们将看到的，模型融合技术依赖于通过匈牙利算法 [28] 或 Wasserstein barycenters[1, 42] 的变体进行的双点匹配 [49]。这些方法在平均参数时考虑到了模型组件（如神经元权重）之间的相似性，这使得它们能够在短短一个通信回合内产生良好的全局模型。缺点是，对于 VGG 架构等深度神经网络来说，相应的优化问题对准模型组件变得难以解决 [41]。

为了处理深层神经网络，我们也考虑了 [47] 的层级匹配策略，它可以在固定的通信回合数（取决于神经网络的深度）内训练一个强大的联合模型。然而，这种方法需要客户端存储数据，以便像其他联合学习算法那样在本地执行模型更新。

最后，我们还探讨了模型融合的统计特性。特别是，我们将模型融合的方法与一个假设的集中式方法进行比较，在这个方法中，一个服务器聚集了所有来自客户端的数据，并在没有任何通信限制的情况下训练一个 ML 模型。这在联合学习的许多应用中是不现实的，但它是一个黄金标准，我们可以将模型融合的统计性能与之进行比较。理想情况下，我们希望模型融合能够与这种（假设的）集中式方法的统计效率相匹配。正如我们将看到的，这几乎是事实。

在这一章中，我们回顾了模型融合技术，并展示了它们在单轮通信中与较简单的神经网络架构的联合学习的适用性 [49]；与混合模型、隐马尔科夫模型和主题模型的无监督学习 [48, 50]。我们提出了用于后验融合的扩展 [14]，以支持贝叶斯神经网络的联合学习 [36]。我们回顾了 [47] 的方法，该方法适用于有限的通信预算下的深度神经网络的联合学习。最后，我们研究了模型融合的统计特性，最后我们讨论了开放的挑战和有希望的未来工作方向。

## 3.2 模型的互换-不变结构

许多机器学习模型可以用参数向量集而不是参数单向量来描述。例如，一个用于聚类的混合模型是由一组聚类中心点来描述的。集合中的中心点顺序的任何变化都会产生等效的聚类质量，即相同的数据可能性。在联合学习的背景下，现在很明显，为什么简单地对从不同客户处获得的两套聚类中心点进行平均化可能是有害的：即使在最简单的情况下，两个客户在同质数据集上拟合聚类模型并恢复相同的中心点，其解决方案中中心点的排序是任意的，元素平均化可能会导致糟糕的联合全球模型。本章中我们将介绍的其他突出的包络变量无监督模型的例子是主题模型和隐马尔科夫模型。

模型参数的互换不变性也存在于监督学习中，特别是神经网络中。考虑一个简单的具有 L 个隐藏单元的单隐藏层全连接神经网络

$$f(x) = \sigma(xW_1)W_2 \tag{3.1}$$

其中 $\sigma(\cdot)$ 是一个非线性的应用，$x \in \mathbb{R}^D, W_1 \in \mathbb{R}^{D\times L}, W_2 \in \mathbb{R}^{L\times K}$。$D$ 和 $K$ 是输入和输出维度，我们省略偏置项，不失为一种通用的方法。让 $W_{1,\cdot l}$ 表示 $W_1$ 的第 $l$ 列，$W_{2,l\cdot}$ 表示 $W_2$ 的第 $l$ 行，那么我们可以将（3.1）写成

$$f(x) = \sum_{l=1}^{L} W_{2,l\cdot}\sigma(< x, W_{1,\cdot l} >) \tag{3.2}$$

和是一个不变的操作，因此任何神经元的重新排序，即 $W_1$ 的列和 $W_2$ 的相应行，都会导致一个具有相同预测规则的神经网络。为了说明互换不变性，我们将（3.1）重写为

