成绩_____

苏州大学

课　　程　　文献阅读和科技写作

学院(部)　　计算机科学与技术学院

类　　型　　专题翻译

学　　号　　1927405160

姓　　名　　张昊

日　　期　　2022 年 4 月 23 日

资源受限的边缘计算中通信高效的异步联邦学习

1927405160 张昊

(苏州大学 计算机科学与技术学院，江苏 苏州 215006)

摘　要 在边缘计算中，联邦学习被广泛应用于基于海量数据训练机器学习模型．然而现有的联邦学习解决方案可能会导致较长的训练时间和（或）较高的资源（如带宽）消耗，因此不能直接应用于资源受限的边缘节点，如基站和接入点．在本文中，我们提出了一种新的通信高效的异步联邦学习（Communication-Efficient Asynchronous Federated Learning，CE-AFL）机制．在该机制中，参数服务器将只利用所有边缘节点中占比α的部分（$0<\alpha<1$）来聚合本地模型的更新，并按其在每轮迭代中的到达顺序进行聚合．作为研究案例，我们提出了一种有效的算法来确定 CE-AFL 在单个学习任务和多个学习任务两种情况中α的最优值．我们正式地证明了所提出算法的收敛性．我们利用 Jetson TX2、深度学习工作站和大量仿真实验，评估了算法的性能．在经典模型和数据集上的实验和仿真结果都表明本文提出的机制和算法是有效的．例如，与最先进的解决方案相比，CE-AFL 可以在达到近似精度的情况下减少约 69%的训练时间，并且在资源约束下训练的模型精度提高了约 18%．

关键词 联邦学习；边缘计算；通信高效；异步

1 引言

随着物联网的快速发展，物理世界中每天都有大量的数据产生[1][2]．在传统的解决方案中，这些数据会通过核心网络转发到云端进行训练或处理，但这样做会消耗大量的带宽．因此，将数据在本地处理和将更多的计算任务交给边缘节点（称为边缘计算[3]）变得越来越有吸引力．这推动了在网络边缘实现分布式机器学习，即联邦学习（Federated Learning，FL）[4][5][6]的应用．

如图 1 所示，一个联邦学习系统通常由一个或多个参数服务器（服务器组）和大量计算节点（如边缘节点）组成，遵循典型的参数服务器架构[7]．每个参数服务器由一个管理器控制，并维护全局共享参数的分区．简单起见，我们假设只有一个参数服务器，这种解决方案可以很容易地推广到多个参数服务器的情况．每个计算节点负责通过训练本地数据来计算本地的统计数据（如梯度），并且只与参数服务器通信．具体来说，计算节点向参数服务器发送本地更新，并从参数服务器接收更新的全局模型．由于计算节点只将训练好的模型暴露给参数服务器，而不是参与训练的数据，因此联邦学习可以有效地保护用户的隐私．

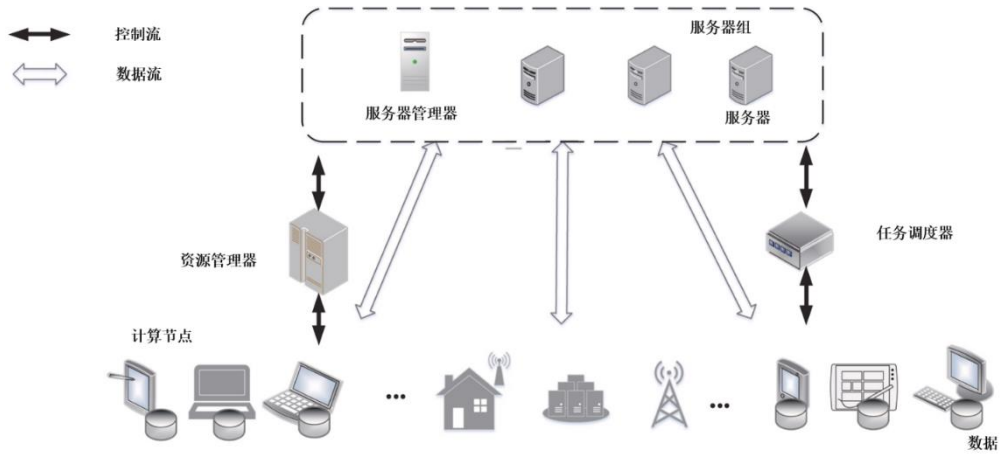

图 1 典型的参数服务器架构图

为了在边缘计算中实现高效的联邦学习，我们需要考虑以下约束和因素.

- **资源约束**：边缘节点和远程参数服务器之间的带宽通常受到限制[8]，然而边缘节点可能会频繁地转发和接收更新的模型，这需要庞大的带宽开销[9]. 例如 AlexNet 模型中参数的大小约为 60M[10]. 考虑 1GB 的带宽，由于本地和全局模型的频繁传输，网络很容易拥塞. 同时，与云平台相比，边缘节点通常是资源受限的，如计算能力、内存大小等.
- **数据不平衡性**：由于设备的移动性（如车辆网络），每个边缘节点将处理来自各种设备的不同的数据，从而导致边缘节点之间的数据不平衡. 例如，文献[11]的作者表明，在一天的时间内，不同边缘节点上的数据量可能从 10GB 到 1TB 不等.
- **边缘不确定性**：由于边缘节点通常部署在室外，一些节点偶尔会出现系统崩溃、电池电量不足、网络中断等故障[12].

在边缘计算中有两种主要的联邦学习方案. 第一种方案是同步方案[13][14][15]. 具体来说，参数服务器一旦从所有指定的边缘节点（或计算节点）接收到了训练后的（本地）模型，就将会聚合这些本地模型，并将更新后的全局模型分发给每轮迭代的所有边缘节点. 这种方案能够保证分布式算法和集中式算法[16]是等价的，但是也带来了两个主要的缺点. 第一个缺点是每轮迭代的训练时间主要取决于所有边缘节点的最大训练时间. 由于数据不平衡性、节点的异质性（heterogeneity，如 CPU 容量、内存大小），以及各种网络连接（4G、5G 以及 Wi-Fi），不同边缘节点的训练时间可能会有很大的差异，称为掉队者（straggler）[17]. 这会导致模型训练时间更长甚至难以接受. 第二个缺点是在同步方案中，所有边缘节点更新后的本地模型会被频繁地转发到参数服务器进行聚合，从而消耗了大量的网络带宽.

第二种方案是在边缘进行异步联邦学习[18][19][20]. 这种方案允许部分（而不是全部）

计算节点将更新后的模型转发给参数服务器，以便在每轮迭代中进行模型聚合．由于该方案不需要参数服务器等待全部边缘节点的本地更新，因此与同步方案相比，该方案能较好地处理数据不平衡性和边缘动态问题．但是这些工作忽略了有限的资源对训练性能的影响，可能需要更多的迭代轮次，或者每轮迭代需要更多的计算节点参与，从而导致大量的带宽消耗或更长的训练时间．

为了更好地应对上述约束和因素，我们提出了一种通信高效的异步联邦学习（Communication-Efficient Asynchronous Federated Learning，CE-AFL）机制．这一机制由参数服务器根据每轮迭代中所有边缘节点的到达顺序，利用所有边缘节点中占比α的部分来对更新后的模型进行聚合．需要注意的是，在全局聚合中涉及的本地更新的子集将在不同的迭代轮次发生变化，这将在 2.3 节中说明．根据 2.4 节的理论分析，训练的性能取决于α的值，因此如何根据资源约束条件确定α的最优值是一个关键问题．本文的主要贡献有：

- 我们设计了一种通信高效的异步联邦学习（Communication-Efficient Asynchronous Federated Learning，CE-AFL）边缘计算机制，并正式地证明了 CE-AFL 的收敛性．
- 作为研究案例，我们提出了一种有效的算法来确定 CE-AFL 在单个学习任务和多个学习任务两种情况中α的最优值，从而在带宽约束下花费更少的训练时间．我们同样证明了该算法的收敛性．
- 在经典模型和数据集上广泛的实验表明了我们提出的机制和算法是有效的．具体来说，与最先进的解决方案相比，我们的 CE-AFL 机制可以减少约 69%的训练时间，并且在资源受限的情况下训练的模型精度提高了约 18%．

本文的其余部分组织如下．第 2 节介绍了一些准备工作，提出了联邦学习机制，给出了收敛性分析，并对问题进行了形式化描述；第 3 节中提出了两种针对单个学习任务和多个学习任务的有效算法；第 4 节展示了我们的模拟和实验的结果；第 5 节讨论了相关工作；第 6 节对本文做出总结．

2 准备工作

在这一节中，我们介绍了联邦学习的概念（0 节），并提出了新的联邦学习机制（2.2 节）．在 2.3 节中，我们通过一个例子来说明 CE-AFL．然后，我们给出 CE-AFL 的收敛性分析（2.4 节），并对问题进行了形式化描述（2.5 节）．为了便于描述，表 1 中列出了一些关键的符号．

表 1 关键符号

符号	含义
V	边缘节点集合
Γ_i	边缘节点v_i上的本地数据集
D	本地更新的迭代次数
T	训练结束前总共的训练轮次
$\mathbb{T}$	转置运算
$\mathcal{K}$	资源类别的数量
η	学习率
n	边缘节点的数量
g_k	边缘节点的本地更新中资源k的消耗
b_k	在服务器和计算节点间通信中资源k的消耗
B_k	每类资源k的总预算
L	学习任务的集合
α_j	将参与服务器上学习任务j全局聚合的一定比例的本地更新
Φ	α_j的集合，其中$j \in \{1,2,\cdots,L\}$
$\widehat{w}^D$	D次本地更新后的模型参数
β_i^t	边缘节点v_i的本地更新是否参与迭代轮次t
$F(w^T)$	T轮迭代后的全局损失函数
$F(w^*)$	损失函数$F(w)$的最优值

2.1 联邦学习

2.1.1 联邦学习的目标

联邦学习可以在资源受限的边缘计算中利用分布式数据集训练全局模型．理想情况下，来自不同用户或设备的训练数据可以改进机器学习模型的表示和泛化能力[6]．每个边缘节点作为计算节点，使用私有数据在本地训练模型，同时参数服务器汇总计算节点的本地更新，并将全局更新的模型发送给计算节点．为了保护用户的隐私，计算节点不会将其训练数据暴露给通常位于云端的参数服务器[4]，而只公开已经训练完毕的本地模型．此外，联邦学习还可以处理边缘计算中大规模分布的非独立相同分布（non-Independently Identically Distribution，非 I.I.D.）来训练数据[21].

方便起见，给定训练样本q_i，其损失函数表示为f_i．于是，$\mathcal{N}$个训练样本的损失函数可以表示为$f(w)=\frac{1}{\mathcal{N}}\sum_{i=1}^{\mathcal{N}} f_i(w)$，其中$w$表示模型的参数向量．因此，联邦学习的目标可以表示为$\min_{w\in R^m} f(w)$，其中R^m表示m维实数空间．我们注意到，这一问题结构可以包含简单的模型，例如线性或逻辑回归[22]、支持向量机（Support Vector Machines，SVM）[23]，以及更复杂的模型，例如条件随机场模型或神经网络[10]．对于$\mathcal{N}$个输入输出对$\{x_i,y_i\}_{i=1}^{\mathcal{N}}$，$x_i \in R^m$，$y_i \in R$或者$y_i \in \{-1,1\}$，$f_i$的一些典型例子包括：

- 线性回归：$f_i=\frac{1}{2}\left(x_i^{\mathbb{T}}w-y_i\right)^2$, $y_i \in R$

- 逻辑回归：$f_i = -\log\left(1+\exp(-y_i x_i^{\mathbb{T}} w)\right),\ y_i \in \{-1,1\}$
- 支持向量机：$f_i = \max\{0, 1-y_i x_i^{\mathbb{T}} w\},\ y_i \in \{-1,1\}$

其中$\mathbb{T}$表示向量的转置．神经网络中会涉及到更复杂的非凸问题，通过特征向量x_i的非凸函数进行预测，而不是通过线性特征映射$x_i^{\mathbb{T}} w$进行预测．事实上，由此产生的损失函数仍然可以写成$f_i(w)$．

2.1.2 联邦学习的优化算法

有两种基本算法可以优化联邦学习的损失函数．

梯度下降（Gradient Descent，GD）[25]是经典的一阶方法．GD 的基本思想是在当前状态下最小化目标函数的一阶泰勒展开，从而近似优化目标函数本身．具体来说，对于损失函数f，在当前状态ω下可以解决如下问题：

$$\min_w f(\omega) \approx \min_w f(w_t) + \nabla f(w_t)^T (w - w_t). \tag{1}$$

式（1）的右半部分相对于自变量ω是线性的，并最小化$\nabla f(w_t)^T (w - w_t)$，这与梯度$\nabla f(w_t)$的方向相反．因此，梯度下降的更新规则如下：

$$w_{t+1} = w_t - \eta \nabla f(w_t) \tag{2}$$

其中η是步长，也称为学习率．

随机梯度下降（Stochastic Gradient Descent，SGD）[26]用于随机抽样训练数据，其更新公式为：

$$w_{t+1} = w_t - \eta_t \nabla f_{i_t}(w_t) \tag{3}$$

其中i_t表示第t次迭代中随机抽样的数据标签．随机抽样数据获得的梯度是所有数据对梯度的无偏估计，即$\mathbb{E}_{i_t} \nabla f_{i_t}(w_t) = \nabla f(w_t)$．此外，由于在每个训练轮次仅随机选择一个样本，计算量将大大减少．通过比较两种算法，本文采用了 SGD．这是因为 SGD 是 GD 的天然替代品，并可以大大提高学习效率，减少抽样数据的计算时间．

