面向联合学习的 D2D 计算任务卸载

蔡晓然，莫小鹏，许杰
（广东工业大学信息工程学院，广东 广州 510006）

摘　要：联合学习是一种分布式机器学习，边缘节点的计算和通信资源受限等因素是限制其性能优化的瓶颈。当边缘节点的计算和通信能力异构时，需要对通信和计算进行联合优化。提出了一种面向联合学习的 D2D 计算任务卸载方案，不同边缘节点通过 D2D 通信交换数据样本，平衡节点的处理能力和任务负载，以最小化联合学习模型训练过程的总时延。仿真结果表明，所提出的 D2D 计算任务卸载方案能显著提高联合学习的模型训练速度和效率。

关键词：联合学习；移动边缘计算；任务卸载；D2D 通信
中图分类号：TN929
文献标识码：A
doi: 10.11959/j.issn.2096−3750.2019.00135

D2D computation task offloading for efficient federated learning

CAI Xiaoran, MO Xiaopeng, XU Jie
School of Information Engineering, Guangdong University of Technology, Guangzhou 510006, China

Abstract: Federated learning is a kind of distributed machine learning technique. The factor of communication and computation resource constraints at the edge node is becoming the performance bottleneck. In particular, when different edge node has distinct computation and communication capabilities, the model training performance may degrade severely, thus necessitating the joint communication and computation optimization. To tackle this challenge, a computational task offloading scheme enabled by device-to-device (D2D) communications was proposed, in which different edge node exchanged data samples via D2D communication links to balance the processing capability and task load, in order to minimize the total time delay for machine learning model training. Simulation results show that compared to the benchmark scheme without such D2D task offloading the training speed and efficiency of federated learning has be improved significantly.

Key words: federated learning, mobile edge computing, task offloading, device-to-device communication

1　引言

随着物联网和人工智能的发展，自动驾驶、智慧医疗、工业自动化、虚拟现实和增强现实等新兴技术与应用如春笋般涌现。这些智能应用的有效实现依赖于传感器等不同类型的物联网节点对环境信息的快速获取、传输和汇总处理，如何支持海量无线节点的大规模数据传输和快速处理成为未来无线网络面临的巨大挑战。传统的移动云计算技术将网络边缘的数据传输到云服务器中心进行处理，随着数据量呈几何级增长，移动云计算技术面临着通信网络拥塞、端到端时延过长以及用户数据隐私保护不足等一系列问题。为了有效解决云计算技术中存在的问题，移动边缘计算（MEC，mobile edge computing）技术应运而生。该技术利用无线网络边缘的基站、接入点以及物联网节点等终端设备的通信、计算和存储能力，对数据信息和计算任务在边缘进行处理，能有效减少骨干网络的通信流量，降

收稿日期：2018−06−23；**修回日期：**2019−08−09
基金项目：国家重点研发计划资助项目（No.2018YFB1800800）；广东省重点领域研发计划资助项目（No.2018B030338001）
Foundation Items: The National Key R&D Program Subsidized Projects (No.2018YFB1800800), Research and Development Program Subsidized Projects in Key Areas of Guangdong Province (No.2018B030338001)

低端到端通信和计算时延，提高数据的隐私保护能力，进一步激发各种本地化应用创新[1-3]。

同时，未来的各种智能应用依赖于人工智能技术（如深度学习等），利用本地获取的数据样本进行人工智能模型训练和智能推演。为满足各种智能应用的低时延需求，边缘智能技术已成为一个重要的研究方向，获得了学术界和工业界的广泛关注[4-5]。该技术有效结合了 MEC 和人工智能，在网络边缘支持人工智能模型训练和智能推演。通过赋予网络边缘节点类人智能的实时响应能力，边缘智能技术可有效支持新兴的智能应用场景，提供高质量、低时延的服务体验。

在边缘智能技术中，2016 年由谷歌公司提出的联合学习[6]是一个重要的分支。与传统集中式的机器学习技术不同，联合学习旨在将机器学习模型训练过程分布在 MEC 网络中的多个边缘节点（如物联网节点等无线终端），在边缘服务器的协调下进行机器学习。参与联合学习的边缘节点可以直接利用存储在本地的数据进行模型训练，不需要分享其用户数据。具体而言，联合学习按照如下步骤迭代进行：1) 在每一次迭代中，边缘服务器将当前的全局人工智能模型参数发送给参与联合学习的所有边缘节点；2) 根据接收的模型参数，各边缘节点利用存储在本地的数据样本进行本地模型更新，如根据损失函数计算梯度并更新参数；3) 各边缘节点将更新后的模型参数上传到边缘服务器；4) 边缘服务器进行全局聚合操作，将各边缘节点发送的本地模型参数进行加权平均，得到新的全局模型参数。上述操作不断迭代进行，直至模型训练收敛。一般而言，联合学习可分为同步[7]与异步[8]两种方式，即不同边缘节点的本地模型参数更新和全局聚合需要完全同步，或者可以异步进行。文献[9]对同步联合学习和异步联合学习进行了对比。本文的研究集中于同步联合学习。