$$f(x) = \sigma(xW_1\prod)\prod^{T} W_2 \tag{3.3}$$

其中 $\prod$ 是 $L!$ 个可能的互换矩阵之一。回顾一下，排列矩阵是一个正交矩阵，它在左边应用时作用于行，在右边应用时作用于列。假设 $\{W_1, W_2\}$ 是最佳权重，那么，根据公式（3.3），在两个同质数据集 $X_j$ 和 $X_j'$ 上训练会产生两组权重 $\left\{W_1\prod_j, \prod_j^T W_2\right\}$ 和 $\left\{W_1\prod_{j'}, \prod_{j'}^T W_2\right\}$。这两组参数的天真平均是次优的，也就是说，$\prod_j \neq \prod_{j'}$ 的概率很高，因此 $\frac{1}{2}(W_1\prod_j + W_1\prod_{j'}) \neq W_1\prod$ 为任何 $\prod$。为了优化神经网络权重的平均化，我们首先应该撤销排列组合 $(W_1\prod_j\prod_j^T + W_1\prod_{j'}\prod_{j'}^T)/2 = W_1$。

超越排列组合的神经网络不变性我们在本章将要介绍的模型融合技术是为了在进行融合时考虑到客户模型的排列组合不变性。然而，在神经网络

的情况下，可能存在其他不变性。神经网络通常是巨大的超参数化，学习相应的权重是一个非凸的优化问题，可能有许多等价的（在训练损失的意义上）局部优化。对于一个有 $L$ 个神经元的单隐层神经网络，如方程（3.1），由于包络不变性，对于任何解决方案，至少有 $L$ 个等价的解决方案。有可能存在其他不变的（或其他相等的）解决方案。这是文献中一个尚未解决的问题。一个潜在的富有成效的观点是研究神经网络的损失景观–对这个问题还没有完整的理论认识；然而，已经取得了一些进展 [15, 18, 21, 30]。考虑损失景观的角度来开发新的联合学习和模型融合算法是一个有趣的未来工作方向。

### 3.2.1 匹配平均法的一般公式

# 第四章　分离学习：分布式深度学习的一种资源节约型模型和数据并行方法


## 摘要

资源限制、工作量开销、缺乏信任和竞争阻碍了多个机构之间共享原始数据。这导致了用于训练最先进的深度学习模型的数据短缺。分裂学习是一种分布式机器学习的模型和数据并行方法，是克服这些问题的高度资源效率的解决方案。分离式学习通过对传统的深度学习模型架构进行划分，使网络中的一些层对客户是私有的，其余的在服务器上集中共享。这使得分布式机器学习模型的训练不需要任何原始数据的共享，同时减少任何客户端所需的计算或通信量。分离式学习的范式有几种变体，取决于手头正在考虑的具体问题。在本章中，我们将分享执行分割学习的理论、经验和实践方面，以及一些可以根据你选择的应用而选择的变体。


## 4.1　分离学习的介绍

联合学习是一种数据并行的方法，其中数据是分布式的，而作为训练回合一部分的每个客户端都使用自己的本地数据训练完全相同的模型架构。在现实世界中，服务器可能是一个强大的计算资源，但最终却要进行一个相对简单的计算，即对每个客户端学习的权重进行加权平均。在现实世界中，往往存在着与服务器相比资源相对有限的客户。

分离式学习通过将模型架构分割成不同的层，使每个客户保持权重，直到一个被称为分离层的中间层，从而迎合了这种现实的设置。其余的层都在

服务器上保存。

**优点和局限性。**这种方法不仅减少了任何客户端要进行的计算工作，而且还减少了分布式训练期间需要发送的通信有效载荷的大小。这是因为它只需要在前向传播步骤中从任何客户端向服务器发送来自一个层（分裂层）的激活信息。同时，在反向传播步骤中，只有一个层（分裂层之后的层）的梯度需要由服务器发送至客户端。在模型性能方面，我们根据经验观察到，SplitNN 的收敛速度仍然比联合学习和大批量同步随机梯度下降快得多。也就是说，当在较少的客户上进行训练时，它需要相对较大的整体通信带宽，尽管在有大量客户的情况下，它最终比其他方法低得多。先进的神经网络压缩方法，如，可以用来减少通信负荷。通信带宽也可以通过允许在客户端有更多的层来表示进一步压缩的表征来换取客户端的计算。