2.2 通信高效的异步联邦学习

假设边缘计算系统中有一组边缘节点$\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$，其中$|\mathcal{V}| = n > 2$．每个边缘节点在本地数据集$\Gamma_i$上训练模型，其大小为$|\Gamma_i|,\ i \in \{1,2,\cdots,n\}$．对于每个节点$v_i$，本地数据集$\Gamma_i$上的损失函数是

$$F_i(w) = \frac{1}{|\Gamma_i|} \sum_{q_j \in \Gamma_i} f_j(w). \tag{4}$$

在本节中，我们提出了通信高效的异步联邦学习（Communication-Efficient Asynchronous

Federated Learning，CE-AFL）机制，并使用算法 1 形式化描述．与同步和异步方案类似，CE-AFL 也由多轮迭代训练组成．设变量T表示训练结束前总共的训练轮次．每轮迭代将执行一次全局更新（模型聚合）．假设连续两次全局更新之间有$D(\geq 1)$次本地更新（迭代）．设$\alpha \in \{\frac{1}{n},\frac{2}{n},\cdots,1\}$是来自所有边缘节点的本地模型更新的一部分，用于每轮迭代$t \in \{1,\cdots,T\}$中在参数服务器上聚合全局模型．在一个迭代轮次中，一旦收到来自任意$\alpha \cdot n$个计算节点的本地更新的模型，参数服务器将按其到达顺序执行模型聚合，并计算得到更新后的全局模型（第 3~7 行）．然后，参数服务器将全局模型分发给已经向服务器发送本地更新的计算节点（第 8 行），并更新资源预算（第 9 行）．在边缘节点一侧（第 11~15 行），一旦接收到全局模型，计算节点将为本地模型执行几次更新，然后将更新的本地模型推送到参数服务器进行模型聚合．在T轮迭代之后，训练模型的全局损失函数$F(w^T)$为

$$F(w^T)=\frac{\sum_{i=1}^{n}\sum_{q_j\in\Gamma_i}\beta_i^T f_j(w^T)}{\sum_{i=1}^{n}\beta_i^T|\Gamma_i|}=\frac{\sum_{i=1}^{n}\beta_i^T|\Gamma_i|F_i(w^T)}{\sum_{i=1}^{n}\beta_i^T|\Gamma_i|} \tag{5}$$

其中β_i^t是一个二元变量，用于指示边缘节点v_i的本地更新是否参与了第t轮迭代．因此，它满足$\sum_{i=1}^{n}\beta_i^t=\alpha\cdot n,\ \forall t\in\{1,\cdots,T\}$．整个训练过程将一直持续到违背资源限制或者达到全局收敛．也就是说，$F(w^t)-F(w^*)<\varepsilon$，其中$\varepsilon$是任意的小正值，$w^*$是损失函数$F(w)$的最优值．

算法 1.　通信高效的异步联邦学习（CE-AFL）．

输入：训练轮次T．

输出：模型w和损失函数$F(w)$．

对于 每轮迭代$t\in\{1,2,\cdots,T\}$：

参数服务器进程

如果 资源约束满足：

当 收到的本地更新数量 $<\alpha\cdot n$：

等待计算节点的本地更新．

根据到达顺序聚合本地模型．

使用公式（5）计算全局损失．

向计算节点分发更新的全局模型参数w为每个资源类别k更新预算B_k．

边缘节点v_i进程

当 已向服务器发送本地更新 **并且** 没有收到全局模型：

等待更新的全局模型．

对于 每次本地更新$d\in\{1,2,\cdots,D\}$：

更新本地模型：$w_{t+1}=\omega_t-\eta_t\nabla f_{i_t}(w_t)$．

向服务器推送本地更新．

返回 最终模型w和损失函数$F(w)$．

2.3 CE-AFL的例子

我们在图 2 中举了一个例子，以更好地说明 CE-AFL．假设边缘计算系统中有一个参数服务器和十个计算节点．由于某些原因（如不可用），现有的工作[13][14][15]通常选择部分计算节点来完成模型训练任务．因此，选择了四个计算节点（编号为 1~4）来参加本次的模型训练任务．此图展示了在固定长度的时间段内同步方案和 CE-AFL 的本地（浅灰色）和全局（深橙色）更新．对于左图中的同步方案，只有在参数服务器收到来自四个计算节点的所有本地更新后，才会执行模型聚合以计算得到更新的全局模型．当计算节点收到全局模型时，他们将继续使用本地数据进行训练．在左图中，同步方案只有两次全局模型更新．

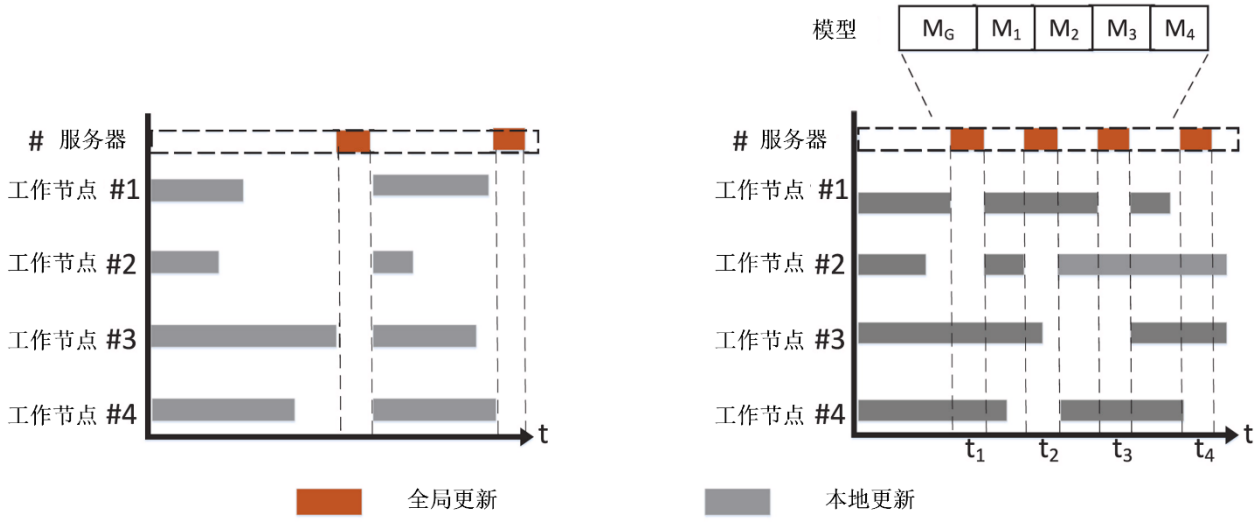

图 2 同步联邦学习和 CE-AFL．不同的颜色（浅灰色和深橙色）分别表示本地和全局更新．给定时间长度，同步方案（左图）和 CE-AFL（右图）有 2 个和 4 个全局更新．

在 CE-AFL 中，假设$\alpha = \frac{1}{2}$．换言之，一旦接收到了来自任意两个计算节点的本地更新，参数服务器将执行模型聚合，正如图 2 中右图所示．实际情况中，计算节点的数据和资源（如计算能力和带宽预算）总是时时变化的．因此，参与全局更新的本地更新子集将在不同迭代轮次有所不同．例如，第一轮迭代中参数服务器聚合了来自第 1、2 个计算节点的本地更新，而第二轮迭代中是来自第 2、4 个计算节点的．需要注意的是，如果服务器在当前聚合期间收到了本地更新的模型，那么其将在下一次全局更新中聚合．在给定固定的时间段中，CE-AFL 中有四个全局更新（右图），而同步方案中只有两个全局更新（左图）．因此在相同的时间预算约束下，CE-AFL 将会执行更多的全局更新，并且收敛速度比同步方案更快．

注意到我们提出的 CE-AFL 机制可能会遇到另一个问题——延迟更新．例如，当第 3 个计算节点首次将其本地更新模型发送到服务器进行全局模型聚合时，服务器在时间点t_1和t_2聚合了第 1、2、4 个计算节点的本地更新模型．我们采用延迟补偿机制[27]来缓解这个问题．在

图 2 中，我们使用M_G表示当前全局模型，使用$M_i, \forall i \in \{1, \cdots, 4\}$表示来自第$i$个计算节点最新的本地更新模型．这些模型将记录在服务器上，以便对过时的模型进行延迟补偿．例如，考虑介于t_2和t_3的一个时间点t，第 1 个计算节点只向服务器发送了一次本地更新模型，而服务器执行了两次全局模型聚合．那么，第 1 个计算节点的陈旧度为全局更新的数量和本地模型更新的数量之间的差距，如这里是$2-1=1$．在服务器两次收到第 1 个计算节点的本地模型后，模型M_1将使用衰变系数$\varsigma\ (0<\varsigma<1)$更新，即$M_1=\varsigma^x \cdot M_1+(1-\varsigma^x)\cdot M_G$，其中$x$表示第 1 个计算节点的陈旧度．通过这种方法可以减轻过时的模型带来的影响．由于如何确定系数ς不是本文的重点，我们根据文献[27]在评估中设置了ς的值．

此外，我们举了一个例子来展示我们提出的解决方案如何解决边缘不确定性问题．如图 3 所示，由于系统崩溃或网络断开，参数服务器无法接收第 4 个计算节点的本地更新．因此，同步方案中没有全局模型聚合（左图）．参数服务器在第一轮迭代中收到第 1、2 个计算节点（$\alpha=\frac{1}{2}$）的本地更新后执行全局更新，在第二轮迭代中收到第 2、3 个计算节点的本地更新后执行全局更新．尽管我们还没有收到第 4 个计算节点的本地模型更新，但 CE-AFL（右图）仍然有三次全局更新．因此，我们提出的解决方案可以有效地解决边缘不确定性问题．

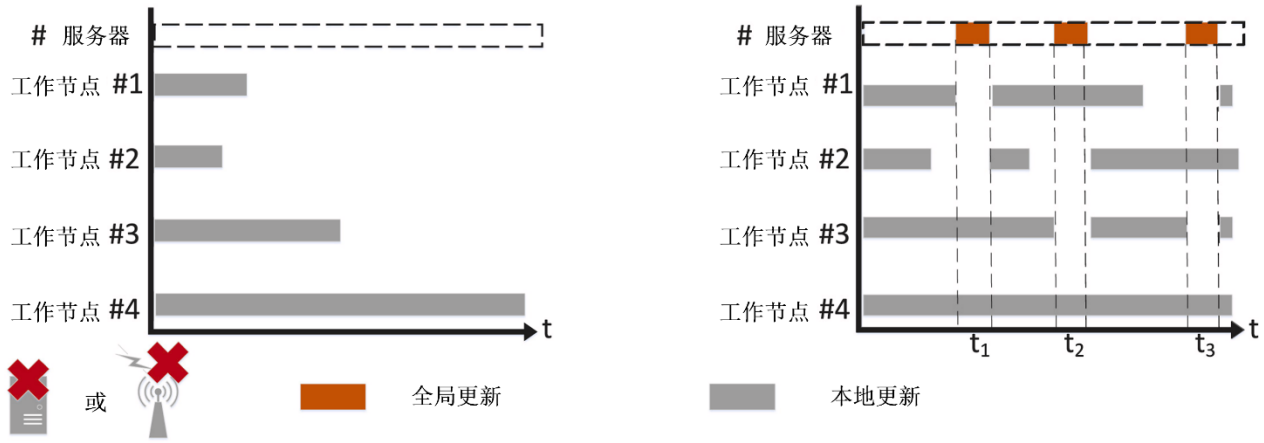

图 3　CE-AFL 如何处理边缘不确定性．左图：同步方案；右图：CE-AFL．

2.4 收敛性分析

为了分析我们提出的模型训练机制的可行性，我们证明了 CE-AFL 可以达到一个恒定的收敛界限．我们首先做出以下三个假设[20]．

假设 1（光滑性）．令$L>0$．若对于$\forall w_1, w_2$，

$$f(w_2)-f(w_1)\geq\langle\nabla f(w_1), w_2-w_1\rangle+\frac{L}{2}||w_2-w_1||^2 \tag{6}$$

则损失函数f关于模型参数是 L-光滑的．

假设 2（强凸性）．令$\mu \geq 0$．若对于$\forall w_1, w_2$，

$$f(w_2) - f(w_1) \geq \langle \nabla f(w_1), w_2 - w_1 \rangle + \frac{\mu}{2}||w_2 - w_1||^2 \quad (7)$$