在分割学习中共享中间层的激活也与局部并行 [8]、特征重放 [9] 和分割征服量化 [10] 的分布式学习方法有关。这是与联合学习中的权重共享相对应的。

### 4.1.1　普通的分离学习

在这种方法中，每个客户端训练网络到某一层，即分裂层，并将权重发送到服务器（图 19.1）。然后服务器训练网络的其他层。这就完成了前向传播的过程。然后，服务器为最后一层生成梯度，并反向传播错误，直到分裂层。然后，梯度被传递给客户端。其余的反向传播由客户端完成。这样一直持续到网络训练完成。分割的形状可以是任意的，不一定是垂直的。在这个框架中，也没有明确的原始数据的共享。

#### 4.1.1.1　同步步骤

在每个客户端完成其历时后，下一个排队完成其历时的客户端会收到来自前一个客户端的本地权重（直到分割层的权重）作为其历时的初始化。

#### 4.1.1.2　放宽同步要求

客户端之间的这种额外的通信可以通过像 BlindLearning[11] 这样的分割学习方法来避免，该方法基于使用一个损失函数，该函数是每个客户端完成的前向传播获得的损失的平均值。同样，通过 splitFedv1[12]、splitFedv2[12] 和 splitFedv3[13] 进一步减少了通信和同步要求，这些都是分割学习和联合

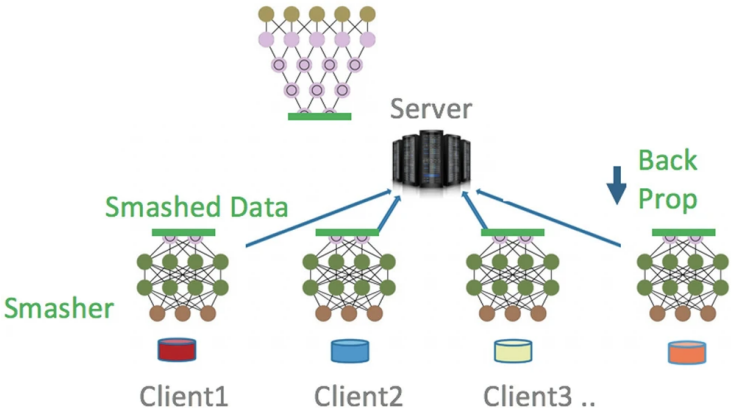



图 4.1: 有多个客户端和一个服务器的分割学习设置，绿色虚线显示了客户端的层份额和服务器的层份额之间的分割。在前向传播过程中，只有分割层（客户的最后一层）的激活被共享，在反向传播过程中，只有服务器的第一层的梯度与客户共享。

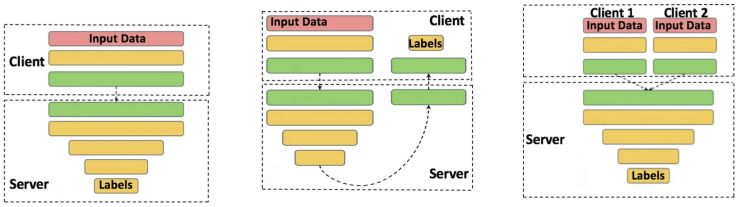
学习的混合方法。在 [14] 中提供了一种改善延迟的混合方法。

## 4.2 通信效率 [15]

在这一节中，我们描述了我们对分割学习和联合学习这两种分布式学习设置的通信效率的计算。为了分析通信效率，我们考虑了每个客户端为训练和客户端权重同步所传输的数据量，因为影响通信速率的其他因素取决于训练集群的设置，而与分布式学习设置无关。我们使用以下符号来数学地衡量通信效率。

**符号** $K = \#$ 客户，$N = \#$ 模型参数，$p =$ 总数据集大小，$q =$ 分割层大小，$\eta =$ 客户的模型参数（权重）的比例，因此 $1 - \eta$ 是服务器的参数比例。

在表 19.1 中，我们显示了每个客户端在一个历时中所需要的通信，以及所有客户端在一个历时中所需要的总通信。由于有 $K$ 个客户端，当每个客户端的训练数据集的大小相同时在分割学习中，每个客户会有 $p/K$ 的数据记录。因此，在前向传播过程中，分裂学习中每个客户端传递的激活大小为 $(p/K)q$，在后向传播过程中，每个客户端传递的梯度大小也为 $(p/K)q$。在有客户端权重共享的虚无分裂学习情况下，将权重传递给下一个客户端




(a) (b) (c)

图 4.2: 健康的分割学习配置显示原始数据不在客户端和服务器健康实体之间传输，用于用 SplitNN 对分布式深度学习模型进行训练和推理。(a) 简单的香草式分割学习。(b) 没有标签共享的分割学习。(c) 纵向分割数据的分割学习

将涉及 N 的通信。在联合学习中，在上传单个客户端权重和下载平均权重的过程中，权重/梯度的通信都是 $\eta N$。平均值的过程中，权重/梯度的通信量都是 $N$ 的大小。

## 4.3　延迟

根据客户端和服务器的计算能力限制，计算的延迟需要最小化，同时保持高的通信效率。为此，[14] 对普通分割学习、splitFed 和 [14] 中提出的方法的延迟进行了分析比较。他们考虑了以下模型大小的符号

## 4.4　分离学习的拓扑结构

### 4.4.1　多样化的配置

除了所讨论的普通式分离学习及其需要较少同步的变体外，还有其他可以使用分离学习的拓扑结构，如下文所述。

1. **无标签共享的分割学习的 U 型配置 [3] [16]:** 本节描述的另外两种配置涉及到标签的共享，尽管它们彼此之间不共享任何原始输入数据。我们可以通过一个不需要客户共享任何标签的 U 型配置来完全缓解这个问题。在这个设置中，我们在服务器网络的末端层将网络包裹起来，并将输出送回客户实体，如图 19.2b 所示。虽然服务器仍然保留着它的大部分层，但客户端从末端层产生梯度，并将其用于反向传播，而不

共享相应的标签。在标签包括高度敏感信息的情况下，如病人的疾病状况，这种设置是分布式深度学习的理想选择。

2. **分割学习的垂直分区数据 [17]:** 这种配置允许持有不同模式的病人数据的多个机构学习分布式模型，而无需共享数据。在图 19.2c 中，我们展示了一个适合这种多模式多机构合作的 SplitNN 的配置实例。作为一个具体的例子，我们介绍了放射科中心与病理测试中心和疾病诊断的服务器合作的情况。如图 19.2c 所示，持有成像数据模式的放射科中心训练一个部分模型，直到分裂层。以同样的方式，拥有病人测试结果的病理测试中心训练一个部分模型，直到它自己的分割层。然后，来自这两个中心的分割层的输出被串联起来，并被发送到疾病诊断服务器，以训练模型的其余部分。这个过程来回继续，完成前向和后向传播，以训练分布式深度学习模型，而不分享彼此的原始数据。

3. **扩展的香草分割学习:** 如图 19.3a 所示，我们给出了香草分割学习的另一个修改方案，即连接输出的结果在传递给服务器之前在另一个客户端进一步处理。

4. **多任务分割学习的配置:** 如图 19.3b 所示，在这个配置中，来自不同客户的多模式数据被用来训练部分网络，直到其相应的分割层。每个分割层的输出都被串联起来，然后发送到多个服务器。每个服务器使用这些数据来训练多个模型，以解决不同的监督学习任务。

5. **类似 Tor[18] 的配置，用于多跳分割学习:** 这种配置是 vanilla 配置的一个类似的扩展。在这种情况下，多个客户端依次训练部分网络，每个客户端最多训练一个分割层，并将其输出转移到下一个客户端。这个过程继续进行，如图 19.3c 所示，最终客户将其激活从其分割层发送到服务器以完成训练。