则损失函数f关于模型参数是μ-强凸的．

注意到，若$\mu = 0$，则f是凸函数，即

$$f(w_2) - f(w_1) \geq \langle \nabla f(w_1), w_2 - w_1 \rangle. \quad (8)$$

这个假设可以满足具有凸函数的模型，例如线性回归和 SVM．

假设 3（有界梯度方差）．每个边缘节点随机梯度的方差是有界的：

$$\mathbb{E}||\nabla f(w;\, q) - \nabla f(w)||^2 \leq \mathcal{P},\ \forall w \in R^m,\ q \in \Gamma_i,\ i \in \{1, \cdots, n\}$$

其中$\mathcal{P}$是一个正数．

假设 4（全局最优存在性）．假设至少存在一个达到损失函数$f(w)$的全局最小的解，记为w^*．

CE-AFL 将执行T轮迭代，直至达到全局收敛．在每轮迭代中，边缘节点将执行D次本地更新．我们首先推导了每次本地更新的收敛边界（定理 5）．然后我们证明了每轮迭代t模型训练的收敛性和T轮迭代后的收敛边界（定理 6）．

定理 5．假设全局损失函数F是 L-光滑的和μ-强凸的，每个计算节点在向参数服务器报告更新的模型之前执行D次本地更新．当满足以下三个条件时：

- $\eta < \frac{1}{L}$
- $\eta\mu > 1 - \sqrt[D]{\frac{1}{4n}}$
- $F(w^0) - F(w^*) > \frac{D\eta\mathcal{P}}{4(1-\eta\mu)^D}$

我们有

$$\begin{cases} 2\alpha n(1-\eta\mu)^D \in (0,1) \\ \mathbb{E}[F(\widehat{w}^D) - F(w^*)] \leq (1-\eta\mu)^D[F(w^0) - F(w^*)] + \dfrac{D\eta\mathcal{P}}{2} \end{cases}$$

其中$\widehat{w}^D$表示D次本地更新后的模型参数，w^0表示初始模型参数．

证明．我们首先考虑D次本地更新的收敛性．对于每个本地更新$d \in \{1, \cdots, D\}$，根据平滑性和强凸性的假设，我们有

$$\begin{aligned}
\mathbb{E}[F(\widehat{w}^{d})-F(w^{*})] &\leq F(\widehat{w}^{d-1})-F(w^{*})-\eta\mathbb{E}[\langle\nabla F(\widehat{w}^{d-1}),\nabla f(\widehat{w}^{d-1};q_{d})\rangle] \\
&\quad+\frac{L\eta^{2}}{2}\mathbb{E}[||\nabla f(\widehat{w}^{d-1};q_{d})||^{2}] \\
&\leq F(\widehat{w}^{d-1})-F(w^{*})-\frac{\eta}{2}||\nabla F(\widehat{w}^{d-1})||^{2} \\
&\quad+\frac{\eta}{2}\mathbb{E}[||\nabla F(\widehat{w}^{d-1})-\nabla f(\widehat{w}^{d-1};q_{d})||^{2}] \\
&\leq F(\widehat{w}^{d-1})-F(w^{*})-\frac{\eta}{2}||\nabla F(\widehat{w}^{d-1})||^{2}+\frac{\eta\mathcal{P}}{2} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\triangleright\ \mathbb{E}||\nabla f(w;\,q)-\nabla f(w)||^{2}\leq\mathcal{P} \\
&\leq F(\widehat{w}^{d-1})-F(w^{*})-\eta\mu[F(\widehat{w}^{d-1})-F(w^{*})]+\frac{\eta\mathcal{P}}{2} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\triangleright\ F(w)\leq F(w^{*})+\frac{1}{2\mu}||\nabla F(w)||^{2},\forall x \\
&\leq(1-\eta\mu)[F(\widehat{w}^{d-1})-F(w^{*})]+\frac{\eta\mathcal{P}}{2}. \qquad (9)
\end{aligned}$$

我们推导了每个本地更新$d\in\{1,\cdots,D\}$的收敛结果. 通过放缩并求总期望，在D次本地更新后，我们有

$$\begin{aligned}
\mathbb{E}[F(\widehat{w}^{D})-F(w^{*})] &\leq(1-\eta\mu)[F(\widehat{w}^{D-1})-F(w^{*})]+\frac{\eta\mathcal{P}}{2} \\
&\leq(1-\eta\mu)\left[(1-\eta\mu)[F(\widehat{w}^{D-2})-F(w^{*})]+\frac{\eta\mathcal{P}}{2}\right]+\frac{\eta\mathcal{P}}{2} \\
&\cdots \qquad\qquad (按式（9）放缩) \\
&\leq(1-\eta\mu)^{D}[F(w^{0})-F(w^{*})]+\frac{\eta\mathcal{P}}{2}\sum_{d=1}^{D}(1-\eta\mu)^{d-1} \\
&\leq(1-\eta\mu)^{D}[F(w^{0})-F(w^{*})]+\frac{\eta\mathcal{P}}{2}\frac{1-(1-\eta\mu)^{D}}{1-(1-\eta\mu)} \\
&\leq(1-\eta\mu)^{D}[F(w^{0})-F(w^{*})]+\frac{\eta\mathcal{P}}{2}\frac{D\eta\mu}{1-(1-\eta\mu)} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\triangleright\ \eta\mu<1,\ 1-(1-\eta\mu)^{D}<D\eta\mu \\
&\leq(1-\eta\mu)^{D}[F(w^{0})-F(w^{*})]+\frac{D\eta\mathcal{P}}{2}. \qquad (10)
\end{aligned}$$

在参数服务器端，第t次迭代收到$\alpha\cdot n$个更新的模型后将执行全局模型聚合. 它遵循$w^{t}=\frac{1}{\alpha n}\sum_{i=1}^{\alpha n}\widehat{w}_{i}^{D}$，其中$\widehat{w}_{i}^{D}$表示计算节点$v_{i}$在$D$次本地更新后的本地更新模型.

定理 6．在 T 轮迭代后，CE-AFL 的收敛边界为

$$\mathbb{E}[F(w^T)-F(w^*)] \le \tau[F(w^0)-F(w^*)]+(1-\tau)\frac{D\eta\mathcal{P}}{4\varphi}$$

其中$\varphi=(1-\eta\mu)^D$，$\tau=(2\alpha n\varphi)^T$.

证明．根据w^t的定义，在第$t\in\{1,\cdots,T\}$次迭代，对于每一次全局更新，我们有

$$\begin{aligned}
\mathbb{E}[F(w^t)-F(w^*)] &\le \frac{1}{\alpha n}\sum_{i=1}^{\alpha n}[F(\widehat{w}_i^D)-F(w^*)] \\
&\le \frac{1}{\alpha n}\sum_{i=1}^{\alpha n}\left[(1-\eta\mu)^D\left[F\left(w_i{}^0\right)-F(w^*)\right]+\frac{D\eta\mathcal{P}}{2}\right] \\
&\le \alpha n\left[(1-\eta\mu)^D\left[F\left(w^0\right)-F(w^*)\right]+\frac{D\eta\mathcal{P}}{2}\right] \\
&\le \alpha n\left[(1-\eta\mu)^D\left[F\left(w^0\right)-F\left(w^{t-1}\right)+F\left(w^{t-1}\right)-F(w^*)\right]+\frac{D\eta\mathcal{P}}{2}\right] \\
&\le \alpha n(1-\eta\mu)^D\left(F\left(w^0\right)-F\left(w^{t-1}\right)\right)+\alpha n(1-\eta\mu)^D\left(F\left(w^{t-1}\right)-F(w^*)\right) \\
&\quad +\frac{\alpha nD\eta\mathcal{P}}{2} \\
&\le \alpha n(1-\eta\mu)^D\left(F\left(w^{t-1}\right)-F(w^*)\right)+\alpha n(1-\eta\mu)^D\left(F\left(w^{t-1}\right)-F(w^*)\right) \\
&\quad +\alpha\frac{nD\eta\mathcal{P}}{2} \\
&\le (2\alpha n(1-\eta\mu)^D)[F(w^{t-1})-F(w^*)]+\alpha\frac{nD\eta\mathcal{P}}{2}. \qquad (11)
\end{aligned}$$

则 T 轮训练后的收敛边界可经过如下推导得到

$$\begin{aligned}
\mathbb{E}[F(w^T)-F(w^*)] &\le (2\alpha n(1-\eta\mu)^D)[F(w^{T-1})-F(w^*)]+\alpha\frac{nD\eta\mathcal{P}}{2} \\
&\cdots \qquad \text{（按式（11）放缩）} \\
&\le (2\alpha n(1-\eta\mu)^D)^T[F(w^0)-F(w^*)] \\
&\quad +\alpha\frac{nD\eta\mathcal{P}}{2}\frac{1-(2\alpha n(1-\eta\mu)^D)^T}{1-(2\alpha n(1-\eta\mu)^D)} \\
&\quad \triangleright\ \eta\mu>1-\sqrt[D]{1/4n},\ 2\alpha n(1-\eta\mu)^D<\frac{1}{2} \\
&\le (2\alpha n(1-\eta\mu)^D)^T[F(w^0)-F(w^*)] \\
&\quad +\frac{(1-(2\alpha n(1-\eta\mu)^D)^T)\alpha nD\eta\mathcal{P}}{4\alpha n(1-\eta\mu)^D}
\end{aligned}$$

$$\leq (2\alpha n(1-\eta\mu)^D)^T[F(w^0)-F(w^*)] + \frac{D\eta\mathcal{P}}{4(1-\eta\mu)^D}(1-(2\alpha n(1-\eta\mu)^D)^T). \quad (12)$$

为简单起见，设$\varphi=(1-\eta\mu)^D$．我们将上式写成：

$$\mathbb{E}[F(w^T)-F(w^*)] \leq \tau[F(w^0)-F(w^*)] + (1-\tau)\frac{D\eta\mathcal{P}}{4\varphi} \quad (13)$$

其中$\tau=(2\alpha n\varphi)^T$．■

我们注意到收敛边界（最优差距）$F(w^T)-F(w^*)$是与α的值和迭代轮数T有关的不等式（13）．此外，当α和T都变大时，可以减小最优差距，但这需要大量的资源成本．因此，在资源约束下，如何确定α和T的最优值来优化训练性能（如训练时间）是一个挑战．

2.5 问题定义

我们定义了具有资源约束的异步联邦学习（Asynchronous Federated Learning with Resource Constraints，AFL-RC）问题．具体来说，我们将为一个学习任务确定α和T的值，以便在资源约束下最小化训练时间．模型训练过程中需要消耗几类资源（如网络带宽、CPU 等），包括计算节点上的本地更新、计算节点与参数服务器之间的模型交换、参数服务器上的全局更新．这里我们主要考虑的是静态场景，即边缘节点是固定的，如基站或监控摄像头等．我们将在第 6 节讨论动态场景的问题．假设有$\mathcal{K}$种完全不同类别的资源．设g_k表示表示在边缘节点上进行本地更新时消耗的资源$k\in\{1,2,\cdots,\mathcal{K}\}$．同时，$b_k$表示在边缘节点和参数服务器之间进行一次模型交换时消耗的资源k．一旦更新了全局模型，服务器就会将更新后的模型分发给所有计算节点．因此，在T轮迭代后，对于每个资源类别k，所有节点的本地更新和全局更新的总资源消耗分别为$T\cdot n\cdot g_k$和$T\cdot(\alpha+1)\cdot n\cdot b_k$．设$B_k$表示每个资源类别$k$的总预算．因此，我们将 AFL-RC 问题表述如下：

$$\min_{T\in\{1,2,3,\cdots\}} F(w^T) \quad s.t.\begin{cases} T\cdot n\cdot[g_k+2\alpha\cdot b_k]\leq B_k,\ \forall k \\ \alpha\in\{\frac{1}{n},\frac{2}{n},\cdots,1\} \end{cases} \quad (14)$$

第一组不等式表示每个资源类别$k\in\{1,2,\cdots,\mathcal{K}\}$在整个$T$轮训练中的约束．第二组等式表示变量$\alpha$的界．AFL-RC 问题的目标是最小化全局损失函数$F(w^T)$．

由式（14）中的第一组约束，我们发现两个参数α和T是相关的．为简单起见，我们考虑如何在边界计算中确定两种情况下α的最优值．一种情况是只有一个学习任务．另一种是更普遍的情况，即有多个学习任务．然后，我们可以相应地确定参数T的值．

3 一个带宽优化的研究案例

在本章节中，我们设计了高效的算法来确定α的取值，以便在带宽约束下以较少训练时间达到收敛．这里，我们主要将边缘计算中的通信瓶颈——带宽资源[8]作为最重要的资源消耗．我们首先在 3.1 节考虑只有单次训练任务的场景，在该场景中我们准确地获得了进行参数聚合的α的最优值．然后，我们在 3.2 节考虑多任务模型训练的场景．为了确定每个任务的α值，我们提出了基于序列二次规划的比例分配（Sequence Quadratic Program based Proportion Assignment，SQP-PA）算法来解决这个问题，并证明了该算法的全局收敛性．

3.1 单个学习任务的算法

我们考虑边缘计算中只有单个学习任务的简单情况．式（14）的目标等价于最小化$F(w^T)-F(w^*)$．此外，我们使用式（13）中的上界作为$F(w^T)-F(w^*)$的近似，得出新的目标如下：

$$\min_{\alpha\in\{\frac{1}{n},\cdots,1\}} (2\alpha n\varphi)^T[F(w^0)-F(w^*)]+[1-(2\alpha n\varphi)^T]\frac{D\eta\mathcal{P}}{4\varphi}.$$

由定理 5 可知，由于$2\alpha n\varphi<1$，不难得到目标函数随T的增大而减小．根据式（14）中的第一个不等式，T的最优解为$\left\lfloor \min_k \frac{B_k}{n\cdot[g_k+(\alpha+1)\cdot b_k]} \right\rfloor$．为了简化分析，我们忽略取整运算，将$T\approx \min_k \frac{B_k}{n\cdot[g_k+(\alpha+1)\cdot b_k]}= \max_k \frac{n\cdot[g_k+(\alpha+1)\cdot b_k]}{B_k}$代入目标函数．在这种情况下，本地更新不需要带宽开销，也就是说$\forall k\in\{1,2,\cdots,\mathcal{K}\}, g_k=0$．设$B$表示学习任务的带宽约束．根据上述近似，我们将 AFL-RC 问题重新表述为

$$\min_{\alpha\in\{\frac{1}{n},\cdots,1\}} H(\alpha) \qquad s.t.\begin{cases}2\alpha\cdot n\cdot b\cdot T\le B\\ \alpha\in\{\frac{1}{n},\frac{2}{n},\cdots,1\}\end{cases} \tag{15}$$

其中

$$H(\alpha)=(2\alpha n\varphi)^{\frac{(1+\alpha)nb}{B}}[F(w^0)-F(w^*)]+\left[1-(2\alpha n\varphi)^{\frac{(1+\alpha)nb}{B}}\right]\frac{D\eta\mathcal{P}}{4\varphi}. \tag{16}$$

我们可以推导出最优解$\alpha^*=\mathrm{argmin}_{\alpha\in\{\frac{1}{n},\frac{2}{n},\cdots,1\}}H(\alpha)$.

定理 7．在当有足够的带宽资源，即$B\to\infty$时，在模型训练过程中总能达到收敛边界．

证明．若$B\to\infty$，则有$\frac{(1+\alpha)nb}{B}\to 0$．相应地，我们可以推出$(2\alpha n\varphi)^{\frac{(1+\alpha)nb}{B}}\approx 1$．结合等式（16），$H(\alpha)\approx F(w^0)-F(w^*)$为常数．因此，无论$\alpha$的值是多少都可以达到收敛． ■

为了确定函数H在带宽资源约束下α的最优值，我们考虑函数$e^{G(\alpha)}=(2\alpha n\varphi)^{\frac{(1+\alpha)nb}{B}}=e^{\frac{(1+\alpha)nb}{B}\ln(2\alpha n\varphi)}$的单调性，其中$G(\alpha)=\frac{(1+\alpha)nb}{B}\ln(2\alpha n\varphi)$. 由于指数函数$e^x$是单调递增的，我们可以通过求解目标函数$G$很容易地得到最优值. 我们根据$G'(\alpha)$构造一个辅助函数$h(\alpha)=\frac{nb}{B}\left[\ln(2\alpha n\varphi)+\frac{1}{\alpha}+1\right]$. 当$\alpha\in\left[\frac{1}{n},1\right]$时，$h'(\alpha)=\frac{nb}{B}\left(\frac{1}{\alpha}-\frac{1}{\alpha^2}\right)<0$. 因此$h(\alpha)$关于$\alpha$单调递减. 接下来假设满足定理 5 中的两个条件. 我们考虑函数$h(\alpha)$的三种情况.

定理 8. 假设在$\alpha\in\left[\frac{1}{n},1\right]$内$h(\alpha)>0$. 若$\varphi>\frac{e^{-2}}{2n}$，我们有$\alpha^*=\frac{1}{n}$.

证明. 为了证明该定理，我们首先将α松弛到一个连续的区间$\alpha\in\left[\frac{1}{n},1\right]$. 然后采用随机四舍五入方法[28]得到$\alpha\in\{\frac{1}{n},\frac{2}{n},\cdots,1\}$.

由于$\varphi>\frac{e^{-2}}{2n}$，我们有

$$h(1)=\frac{nb}{B}[\ln(2n\varphi)+2]>0. \tag{17}$$

此外，在$\alpha\in\left[\frac{1}{n},1\right]$内$h(\alpha)$单调递减，则有

$$h(\alpha)=\frac{nb}{B}\left[\ln(2\alpha n\varphi)+\frac{1}{\alpha}+1\right]>0\Longrightarrow G'(\alpha)>0. \tag{18}$$

因此，函数G的最小值是$G\left(\frac{1}{n}\right)$. 综上所述，当$\varphi>\frac{e^{-2}}{2n}$时，$\alpha^*=\frac{1}{n}$. ■

定理 9. 假设在$\alpha\in\left[\frac{1}{n},1\right]$内$h(\alpha)<0$. 若$\varphi<\frac{e^{-(n+1)}}{2}$，我们有$\alpha^*=1$.

证明. 由于该定理的证明类似于定理 8，我们在这里省略详细的证明.

定理 10. 假设对$\exists\alpha\in\left(\frac{1}{n},1\right)$有$h(\alpha)=0$. 若$\frac{e^{-(n+1)}}{2}\le\varphi\le\frac{e^{-2}}{2n}$，我们有$\alpha^*\approx\sqrt[3]{-\frac{\mathcal{E}}{2}+\mathcal{Q}}+\sqrt[3]{-\frac{\mathcal{E}}{2}-\mathcal{Q}}$，其中$\mathcal{Q}=\sqrt{\left(\frac{\mathcal{E}}{2}\right)^2+\left(\frac{\mathcal{F}}{3}\right)^3}$，$\mathcal{E}=\frac{-27n\varphi-23}{54n^3\varphi^3}$，$\mathcal{F}=-\frac{13}{12n^2\varphi^2}$.

证明. 由于$\alpha^*\in\left(\frac{1}{n},1\right)$，我们对$\ln(2\alpha n\varphi)$在$\alpha=\frac{1}{2n\varphi}$处进行泰勒近似展开[29]，只关注前两项展开，即一阶展开和二阶展开. 结合这两项扩展，我们有

$$h(\alpha)=\frac{nb}{B}\left[\ln(2\alpha n\varphi)+\frac{1}{\alpha}+1\right]$$

$$\approx \frac{nb}{B}\left[-2n^2\varphi^2\alpha^2+4n\varphi\alpha+\frac{1}{\alpha}-\frac{1}{2}\right]=\tilde{h}(\alpha)=0. \quad (19)$$

然后，我们推出α'的值，使得$\tilde{h}(\alpha')=0$. 对于$\exists\alpha$，若$\frac{1}{n}\le\alpha\le\alpha'$, $\tilde{h}(\alpha)\le 0$; $\alpha'\le\alpha\le 1$, $\tilde{h}(\alpha)\ge 0$. 因此，我们推出$G(\alpha^*)=G(\alpha')$，且$\alpha^*=\alpha'\approx\sqrt[3]{-\frac{\mathcal{E}}{2}+\mathcal{Q}}+\sqrt[3]{-\frac{\mathcal{E}}{2}-\mathcal{Q}}$，其中$\mathcal{Q}=\sqrt{\left(\frac{\mathcal{E}}{2}\right)^2+\left(\frac{\mathcal{F}}{3}\right)^3}$, $\mathcal{E}=\frac{-27n\varphi-23}{54n^3\varphi^3}$, $\mathcal{F}=-\frac{13}{12n^2\varphi^2}$. ■

3.2 多个学习任务的算法

在许多实际场景中，分布式边缘节点上通常存在多个同时进行的学习任务，如机器翻译、人脸识别、语音识别等[30][31][32]. 在本文中，我们将重点关注多个独立学习任务，而多依赖学习任务的情况将作为我们未来工作的一部分.

不失一般性，我们只考虑问题定义中的带宽资源约束. 在边缘计算中训练多个学习任务$L=\{l_1,\cdots,l_m\}$时，我们将为每个任务l_j确定参数α_j的最优值，以使资源约束下所有学习任务的最大损失函数最小化. 直观地说，我们期望将每个任务l_j在T轮迭代之后的损失函数降到最低，即$\min F\left(w_j^T\right)$. 对于任务l_j，目标函数$\min F\left(w_j^T\right)$相当于减少在T轮迭代后的损失$F\left(w_j^T\right)$和最优的损失$F\left(w_j^*\right)$之间（即$F\left(w_j^T\right)-F\left(w_j^*\right)$）的差距. 此外，我们使用式（13）中的上界$\mathbb{E}\left[F\left(w_j^T\right)-F\left(w_j^*\right)\right]\le\left(2\alpha_j n\varphi\right)^{T_j}\left[F_j(w^0)-F_j(w^*)\right]+\left(1-\left(2\alpha_j n\varphi\right)^{T_j}\right)\frac{D_j\eta\mathcal{P}}{4\varphi}$作为$F\left(w_j^T\right)-F\left(w_j^*\right)$的近似. 因此，学习任务$l_j$的新目标是$\mathbf{F}(j)=\left(2\alpha_j n\varphi\right)^{T_j}\left[F_j(w^0)-F_j(w^*)\right]+\left(1-\left(2\alpha_j n\varphi\right)^{T_j}\right)\frac{D_j\eta\mathcal{P}}{4\varphi}$. 需要注意的是，我们应该最小化资源约束下所有学习任务的最大损失函数，即$\min_{\alpha_{j'}\in\{\frac{1}{n},\cdots,1\}}\mathbf{F}(j')$，其中$j'=\operatorname{argmax}_{j\in\{1,2,\cdots,m\}}\mathbf{F}(j')$. 因此，问题可以描述如下：

$$\min_{\alpha_{j'}\in\{\frac{1}{n},\cdots,1\}}\ \mathbf{F}(j') \qquad s.t.\begin{cases}\sum_{j\in\{1,2,\cdots,m\}} 2\alpha_j\cdot n\cdot b^j\cdot T_j\le B\\ \alpha_j\in\{\frac{1}{n},\cdots,1\} \qquad\qquad \forall j\end{cases} \quad (20)$$

其中b^j表示将学习任务l_j的全局更新转发到边缘节点的带宽成本，B表示总带宽预算. 设Φ为α_j集合，$\forall j\in I=\{1,\ldots,m\}$. 简便起见，我们用$\mathcal{M}_j(\Phi)$表示任务$l_j$的目标函数，描述为$\min_\Phi\theta(\Phi)=\max_{j\in I}\mathcal{M}_j(\Phi)$. 此外，我们用$\mathcal{A}(\Phi)$表示式（20）中第一组不等式的左边部分. 设$\mathcal{F}_t$表示第$t$轮迭代中第二组等式的左边部分. 则式（20）可以转化为光滑约束优化问题：

$$\min \lambda \qquad s.t. \begin{cases} \mathcal{M}_j(\Phi) \le \lambda, & \forall j, \Phi \\ \mathcal{A}(\Phi) \le B, & \forall \Phi \\ \mathcal{F}_t = \Phi \cdot n, & \forall \Phi, t \end{cases} \tag{21}$$

由于目标函数$\theta(\Phi)$的不可微性，直接用经典梯度法求解式（21）中的这类优化问题是困难的．本文提出了一种基于序列二次规划的比例分配（Sequence Quadratic Program based Proportion Assignment，SQP-PA）算法．为了便于描述，我们定义$I(\Phi) = \{j | \mathcal{M}_j(\Phi) = \theta(\Phi)\}$，$j_r = \min\{j: j \in I(\Phi^r)\}$，其中$\Phi^r$表示第$r$轮搜索的比例分配．我们用$\Theta^m$来表示$(\alpha_1, \cdots, \alpha_m)^{\mathbb{T}}$．

完整的 SQP-PA 算法在算法 2 中进行了正式描述．算法开始时，SQP-PA 初始化了一些变量（第 2~3 行）．Φ的初始值由α的m个随机值组成．与文献[33]类似，我们设$\gamma \in (0, \frac{1}{2})$，$\pi \in (2,3)$，令$\mathcal{D}^r$表示第$r$轮中的搜索方向．我们首先计算搜索方向（第 5~10 行）．在Φ^r上通过解下面的二次问题来计算向量$(\lambda_r, \mathcal{D}^r)$：

$$\min \lambda + \frac{1}{2}\mathcal{D}^{\mathbb{T}} E_r \mathcal{D} \tag{22}$$

服从于$\mathcal{M}_j(\Phi^r) - \theta(\Phi^r) + \nabla\mathcal{M}_j(\Phi^r)^{\mathbb{T}}\mathcal{D} \le \lambda, \forall j$且$\mathcal{A}(\Phi) \le B', \mathcal{F}_t = \Phi^r \cdot n, \forall t$，其中$E_r \in \Theta^{m \times m}$为第$r$轮的二维矩阵．我们计算辅助变量$\bar{\mathcal{M}}_{j_r}, \bar{\mathcal{M}}_{j,j_r}$和矩阵$U$如下：

$$\begin{cases} \bar{\mathcal{M}}_{j,j_r}(\Phi^r) = \mathcal{M}_j(\Phi^r) - \mathcal{M}_j(\Phi^r), j \in J_r \setminus \{j_r\} \\ \bar{\mathcal{M}}_{j_r}(\Phi^r) = \{\bar{\mathcal{M}}_{j,j_r}(\Phi^r), j \in J_r \setminus \{j_r\}\} \\ U = \nabla\bar{\mathcal{M}}_{j_r}(\Phi^r) = \left(\nabla\mathcal{M}_j(\Phi^r) - \nabla\mathcal{M}_{j_r}(\Phi^r)\right) \end{cases} \tag{23}$$