我们想指出的是，尽管这些例子的配置显示了 SplitNN 的一些多功能应用，但它们绝不是唯一可能的配置。

### 4.4.2 用 ExpertMatcher 选择模型 [19]

在某些情况下，一个强大的服务器托管着多个专有模型的存储库，它希望通过预测 API 在机器学习即服务（MLaaS）的商业模式中使用。这些专有模型不能被提供给客户下载。同时，客户往往有敏感的数据集，它希望获

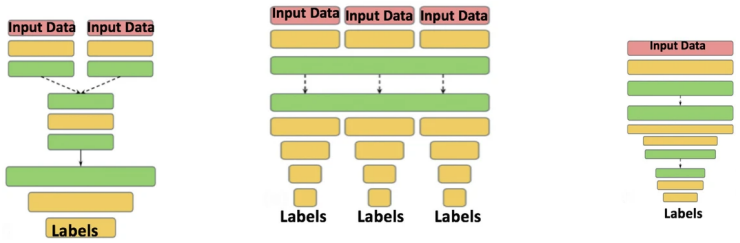



图 4.3: 健康的分割学习配置显示原始数据不在客户端和服务器健康实体之间传输，用于用 SplitNN 进行分布式深度学习模型的训练和推理。(a) 扩展的香草。(b) 垂直分割输入的多任务输出。(c) 'Tor' [18]

得预测结果。在这种情况下，出现了一个问题，即如何从服务器的存储库中找到与客户持有的数据集相匹配的正确模型。ExpertMatcher 就是这样一个基于 U 型回旋镖分割学习拓扑结构的模型选择架构。

### 4.4.3　实现细节

我们假设在集中式服务器上有 $K$ 个预训练的专家网络，每个网络都有其相应的预训练的无监督表征学习模型（本例中我们考虑自动编码器（AE））$\phi_K$ 在特定任务的数据集上训练过。考虑到 AE 所训练的数据集，我们提取整个数据集的编码表征，并计算出数据集的平均表征 $\mu_k \in \mathbb{R}d, k \in \{1, \cdots, K\}$，其中 $d$ 是特征维度。假设数据集由 $N$ 个对象类别组成，我们也计算出数据集中每个类别的平均代表性 $\mu_k n \in \mathbb{R}^d, n \in \{1, \cdots, N\}, k \in \{1, \cdots, K\}$。

客户端（客户 A 和客户 B）利用与服务器类似的方法，客户为每个 $p$ 和 $q$ 的数据集训练他们独特的 AE，客户 A：$p \in \{1 \cdots, P\}$ 和客户端 B：$q \in \{1 \cdots, Q\}$。让我们假设对于客户 A 来说，从隐藏层提取的样本 $X_p^1$ 的中间特征给定为 $x_p^1 = \phi_p^1(X_p^1)$，同样，对于客户 B 来说，它是 $x_q^2 = \phi_q^2(X_q^2)$。为了简洁起见，我们把来自任何客户端的中间表征表示为 $x^{'}$。

我们首先要解释一下标签的粗细概念，我们的意思是，数据中的类是由高层次（粗）或低层次（细）的语义类别分开的。举例来说，把狗和猫分开的类是粗的类别，而把不同类型的狗分开的类是细的类别。在 ExpertMatcher 的概念中，对于客户数据的粗放式分配（CA）。对于编码后的表示 $x'$，我们分配一个服务器 AE，$k^* \in \{1, \cdots, K\}$，该服务器与 $x^{'}$ 具有最大的相似度 $\mu_k$；见图 19.4。