其中$J_r = \{j \mid \mathcal{M}_j(\Phi^r) + \nabla\mathcal{M}_j(\Phi^r)^{\mathbb{T}}\mathcal{D}^r - \theta(\Phi^r) - \lambda_r = 0\}$．然后，我们开始非单调行搜索（第 11 行），计算步长为δ_r，其为序列$\{1, \frac{1}{2}, \frac{1}{4}, \cdots\}$中满足

$$\mathcal{M}\left(\Phi^r + \delta\mathcal{D}^r + \delta^2\tilde{\mathcal{D}}^r\right) \le \mathcal{M}(\Phi^r) - \gamma\delta(\mathcal{D}^r)^{\mathbb{T}} E_r \mathcal{D}^r \tag{24}$$

的第一个数．最后，更新参数和资源预算（第 12~16 行）．

算法 2.　基于序列二次规划的比例分配（SQP based Proportion Assignment，SQP-PA）．

输入：无．

输出：最小化的λ．

第 1 步．初始化．

令$\Phi^0 \in \Theta^m$，$E_0 \in \Theta^{m \times m}$，$B' = B$．

初始化对称正定矩阵U．

第 2 步．计算搜索方向．

按式（22）计算$(\lambda_r, \mathcal{D}^r)$．

若 $(\lambda_r, \mathcal{D}^r) = (0,0)$，**则** 停止．

按式（23）计算$\bar{\mathcal{M}}_{j_r}, \bar{\mathcal{M}}_{j,j_r}, U$．

若 矩阵U是满秩的，**则** 通过计算$U^{\mathbb{T}}s = -||\mathcal{D}^r||^{\pi}q - \bar{\mathcal{M}}_{j_r}(\Phi^r + \mathcal{D}^r)$获得$\tilde{s}^r$，其中$q = (1, \cdots, 1)^{\mathbb{T}}$．

若 $||\tilde{s}^r|| > ||\mathcal{D}^r||$ **或者** 矩阵U不是满秩的，**则** $\widetilde{\mathcal{D}}^r = 0$；

否则 $\widetilde{\mathcal{D}}^r = \tilde{s}^r$.

第 3 步. 开始非单调线路搜索.

第 4 步. 执行更新.

计算一个新的对称定正矩阵E_{r+1}.

$\Phi^{r+1} = \Phi^r + \delta_r \mathcal{D}^r + \delta_r^2 \widetilde{\mathcal{D}}^r, r = r + 1$.

更新资源预算B'.

若 $B' \le 0$，**则** 停止；**否则** 回到 **第 2 步**.

3.2.1 SQP-PA 的全局收敛性

在对 SQP-PA 进行收敛性分析之前，我们做了以下三个假设[33].

假设 11. $\forall j \in I, \mathcal{M}_j(\Phi)$是连续可微的.

假设 12. $\forall \Phi \in \Theta^m, \left(\nabla \mathcal{M}_j(\Phi), -1\right)^{\mathbb{T}}$是线性无关的.

假设 13. 存在两个常量$0 < \epsilon \le \rho$使得$\epsilon||\mathcal{D}||^2 \le \mathcal{D}^{\mathbb{T}} E_r \mathcal{D} \le \rho||\mathcal{D}||^2, \forall \mathcal{D} \in R^m, \forall r$.

引理 14. 假设$(\lambda_r, \mathcal{D}^r)$是式（22）的结果. 若$\mathcal{D}^r = 0$，则$\Phi^r$为满足式（21）的 Kuhn-Tucker 条件[34]的 K-T 点；否则我们有

$$\nabla \mathcal{M}_j(\Phi^r)^{\mathbb{T}} \mathcal{D}^r \le \lambda_r \le -\frac{1}{2}(\mathcal{D}^r)^{\mathbb{T}} E_r \mathcal{D}^r < 0, j \in I(\Phi^r).$$

引理 15. 若$\mathcal{D}^r \neq 0$，则存在r使得行搜索产生步长$\delta_r = \left(\frac{1}{2}\right)^j$.

证明. 根据式（22），我们有

$$\lambda_r + (\mathcal{D}^r)^{\mathbb{T}} E_r \mathcal{D}^r \le 0 \implies \lambda_r \le -(\mathcal{D}^r)^{\mathbb{T}} E_r \mathcal{D}^r < 0$$

对于$\forall j \in I$，我们定义

$$\begin{aligned} \epsilon_r &= \mathcal{M}_j\left(\Phi^r + \delta \mathcal{D}^r + \delta^2 \widetilde{\mathcal{D}}^r\right) - \theta(\Phi^r) + \gamma \delta (\mathcal{D}^r)^{\mathbb{T}} E_r \mathcal{D}^r \\ &\le \left(\gamma - \frac{1}{2}\right) \delta (\mathcal{D}^r)^{\mathbb{T}} E_r \mathcal{D}^r + o(\delta) \end{aligned} \quad (25)$$

因此，存在一些$\bar{\delta}_j > 0$使$\epsilon_r < 0$. 设$\bar{\delta} = \min\{\bar{\delta}_j, j \in I\}$，对所有$\delta \in [0, \bar{\delta}]$均满足式（24）中的条件. 换言之行搜索总是完成的. ■

定理 16．SQP-PA 要么在有限轮数内停止在式（21）的 K-T 点Φ^r，要么产生无穷序列Φ^r，其中任何聚点都是式（21）的 K-T 点．

证明．假设 SQP-PA 算法生成一个无穷序列$\{\Phi^r\}$，并且存在子序列Ω，使得某些变量的值趋于最优，即$\Phi^r \to \Phi^*, E_r \to E_*, \mathcal{D}^r \to \mathcal{D}^*, \lambda_r \to \lambda_*, \exists r \in Q$，其中$\Phi^*, E_*, \mathcal{D}^*, \lambda_*$表示它们的最优值．接下来我们需要证明$\mathcal{D}^* = 0$，即$\mathcal{D}^r \to 0, \exists r \in Q$．根据式（24）和引理 14，可以推导出$\{\theta(\Phi^r)\}$是单调递减的．因此，考虑到$\{\Phi^r\}_{r\in\Omega} \to \Phi^*$以及$\theta(\Phi)$的连续性，我们有

$$\lim_{r\to\infty}\theta(\Phi^r) = \theta(\Phi^*) \Longrightarrow \lim_{r\to\infty}\left(\theta(\Phi^{r+1}) - \theta(\Phi^r)\right) = 0 \tag{26}$$

假设$\mathcal{D}^* \neq 0$．于是$\lambda_* < 0$．根据行搜索成功的结论，Ω上的步长$\delta_r \geq 0$．结合引理 15，我们有

$$\begin{aligned}\lim_{r\to\infty}\left(\theta(\Phi^{r+1}) - \theta(\Phi^r)\right) &\leq -\gamma\delta_r(\mathcal{D}^r)^{\mathbb{T}}E_r\mathcal{D}^r \\ &\leq -\frac{1}{2}\gamma\delta_*(\mathcal{D}^*)^{\mathbb{T}}E_*\mathcal{D}^* < 0\end{aligned} \tag{27}$$

显然式（26）和式（27）矛盾．因此$\mathcal{D}^* = 0$，即$\mathcal{D}^r \to 0, \exists r \in Q$．因此，SQP-PA 具有全局收敛性，并能获得最优结果． ■

4 性能评估

本章节首先介绍两个基准方法和几个性能比较指标（4.1 节）．然后我们描述了一些评价设置，并在 4.2 节给出了广泛的评价结果．最后，我们在小规模的边缘计算平台上实现了我们的算法，并在 4.3 节给出了测试结果．

4.1 基准方法和性能指标

我们选择了两种典型算法作为性能比较的基准（benchmark）．第一种 ADP-FL[35]属于同步联邦学习方案．在一个迭代轮次中，服务器可以通过线性搜索自适应地确定本地更新的次数，以在给定资源预算下最小化损失函数．第二种算法称为 AFO[20]，是一种可证明收敛的异步联邦优化算法，其中参数服务器将在从任意计算节点接收到一个本地更新时执行全局更新．因此，我们选择这两种算法，在同步方案（$\alpha = 1$）和异步方案（$\alpha = \frac{1}{n}$）中作为我们工作的基准方法．

为了评估训练模型的性能，我们采用了四个性能指标．（1）损失函数是模型输出与实际结果之间量化差值的概率分布．损失值反映了模型学习的质量以及是否达到了 2.2 节所述的收敛．（2）准确率是最常用的分类性能指标之一，它使用模型分类正确的数量与所有数据的

数量之比来衡量.（3）我们采用训练时间来估计学习任务的训练速度.（4）当网络中存在多个学习任务时，我们通过测量所有任务的最大损失和最小准确率来评价训练性能.

4.2 仿真评估

4.2.1 评估设定

仿真是在 AMAX 深度学习工作站[1]上进行的（CPU：Intel(R) E5-2620v4，GPU：NVIDIA GeForce RTX 2080Ti），其中我们构建了一个联邦学习仿真环境，并在 PyTorch 框架下实现了 PySyft[36]中列出的所有模型．PySyft 是 PyTorch 框架下用于隐私保护深度学习（包括联邦学习）的 Python 库.

（一）模型和数据集：我们用三个不同的模型和五个不同的数据集来评估训练过程，这些代表了各种各样的小模型、大模型和数据集. 我们采用了三种模型，分别是线性回归（Linear Regression，Linear-R）、逻辑回归（Logistic Regression，Logistic-R）和深度卷积神经网络（deep Convolutional Neural Networks，CNN）[2].

Linear-R 在能量数据集[37]上进行训练，该数据集包含来自无线传感器网络的 14803 个训练数据和 4932 个测试数据．该模型可以预测家电的能源消耗，包括一些环境参数（如温度和湿度）和一个亚计量的电能耗（如光）.

Logistic-R 在原始 MNIST 数据集[38]（称为 MNIST）上进行训练，该数据集包含 70000 个手写数字的灰度图像（60000 个用于训练，10000 个用于测试）．该模型输出一个二进制标号，该标号对应于数字是偶数还是奇数.

CNN 是在三个不同的数据集上训练的. 第一个是时尚 MNIST 数据集[39]（称为 FMNIST），其中有 70000 张时尚物品的图像（60000 张用于训练，10000 张用于测试），格式与 MNIST 相同．第二个是 CIFAR10 数据集，包括 60000 张 10 种不同类型的彩色图像（其中 50000 张用于训练，10000 张用于测试）．第三个是 CIFAR100 数据集，包含了 60000 张 100 种不同类型的彩色图像，用于训练和测试[40].

（二）网络资源：在仿真中，我们主要关注边缘计算中的带宽资源开销．具体来说，带宽消耗可以通过模型参数的大小来测量．我们在固定的资源预算（如网络带宽和训练时间）下训练了一些模型．为了实现资源高效的异步联邦学习，我们设置了参数η= 0.01，ς= 0.3，并

1 https://www.amax.com/products/gpu-platforms/.

2 实验中具体的 CNN 网络结构.（1）MNIST 有 9 层：5×5×32 卷积→局部响应归一化层→2×2 最大池化层→5×5×64 卷积→局部响应归一化层→2×2 最大池化层→1600×512 全连接→512×10 全连接→Softmax.（2）CIFAR10 和 CIFAR100 有 10 层：5×5×64 卷积→局部响应归一化层→2×2 最大池化层→5×5×128 卷积→局部响应归一化层→2×2 最大池化→3200×512 全连接→512×256 全连接→256×10 全连接→Softmax.

根据文献[35]实时估计了参数L和μ的值.

尽管 GPU 比 CPU 的训练速度更快[41]，但模型的训练仍然非常耗时. 如文献[42]提出的，为了有效地模拟我们提出的解决方案和基准方法在联邦学习中的训练过程，在仿真中共生成 100 个边缘节点，随机激活其中 10 个边缘节点参与模型训练. 该解决方案可以很容易地扩展到更多边缘节点的情况. 此外，我们以固定均值和方差均为 0.5 的高斯分布向这些边缘节点分发数据. 为便于展示和解释结果，CE-AFL 在两个全局更新之间只执行一个本地更新，即 D= 1. 我们将每个仿真实验重复 10 次，并计算平均结果.

4.2.2 仿真结果

（一）**单一学习任务**：第一组仿真评估了没有资源约束的分类模型（如 Logistic-R 和 CNN）的性能. 我们使用共 300 轮迭代的 ADP-FL 对每个模型进行训练. CE-AFL 采用三个不同的α值（0.3，0.5，0.7）进行模型训练. 为了避免混淆，我们将其表示为 CE-AFL（α）. 如图 4、图 5、图 6、图 7 所示，为了达到类似的性能（如损失函数值或分类精度），AFO 的平均迭代轮次是 ADP-FL 的 4 倍. 另外，随着α的减小，CE-AFL 的训练轮次会逐渐增加以达到与 ADP-FL 近似的性能. CE-AFL 的训练轮次明显少于 AFO. 例如，在 FMNIST 数据集上进行 CNN 训练，CE-AFL（α= 0.3）和 AFO 所需训练轮次的数量分别约为 590 和 1880.

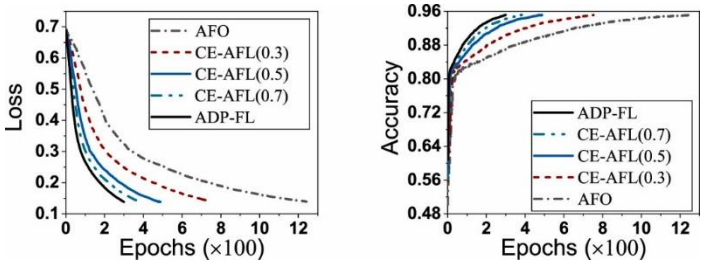

图 4 在 MNIST 上训练 Logistic-R 的损失和准确率.

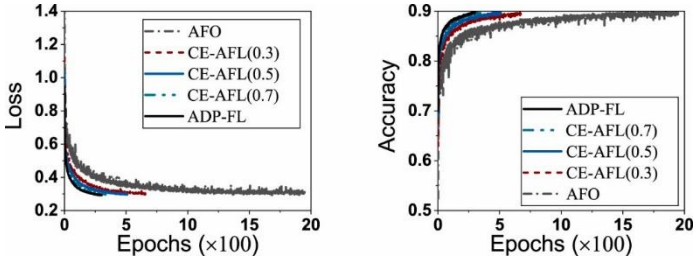

图 5 在 FMNIST 上训练 CNN 的损失和准确率.

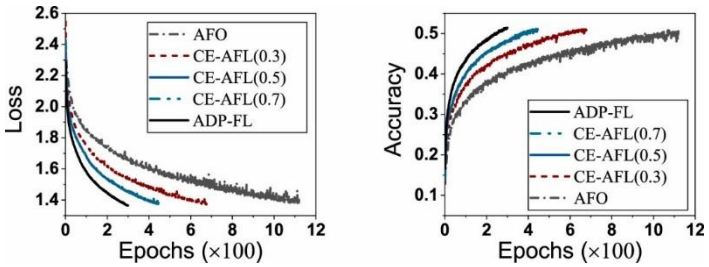

图 6 在 CIFAR10 上训练 CNN 的损失和准确率.

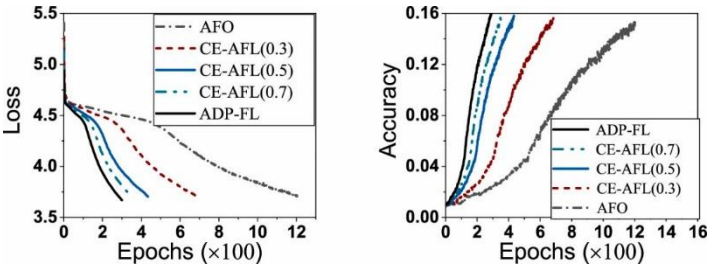

图 7 在 CIFAR100 上训练 CNN 的损失和准确率.

我们还对这些模型的训练时间进行了测试. 由于 ADP-FL 在每轮迭代都需要等待所有计算节点的本地更新，因此需要比 CE-AFL 更长的时间. 在图 8 中，CE-AFL 的训练时间随着α的变化而变化，小于 AFO（α= 0.1）和 ADP-FL（α= 1）. 例例如，当 CE-AFL 采用α的最优值时，在 FMNIST 数据集上进行 CNN 训练，CE-AFL 的最小训练时间约为 750 秒. 而 AFO 和 ADP-FL 的训练时间约为 2690 秒和 2580 秒. 因此，CE-AFL 可以比 AFO 和 ADP-FL 分别减少 72%和 70%的训练时间.

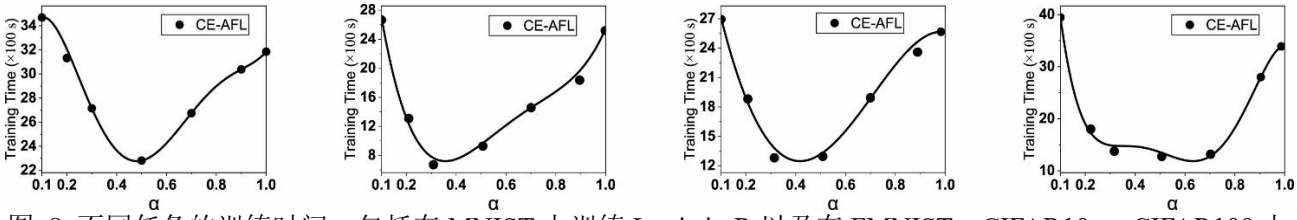

图 8 不同任务的训练时间，包括在 MNIST 上训练 Logistic-R 以及在 FMNIST、CIFAR10 、CIFAR100 上训练 CNN.

在第二组仿真中，我们在能量数据集上运行 Linear-R 模型. 在性能评估方面，我们比较了不同算法的均方误差（Mean Squared Error，MSE）性能. 如图 9 左图所示，ADP-FL 在所有方案中均方误差是最小的. 但是，在图 9 右图的结果中，我们可以看到 CE-AFL 的训练时间比其他两个基准方案要少. 具体来说，CE-AFL 比 AFO 和 ADP-FL 分别减少了 58%和 75%的训练时间.

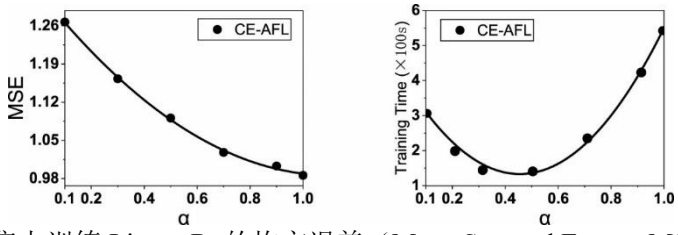

图 9 在能量数据集上训练 Linear-R 的均方误差（Mean Squared Error，MSE）和训练时间.

（二）**多个学习任务**：第三组仿真观察了系统中没有资源约束的多个学习任务的性能. 我们同时在不同的数据集上运行两个模型，包括在 MNIST 上运行的 Logistic-R，在 FMNIST 上运行的 CNN 和 CIFAR10. 每个模型训练任务执行 300 轮迭代. 图 10 展示了三组任务的损失的最大值和准确率的最小值. 可见在三种方案中，ADP-FL 的损失和准确率的性能是最好的. 当

使用 SQP-PA 为每个任务采用最优的Φ时，CE-AFL 比 ADP-FL 的性能稍差（如损失或准确性），但比 AFO 好．如给定 300 轮次的训练，CE-AFL 的损失约为 1.445，ADP-FL 的损失约为 1.338．相应地，ADP-FL 和 CE-AFL 的精度分别约为 51%和 49%．

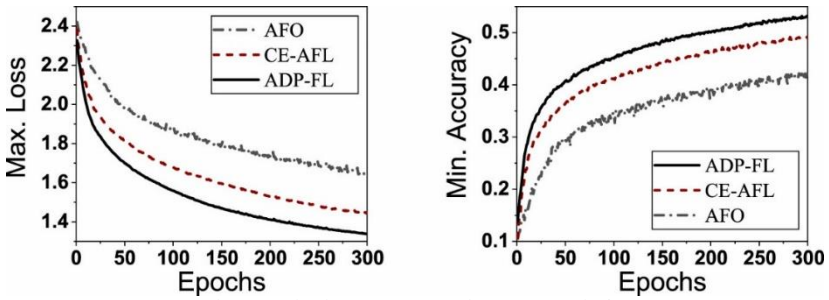

图 10 多个学习任务的损失最大值与准确率最小值．

在最后一组仿真中，我们测试了在有限的训练时间预算下多个学习任务的性能．实际情况中，一些训练任务往往需要在规定的时间内完成．我们首先考虑三个解决方案的训练时间约束．如图 11 所示，三种算法都将训练时间约束由 300 秒改变为 3000 秒后，损失的最大值变小，准确率的最小值提高．与其他两个基准方案相比，本文提出的 CE-AFL 框架具有更小的损失和更高的准确率．例如，当时间预算为 1500 秒时，CE-AFL 准确率的最小值约为 37%，而 ADP-FL 和 AFO 准确率的最小值分别仅为 29%和 19%．因此，CE-AFL 比 ADP-FL 和 AFO 的准确率的最小值分别提高了约 8%和 18%．

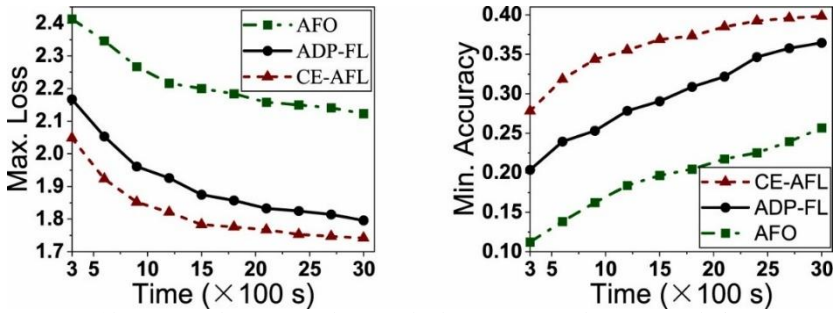

图 11 训练时间约束下，多个学习任务的损失最大值与准确率最小值．

我们进一步观察了在有限带宽预算下多个学习任务的性能．参数服务器与计算节点之间的通信将会消耗大量的网络带宽．我们通过将带宽限制从 100 Mbps 更改为 1000 Mbps 来测试这三个训练任务．如图 12 所示，CE-AFL 的准确率的最小值明显高于 AFO 和 ADP-FL．例如，当带宽约束为 800 Mbps 时，使用 CE-AFL 的三个训练任务的准确率的最小值约为 81%，而使用 AFO 和 ADP-FL 的准确率的最小值约为 72%和 76%．因此，与 AFO 和 ADP-FL 相比，我们提出的 CE-AFL 框架准确率的最小值分别提高了 9%和 5%．结果表明，在资源约束条件下，CE-AFL 能够显著提高分类精度．

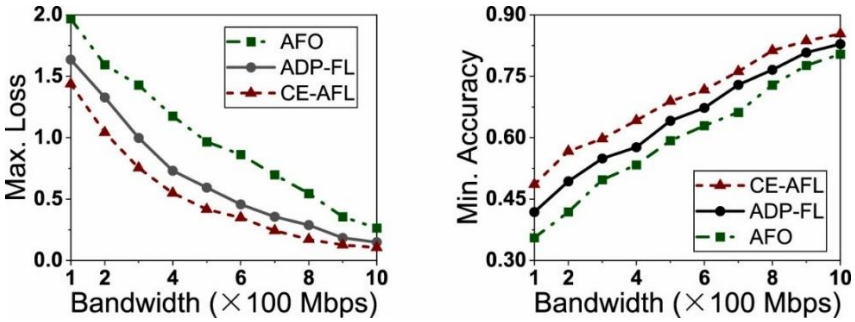

图 12 带宽约束下，多个学习任务的损失最大值与准确率最小值.

综上所述，我们提出的 CE-AFL 在单一学习任务下的训练时间比现有方案减少了约 70%．与 AFO 和 ADP-FL 相比，CE-AFL 可以降低损失的最大值，提高准确率的最小值.

4.3 实验台评估

4.3.1 实验台的实现

如图 13 所示，我们在一个实际的小规模实验台上实现了 AFO、ADP-FL 和 CE-AFL 算法，它由两个主要部分组成：一个有四个 NVIDIA GeForce RTX Titan GPU 的深度学习工作站和 10 个 Jetson TX2 开发套件[3]（CPU：ARMv8 Cortex-A57，RAM：8GB）．其中，工作站作为参数服务器，负责模型聚合和全局模型验证．我们采用 Jetson TX2 开发套件作为计算节点，在本地训练模型并将更新发送到服务器进行聚合．我们用 PyTorch 开发了一个分布式模型训练框架．计算节点和参数服务器通过无线网络物理连接在同一台路由器上．此外，它们通过 `torch.distributed` 包（gloo 后端）在逻辑上相连．具体来说，在服务器的 IP 地址和指定端口上通过 TCP 协议在服务器和工作人员之间建立连接．连接建立后，服务器将训练和测试数据集进行分段，并将分段结果发送给每个计算节点．收到结果后，计算节点生成本地数据集进行训练．我们所有的代码都可以在 GitHub 上公开访问[4].

图 13 实验台.

3 https://developer.nvidia.com/embedded/jetson-tx2-developer-kit.

4 https://github.com/lyl617/CE-AFL.

在实验台上，分别为 CIFAR10 和 FMNIST 数据集实现了两种不同类型和结构的 CNN 模型．第一个 CNN 模型用于 CIFAR10 数据集．它有两个 5×5 卷积层（64,64 通道，每层后面都有 3×3 最大池化层），具有 384 和 192 个单元的两个稠密层，一个 10 单元的 Softmax 输出层．FMNIST 数据集采用第二种 CNN 模型，该模型有两个 5×5 卷积层（32,64 通道，每层后面都有 2×2 最大池化层）、一个 1024 个单元的稠密层和一个 10 单元的 Softmax 输出层（与 FMNIST 中的 10 个类别相关）.