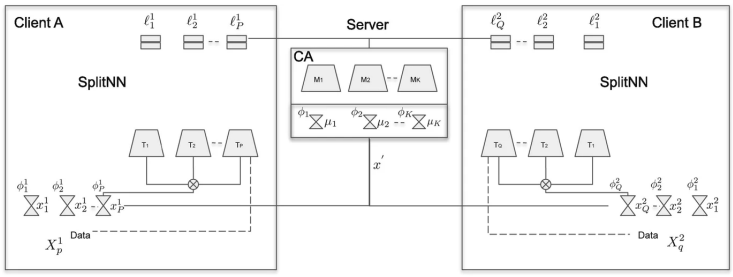



图 4.4: pass

对于客户数据的细粒度（FA）分配: 对于编码后的表示 $x^{'}$，我们分配一个专家网络 $M_n$ $n \in \{1, \cdots N\}$，该网络具有 $x^{'}_{k^*}$ 与 $\mu^n_k$ 的最大相似度。用于分配的相似性的选择取决于用户。余弦相似性、距离相关性、信息论测量、希尔伯特-施密特独立准则、最大平均差异、内核目标对齐和积分概率指标只是相似性指标的几种可能性。

最后在给定的样本分配到模型后，人们可以很容易地训练一个 SplitNN 类型的架构 [3]。

在目前的设置中，由于服务器不能接触到客户的原始数据，而是一个非常低维的编码表示，因此保留了一个弱水平的隐私。

请注意，这种方法有一个不足之处。如果服务器没有专门用于客户数据的 AE 模型，客户数据就会因为最大余弦相似性准则而发生错误分配–这可以通过在服务器上增加一个额外的模型来解决，该模型执行二元分类：客户数据与服务器数据匹配或不匹配。

## 4.5 分离学习的协作推理

由于企业能够利用大规模的计算资源在巨大的数据集上训练超大型机器学习模型，这为打算用这些模型进行预测的外部客户带来了一系列新的问题。鉴于这些大型模型通常有数十亿个参数，客户不愿意在设备上完整地下载这些模型。使用这些模型进行预测，仅在设备上进行计算资源密集。这就开启了私人协作推理（PCI）的问题，在这个问题上，模型被分割到客户端和服务器上（表 19.3）。

客户的数据是私有的，因此在这种情况下交流的激活需要被正式私有化，以防止成员推理和重建攻击。在另一种情况下，服务器打算私下分享训

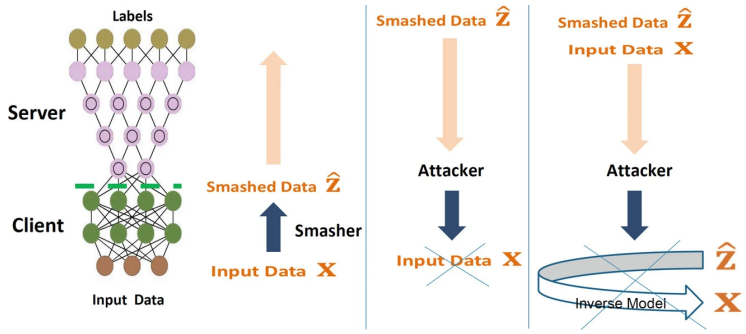



图 4.5: pass

练好的模型的权重，这类工作已经相当多了。在这种 PCI 的设置中，考虑的隐私是关于服务器自己的数据的。而 PCI 的设置是相对较新的，因为它要求在私人推理期间对客户自己的私人数据进行私人共享激活，而不是在私人训练后对服务器的数据进行私人共享权重。这需要在基于激活共享而非权重共享的分布式机器学习和正式隐私的交叉点上进行创新。

### 4.5.1　防止协作推理中的重构攻击

需要获得预测的客户的数据记录是私有的，因此，在这种 PCI 设置中交流的模型的中间表征（或激活）需要被脱敏，以防止重建攻击（图 19.5）。从架构的角度来看，保护隐私的机器学习还没有达到其 AlexNet 时刻。该领域在 DP-SGD[24] 及其变体等正式的隐私机制方面取得了快速的进展。在这些方法中，仍然有很大的空间来改进目前的隐私与效用的权衡，使其适用于许多生产用例。我们现在描述激活共享方面的一些进展，用于 (a) 防止训练数据方面的成员推理攻击和 (b) 防止 PCI 设置中的预测查询数据的重建攻击。