在实验台上，我们主要考虑不同的数据分布（包括数量和类别）对模型训练性能的影响．首先，计算节点的数据量往往是不平衡的，这些数据量随时间和空间的变化而显著变化．因此，我们采用三种不同的数据分布案例来模拟数据的不平衡．（1）案例 1：我们将相同数量的训练数据（如 6000）分配给 10 个计算节点；（2）案例 2：不同计算节点之间的数据量差异不大（如 4000~8000）；（3）案例 3：这些计算节点的数据量差异很大（如 1000~11000）．其次，不同类别的数据分布，即 I.I.D.和非 I.I.D．数据显示，在计算节点中也产生了不同的模型训练效果．例如，在 I.I.D 的情况下，每个计算节点有所有类别的数据样本（如 10 个类别）；但在非 I.I.D 的情况下，每个计算节点可能只有部分类别的（如 5 个类别）．我们采用四种不同的案例来验证数据分布对模型训练的影响：案例 I，每个数据样本随机分配给一个计算节点，因此每个计算节点都有统一（但不完全）的信息（即是 I.I.D.的数据）；案例 II，每个计算节点有 5 类数据样本；案例 III，每个计算节点有 2 类数据样本；案例 IV，每个计算节点只有 1 类数据样本．案例 II-IV 的数据样本是非 I.I.D.的，且数据分布的不均匀程度逐渐增加．

4.3.2 实验结果

在第一组实验中，我们通过分别在 FMNIST 和 CIFAR10 上训练 CNN 来测试平衡且均匀的数据．我们进行了两组实验，每组实验训练 2000 轮．从图 14、图 15 中可以看出，CE-AFL（$\alpha = 0.7$）的训练性能（即损失和准确率）与 ADP-FL 非常接近，并且远优于 AFO．例如，在 FMNIST 数据集上对 CNN 进行 2000 轮的训练时，CE-AFL 的损失值为 0.3737，而 ADP-FL 和 AFO 的损失值分别为 0.3382 和 0.6296．相应的，CE-AFL 的训练准确率约为 86.8%，ADP-FL 和 AFO 的训练准确率分别为 87.8%和 76.9%．因此，与 AFO 相比，我们提出的机制可以提高约 10%的训练准确率．

在第二组实验中，我们观察了三种不同数量的数据分布案例（案例 1-3）下的模型训练（在 FMNIST 上训练 CNN）的性能．在每一种案例下，我们都以 1000 个训练轮数为基线（baseline）方法运行 ADP-FL 算法．从图 16 可以看出，在案例 1 中，CE-AFL 需要更多的训练轮数（约 1435 轮）才能达到基线方法的损失值．这是因为在 AFO 中，服务器每次只聚合来自任意一个

计算节点更新的本地模型．此外，AFO 需要训练 9328 轮才能达到相同的训练损失．也就是说，与 ADP-FL 相比，AFO 需要 9 倍迭代轮数，而 CE-AFL 只需要 1.5 倍迭代轮数．三种不同案例下 CE-AFL 的训练损失如图 17 所示．结果表明，与案例 1 和案例 2 相比，案例 3 的实验需要更多的训练时间才能达到相同的性能（如损失）.

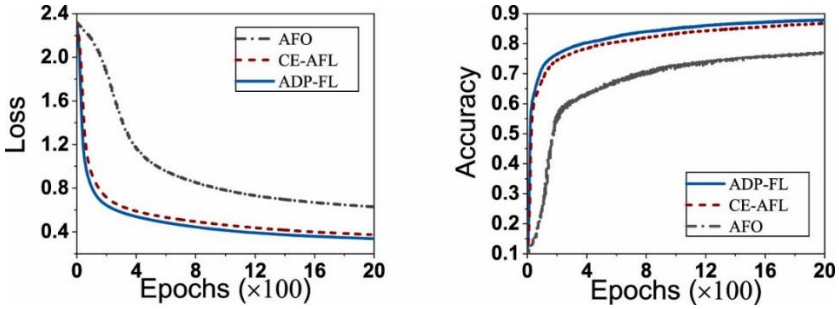

图 14 实验台中在 FMNIST 上训练 CNN 的损失和准确率.

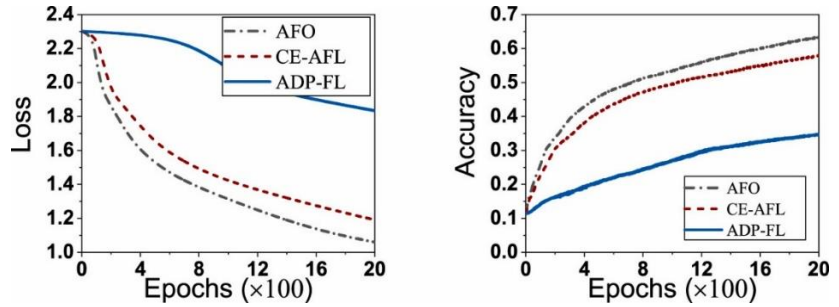

图 15 实验台中在 CIFAR10 上训练 CNN 的损失和准确率.

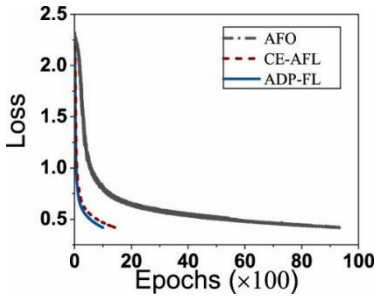

图 16 使用平衡数据的实验台上的损失与迭代轮次.

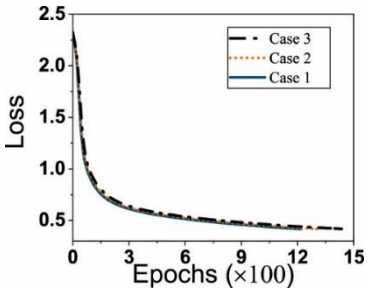

图 17 使用 I.I.D.数据的实验台上的案例 1-3 的损失与迭代轮次.

我们还观察了在案例 1-3 下，三种解决方案的训练准确性和时间．图 18 为每个案例经过训练 1000 轮后的准确率．在 CE-AFL 中，每轮迭代来自计算节点的更新模型都比 AFO 更多地参与到模型聚合中．在每个案例中，CE-AFL 都比 AFO 获得了更优的准确率性能，与 ADP-FL 的性能相近．如图 19 所示，与其他两个基线方法相比，我们提出的机制在达到与基线方法相同训练性能（损失和准确率）的情况下，使用了最少的训练时间．如在案例 1 中，CE-AFL 的训练时间约为 11835 秒，ADP-FL 和 AFO 的训练时间分别约为 22957 秒和 36587 秒．换言之，CE-AFL 比 ADP-FL 和 AFO 分别减少了 48.4%和 67.9%的训练时间.

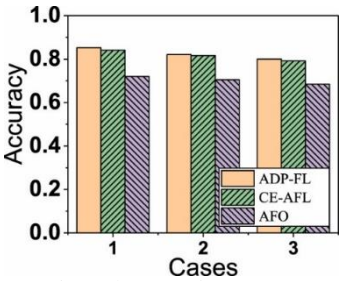

图 18 实验台上的案例 1-3 的训练准确率.

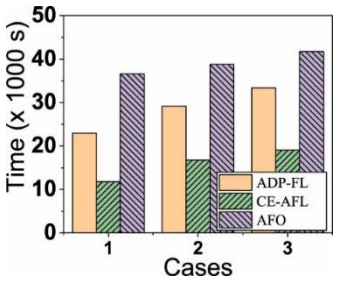

图 19 实验台上的案例 1-3 的训练时间.

最后一组实验测试了四种不同类别的数据分布案例 I-IV 下的模型训练（在 FMNIST 上训练 CNN）的性能．我们首先测试了 CE-AFL 在四种不同的数据类别分布情况下（案例 I-IV）的训练性能．从图 20 可以看出，数据类别的分布会对模型训练的速度产生影响．如在案例 IV 下，运行 5000 轮迭代的实验训练损失约为 1.3834，而在案例 II 下运行 1300 轮迭代的实验训练损失约为 0.5042．换句话说，非 I.I.D.数据的训练表现比 I.I.D.数据更糟糕．然后我们使用案例 II 测试训练性能．如图 21 所示，为达到与 ADP-FL 相同的损失（1000 轮），CE-AFL 需要的训练轮次更多（约 1310 轮）．然而在训练过程中，AFO 的损失会发生较大波动，并随训练过程逐渐增加．因此，AFO 的解决方案不能很好地处理非 I.I.D.数据，但是我们提出的 CE-AFL 可以很好地处理．

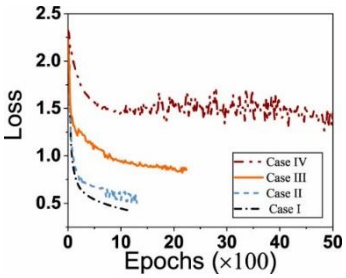

图 20 使用平衡数据的实验台上的案例 I-IV 的损失与迭代轮次.

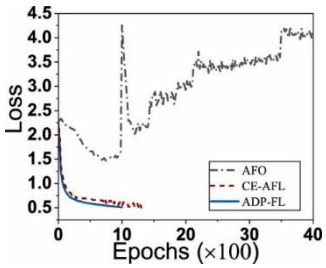

图 21 实验台上的案例 II 的损失与迭代轮次.

此外，三种方案在案例 I-IV 下的训练性能（准确率和时间）如图 22、图 23 所示．设训练共 1000 轮，我们测试了案例 I-IV 的训练准确率．如图 22 所示，在每种情况下，CE-AFL 的训练准确率始终优于 AFO．如 CE-AFL 的准确率约为 78.8%，ADP-FL 和 AFO 的准确率分别为 79.7%和 40.1%．换言之，与 AFO 相比，我们提出的机制可以提高约 39%的训练准确率．然后我们测试了 ADP-FL 和 CE-AFL 在案例 I-IV 下的训练时间．我们提出的解决方案可以有效地避免 ADP-FL 引起的掉队者问题．从图 23 可以看出，在每种情况下，CE-AFL 所需的训练时间始终小于 ADP-FL．如案例 II 中 CE-AFL 所需的训练时间约为 15562 秒，ADP-FL 所需的训练时间约为 29684 秒．因此，与 ADP-FL 相比，CE-AFL 可以减少约 47.6%的训练时间．

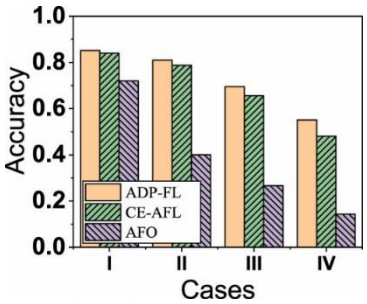

图 22 实验台上的案例 I-IV 的训练准确率.

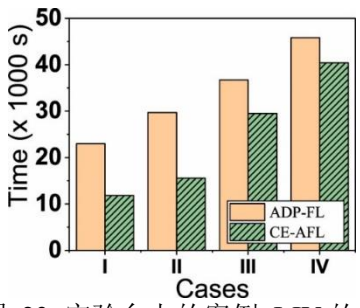

图 23 实验台上的案例 I-IV 的训练时间.

5 相关工作

近年来，联邦学习在学术界和工业界都得到了广泛的关注和研究.

与联邦学习相关的一个研究领域是通过使用工作机器和参数服务器的分布式机器学习（Distributed Machine Learning，DML）[7]. Bao 等人[43]提出了一种在线算法来调度到达的作业，并决定每个作业在其过程中的并发计算节点和参数服务器的数量，从而使所有作业的整体效用最大化. Ho 等人[44]设计了一个参数服务器系统，该系统最大限度地提高了计算计算节点在 DML 的机器学习算法上执行有意义工作的时间，该算法遵循陈旧同步并行（Stale Synchronous Parallel，SSP）计算模型. 文献[41]作者提出了一种基于参数服务器的训练大规模深度神经网络的分布式计算框架. 此外，作者还引入了一种新的学习率调制策略来对抗陈旧梯度的影响，并提出了一种新的同步协议，该协议可以有效地约束陈旧梯度，提高运行时性能，并获得良好的模型准确率.

以上工作主要是研究数据中心中 DML 的有效解决方案. 在此场景下，通常采用共享存储. 但在边缘计算中，在边缘节点之间没有共享的存储. 计算节点不会持久地存储数据，而是在学习过程开始时从共享存储中获取数据. 因此，在数据中心中不同计算节点的数据样本通常是 I.I.D.的.