#### 4.5.1.1　信道剪枝

[20] 中的工作表明，学习一个修剪滤波器来选择性地修剪掉分裂层潜伏表征空间中的通道，有助于在 PCI 设置的预测步骤中凭经验防止各种先进的重建攻击（图 19.6）。** 图 19.6** 参考文献 [20] 显示，学习一个修剪滤波器来选择性地修剪掉分裂层潜伏表征空间中的通道，有助于在 PCI 设置的预测步骤中凭经验防止各种最先进的重建攻击。

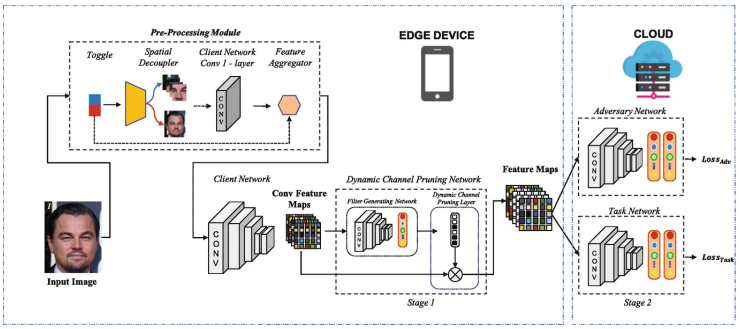



图 4.6: pass

#### 4.5.1.2 相关性

这里的关键思想是通过在常用的分类损失项–分类交叉熵上增加一个额外的损失项来减少信息泄露。我们使用的减少信息泄漏的损失项是距离相关，这是随机变量之间非线性（和线性）统计依赖性的有力措施。距离相关损失在原始输入数据和任何选定的层的输出之间最小化，这些层的输出需要从客户端传达给另一个不受信任的客户端或不受信任的服务器。这种设置对一些流行的分布式机器学习形式至关重要，这些机器学习需要共享中间层的激活。这已经在动机部分的" 激活共享" 小节中得到了激励。

这种损失组合的优化有助于确保由保护层产生的激活有最小的信息来重建原始数据，同时在后处理时仍有足够的作用来达到合理的分类精度。实验部分从质量和数量上证实了在保持合理分类精度的同时防止重建原始输入数据的质量。距离相关性与交叉熵的联合最小化导致了一种专门的特征提取或转换，从而使其在人类视觉系统和更复杂的重建攻击方面无法察觉原始数据集的信息泄露。

#### 4.5.1.3 损失函数

输入数据 $X$ 的 $n$ 个样本、保护层 $Z$ 的激活、真实标签 $Y_{true}$、预测标签 $Y$ 和标量权重 $\alpha$ 的总损失函数由以下公式给出:

$$\alpha DCOR(X,Z) + (1-\alpha)CCE(Y_{true},Y) \tag{4.1}$$

### 4.5.2　激活共享的差异性隐私

Arachchige 等人 [21] 提供了一个差分隐隐私机制，用于分享卷积层和池化层之后得到的扁平化层的激活值。这些扁平化的输出被二进制化，一个受 RAPPOR 启发的" 效用增强的随机化" 机制被应用来创建一个差分隐私的二进制表示。然后，这些数据被传送到服务器，在那里全连接层对它们进行操作以产生最终的预测结果。[22] 中的工作为从深度网络中提取的特征的监督流形嵌入提供了一种差分隐私机制，以便从服务器上的数据库执行图像检索任务。[23] 的工作着眼于防止在分离学习的背景下标签信息的泄漏。它们提供了防止规范和暗示攻击泄露标签信息的防御措施。

## 4.6　未来工作

关于分布式机器学习方法，如分离学习和联邦学习，有几个方面需要研究。这些问题包括资源效率、隐私、收敛性、现实世界数据的非均匀性、训练的延迟、协作推理、滞后的客户端、通信的拓扑结构、攻击测试平台等等，使这一领域成为当前研究的活跃领域。