在联邦学习中，数据直接在边缘侧采集，并在边缘节点上持久化存储，因此不同边缘节点上的数据分布通常是非 I.I.D.的，这一点与数据中心中的 DML 不同[24]. Smith 等人[12]表明，多任务学习很适合处理这种设置的统计挑战，并提出了一种新的系统感知优化方法，该方法对实际系统问题具有鲁棒性. 我们的方法和理论考虑了分布式多任务学习的高通信成本、掉线和容错问题. 文献[45]的作者基于新兴的无服务（serverless）架构提出了一个异步分布式机器学习框架，该框架可以在云中执行云函数（serverless function），而无需构建和维护虚拟

机基础设施．Shi 等人[46]将一些较短的通信任务合并为单个任务，以减少总体通信时间，并提出了一个优化问题，以最小化训练迭代时间．文献[47]的作者为机器学习中的分布式优化引入了一种新的且日益相关的设置，其中用于优化训练的数据分布在非常多的节点上．然而，这些解决方案大多忽略了有限的资源约束对训练性能的影响，这可能会消耗边缘计算系统的大量资源．文献[20]提出了一种新的异步联邦优化算法．对于强凸问题和非强凸问题，以及一类受限的非凸问题，我们证明了所提出的方法具有接近线性的全局最优收敛性．

最后，我们将介绍一些与我们的研究类似的工作[6][15]．文献[6]的作者在根据计算节点的资源状况积极管理计算节点的同时高效地进行联邦学习．具体来说，提出的解决方案解决了资源约束下计算节点的选择问题，这允许服务器聚合尽可能多的本地更新，并加速机器学习模型中的性能改进．Wang[15]提出了一种经验驱动的控制框架，智能选择计算节点参与每一轮联邦学习，以抵消非 I.I.D.数据引入的偏差，加速模型训练的收敛．然而，在选择计算节点的子集参加模型训练后，参数服务器仅在接收到这些计算节点的所有本地更新时才执行模型聚合．换句话说，这些计算节点在服务器上采用同步方案进行全局更新．与异步方案相比，这些研究无法解决同步障碍问题，在一定的资源预算下，同步障碍会导致训练时间变长，训练性能变差．

据我们所知，我们是第一个解决了在边缘计算系统中的联邦学习给定资源预算下确定从计算节点收到的本地更新数量的问题，以优化学习任务的训练性能．

6 总结

本文提出了一种通信高效的边缘计算异步联邦学习（CE-AFL）机制．我们分析了 CE-AFL 的收敛界．为了最小化模型训练的损失函数，我们提出了资源约束异步联邦学习（AFL-RC）问题．在此基础上，我们设计了有效的算法，分别在单个学习任务和多个学习任务中确定 CE-AFL 中α参数的最优值．仿真和实验结果表明，在资源约束条件下，CE-AFL 算法比现有算法具有更高的精度和更少的训练时间．动态场景问题是我们下一步研究的重点，通过经验驱动方法可以很容易地解决这一问题．

7 学习心得

本文基本符合科技论文的写作规范，但也存在一些差异．本文的摘要属于报道型摘要，指出了本文的研究目标是解决现有的联邦学习应用于边缘计算中存在的训练低效和资源消耗过多的问题，并提出了本文的创新点，最后阐述了与现有的方法相比本文方法有更好表现的结论．正文部分中，从联邦学习的目标出发，对其进行形式化定义，并提出了通信高效的异步联邦学习机制．在介绍本文提出的机制时，使用插图的方式对比同步方案来介绍这一机制的优势，并证明了其收敛性，使读者更容易理解．在此基础上，提出本文的问题定义: 具有资源约束的异步联邦学习问题，并将带宽这一资源约束作为研究案例进行具体分析．接下来对基于仿真的实验结果和实验台上的实验结果进行了分析，使用了多种经典的网络模型和数据集，对比现有方法，将结果以图表的方式展现，更加突出直观．文章最后讨论了现有的相关工作，并指出了本文工作的首创性，最后给出总结，并指出后续的研究内容．

在 PPT 汇报中会针对联邦学习算法的同步与异步的差异，以及这一改变为边缘计算中模型训练带来的改进和优化做具体论述．

作者贡献声明

Jianchun Liu：概念，方法，写作—初稿；Hongli Xu：方法，写作—审查和编辑；Yang Xu：形式化分析，写作—审查和编辑；Zhenguo Ma：软件，验证；Zhiyuan Wang：数据管理，验证；Chen Qian：写作—审查和编辑；He Huang：写作—审查和编辑．

竞争利益声明

作者声明，他们没有已知的可能会影响本文发表工作的相互竞争的经济利益或个人关系．

致谢

本文由国家自然科学基金 62132019、61936015、62102391 和 U1709217 资助．

参 考 文 献

[1] D. Anguita, A. Ghio, L. Oneto, X. Parra, J.L. Reyes-Ortiz, A public domain dataset for human activity recognition using smartphones, in: Esann, 2013.

[2] G. Zhu, D. Liu, Y. Du, C. You, J. Zhang, K. Huang, Towards an intelligent edge: Wireless communication meets machine learning, 2018, arXiv preprint arXiv:1809.00343.

[3] M. Satyanarayanan, The emergence of edge computing, Computer 50 (1) (2017) 30–39.

[4] H.B. McMahan, D. Ramage, 2017, http://www.googblogs.com/federated-learning-collaborative-machine-learning-without-centralized-training-data/, Google.

[5] X. Wei, Q. Li, Y. Liu, H. Yu, T. Chen, Q. Yang, Multi-agent visualization for explaining federated learning, in: Proceedings of the 28th International Joint Conference on Artificial Intelligence, AAAI Press, 2019, pp. 6572–6574.

[6] T. Nishio, R. Yonetani, Client selection for federated learning with heterogeneous resources in mobile edge, in: ICC 2019-2019 IEEE International Conference on Communications, ICC, IEEE, 2019, pp. 1–7.

[7] M. Li, D.G. Andersen, J.W. Park, A.J. Smola, A. Ahmed, V. Josifovski, J. Long, E.J. Shekita, B.-Y. Su, Scaling distributed machine learning with the parameter server, in: 11th {USENIX} Symposium on Operating Systems Design and Implementation, {OSDI} 14, 2014, pp. 583–598.

[8] M. Li, D.G. Andersen, A.J. Smola, K. Yu, Communication efficient distributed machine learning with the parameter server, in: Advances in Neural Information Processing Systems, 2014, pp. 19–27.

[9] N. Wang, B. Varghese, M. Matthaiou, D.S. Nikolopoulos, ENORM: A framework for edge node resource management, IEEE Trans. Serv. Comput. (2017).

[10] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.

[11] W. Xu, H. Zhou, N. Cheng, F. Lyu, W. Shi, J. Chen, X. Shen, Internet of vehicles in big data era, IEEE/CAA J. Autom. Sin. 5 (1) (2017) 19–35.

[12] V. Smith, C.-K. Chiang, M. Sanjabi, A.S. Talwalkar, Federated multi-task learning, in: Advances in Neural Information Processing Systems, 2017, pp. 4424–4434.

[13] B. McMahan, E. Moore, D. Ramage, S. Hampson, B.A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Artificial Intelligence and Statistics, 2017, pp. 1273–1282.

[14] S. Wang, T. Tuor, T. Salonidis, K.K. Leung, C. Makaya, T. He, K. Chan, Adaptive federated learning in resource constrained edge computing systems, IEEE J. Sel. Areas Commun. 37 (6) (2019) 1205–1221.

[15] H. Wang, Z. Kaplan, D. Niu, B. Li, Optimizing federated learning on non-IID data with reinforcement learning, in: IEEE INFOCOM 2020-IEEE Conference on Computer Communications, IEEE, 2020, pp. 1698–1707.

[16] W. Dai, A. Kumar, J. Wei, Q. Ho, G. Gibson, E.P. Xing, High-performance distributed ML at scale through parameter server consistency models, in: Twenty-Ninth AAAI Conference on Artificial Intelligence, 2015.

[17] T. Li, A.K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, V. Smith, Federated optimization in heterogeneous networks, 2018, arXiv preprint arXiv:1812.06127.

[18] Y. Chen, Y. Ning, H. Rangwala, Asynchronous online federated learning for edge devices, 2019, arXiv preprint arXiv:1911.02134.

[19] Y. Lu, X. Huang, Y. Dai, S. Maharjan, Y. Zhang, Differentially private asynchronous federated learning for mobile edge computing in urban informatics, IEEE Trans. Ind. Inf. (2019).

[20] C. Xie, S. Koyejo, I. Gupta, Asynchronous federated optimization, 2019, arXiv preprint arXiv:1903.03934.

[21] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, M. Chen, In-edge ai: Intelligentizing mobile edge computing,

caching and communication by federated learning, IEEE Netw. 33 (5) (2019) 156–165.

[22] R. Christensen, Log-Linear Models and Logistic Regression, Springer Science & Business Media, 2006.

[23] J.A. Suykens, J. Vandewalle, Least squares support vector machine classifiers, Neural Process. Lett. 9 (3) (1999) 293–300.

[24] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, V. Chandra, Federated learning with non-iid data, 2018, arXiv preprint arXiv:1806.00582.

[25] S. Ruder, An overview of gradient descent optimization algorithms, 2016, arXiv preprint arXiv:1609.04747.

[26] L. Bottou, Stochastic gradient descent tricks, in: Neural Networks: Tricks of the Trade, Springer, 2012, pp. 421–436.

[27] S. Zheng, Q. Meng, T. Wang, W. Chen, N. Yu, Z.-M. Ma, T.-Y. Liu, Asynchronous stochastic gradient descent with delay compensation, in: International Conference on Machine Learning, 2017, pp. 4120–4129.

[28] P. Raghavan, C.D. Tompson, Randomized rounding: a technique for provably good algorithms and algorithmic proofs, Combinatorica 7 (4) (1987) 365–374.

[29] R. Kanwal, K. Liu, A Taylor expansion approach for solving integral equations, Internat. J. Math. Ed. Sci. Tech. 20 (3) (1989) 411–414.

[30] L. Jacob, J.-p. Vert, F.R. Bach, Clustered multi-task learning: A convex for- mulation, in: Advances in Neural Information Processing Systems, 2009, pp. 745–752.

[31] Y. Zhang, D.-Y. Yeung, A convex formulation for learning task relationships in multi-task learning, 2012, arXiv preprint arXiv:1203.3536.

[32] A.R. Gonçalves, F.J. Von Zuben, A. Banerjee, Multi-task sparse structure learning with gaussian copula models, J. Mach. Learn. Res. 17 (1) (2016) 1205–1234.

[33] Z. Zhu, X. Cai, J. Jian, An improved SQP algorithm for solving minimax problems, Appl. Math. Lett. 22 (4) (2009) 464–469.

[34] M.A. Hanson, On sufficiency of the Kuhn–Tucker conditions, J. Math. Anal. Appl. 80 (2) (1981) 545–550.

[35] S. Wang, T. Tuor, T. Salonidis, K.K. Leung, C. Makaya, T. He, K. Chan, When edge meets learning: Adaptive control for resource-constrained distributed machine learning, in: IEEE INFOCOM 2018-IEEE Conference on Computer Communications, IEEE, 2018, pp. 63–71.

[36] T. Ryffel, A. Trask, M. Dahl, B. Wagner, J. Mancuso, D. Rueckert, J. Passerat-Palmbach, A generic framework for privacy preserving deep learning, 2018, arXiv preprint arXiv:1811.04017.

[37] L.M. Candanedo, V. Feldheim, D. Deramaix, Data driven prediction models of energy use of appliances in a low-energy house, Energy Build. 140 (2017) 81–97.

[38] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al., Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.

[39] H. Xiao, K. Rasul, R. Vollgraf, Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017, arXiv preprint arXiv:1708. 07747.

[40] Krizhevsky, G. Hinton, et al., Learning Multiple Layers of Features from Tiny Images, Tech. Rep., Citeseer, 2009.

[41] S. Gupta, W. Zhang, F. Wang, Model accuracy and runtime tradeoff in distributed deep learning: A systematic study, in: 2016 IEEE 16th International Conference on Data Mining, ICDM, IEEE, 2016, pp. 171–180.

[42] J. Konečny`, H.B. McMahan, D. Ramage, P. Richtárik, Federated optimization: Distributed machine learning for on-device intelligence, 2016, arXiv preprint arXiv:1610.02527.

[43] Y. Bao, Y. Peng, C. Wu, Z. Li, Online job scheduling in distributed machine learning clusters, in: IEEE INFOCOM 2018-IEEE Conference on Computer Communications, IEEE, 2018, pp. 495–503.

[44] Q. Ho, J. Cipar, H. Cui, S. Lee, J.K. Kim, P.B. Gibbons, G.A. Gibson, G. Ganger, E.P. Xing, More effective distributed ml via a stale synchronous parallel parameter server, in: Advances in Neural Information Processing Systems, 2013, pp. 1223–1231.

[45] H. Wang, D. Niu, B. Li, Distributed machine learning with a serverless architecture, in: IEEE INFOCOM 2019-IEEE Conference on Computer Communications, IEEE, 2019, pp. 1288–1296.

[46] S. Shi, X. Chu, B. Li, MG-WFBP: Efficient data communication for distributed synchronous SGD algorithms, in: IEEE INFOCOM 2019-IEEE Conference on Computer Communications, IEEE, 2019, pp. 172–180.

[47] J. Konečnỳ, B. McMahan, D. Ramage, Federated optimization: Distributed optimization beyond the datacenter, 2015, arXiv preprint arXiv:1511.03575.