联合学习的高效实施面临一系列技术挑战：一方面，边缘节点的计算资源比较有限；另一方面，联合学习的实施依赖于边缘节点和边缘服务器之间频繁的参数更新和聚合，而随着边缘节点个数增加和人工智能模型维度变大，上述参数的更新和聚合将导致通信开销很大。因此，计算和通信资源受限是联合学习性能提升的主要瓶颈。在实际网络中，不同边缘节点在计算能力上存在异构性，并且不同边缘节点需要处理的数据样本大小也各不相同，因此，进行本地模型更新的计算执行时间会存在差异。同时，由于部署位置的差异和无线信道的衰落特性，不同边缘节点与边缘服务器之间的信道状态也不相同，使得模型参数在上传和下载过程中的性能存在差异。因此，如何优化网络的通信和计算资源分配是提高联合学习性能的重要手段。在已有工作中，文献[10]研究了在资源受限的 MEC 系统中，如何高效利用有限的资源实现自适应联合学习。然而，文献[10]只考虑了本地模型更新和全局聚合这两个过程所消耗的资源，而忽略了模型参数更新过程所消耗的通信资源。文献[11-12]研究了如何减小联合学习所需要的通信开销。如文献[11]提出了一种三元梯度法以减少联合学习的通信时间；文献[12]提出了两种降低模型参数上传链路的通信成本的方法，以提高联合学习的通信效率；文献[13]提出了一种新的多址接入方式，以实现全局模型参数的快速聚合。然而，上述已有研究都忽略了边缘节点计算和通信能力的异构性对联合学习模型训练的影响。由于这种异构性，在联合学习过程中，不同边缘节点完成本地模型更新和参数上传所消耗的时间会存在差异，而率先完成模型参数上传的节点需要等待其他节点上传模型参数，从而造成计算和通信资源的浪费，导致联合学习性能下降。因此，本文提出利用卸载技术来解决此问题。

在 MEC 系统中，卸载是一种重要的技术，可有效提高边缘节点的计算能力，缓解边缘节点计算和通信能力与任务负载不匹配的情况[14]。通常，根据卸载对象的不同，卸载技术可分为两种，分别是设备与基础设施（如基站）之间的卸载[3]以及 D2D（device-to-device）的卸载[2,15]。设备与基础设施之间的卸载是指计算能力弱或计算资源短缺的设备将部分或全部计算任务卸载到与其距离较近的基础设施（如基站）上进行处理。而 D2D 的卸载则是指计算能力较弱或计算资源短缺的设备，利用 D2D 通信等技术，将部分计算任务卸载到与其邻近的计算能力强或空闲的设备进行处理。目前，如何将卸载技术应用到联合学习中的方案还未被提出。

本文研究 MEC 网络中的联合学习。MEC 网络中面向联合学习的 D2D 计算任务卸载如图 1 所示，系统包括一个边缘服务器与多个异构的边缘节点，每个节点存储自身的用户数据。为了解决节点异构场景下的高效联合学习问题，本文提出一种面向联合学习的 D2D 计算任务卸载方案。针对边缘节点

计算和通信能力的异构性，在联合学习模型训练开始前，边缘节点利用 D2D 通信进行任务卸载，通过合理分配不同边缘节点的计算任务量，使各边缘节点的计算任务量与其计算和通信能力相匹配，从而使得 D2D 计算任务卸载与联合学习模型训练的总时间最小化，提高联合学习的效率。

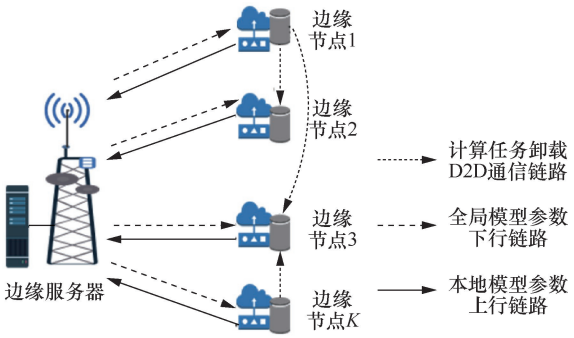

图 1 MEC 网络中面向联合学习的 D2D 计算任务卸载

本文设定不同边缘节点之间的 D2D 任务卸载和上行链路的模型参数上传均采用频分多址（FDMA，frequency division multiple access）接入协议，以避免不同边缘节点在传输过程中的相互干扰。基于此，本文的目标是优化所有边缘节点卸载的计算任务量，使得计算任务卸载过程与联合学习模型训练过程所消耗的总时间最小化。然而，由于边缘节点的计算任务卸载量是离散变量，因此，该问题是一个难以求解的非凸优化问题。为了便于求解，本文利用离散变量连续化的手段将非凸优化问题转化为凸优化问题进行求解，再对求出的连续解进行取整，以获得原问题的解。仿真结果表明，本文所提出的面向联合学习的 D2D 计算任务卸载方案能够降低边缘节点计算和通信能力的异构性对联合学习模型训练的影响，大幅度减少联合学习模型训练所消耗的时间，显著提高联合学习模型的训练效率，同时减弱数据的非独立同分布特性带来的影响，提高模型训练的准确度。

2 联合学习

本文研究基于 MEC 的联合学习系统。如图 1 所示，系统包括一个边缘服务器与 K 个边缘节点，边缘节点集合表示为 $\mathcal{K}=\{1,2,\cdots,K\}$。任意边缘节点 $i\in\mathcal{K}$ 拥有由本地存储的所有数据样本所组成的本地数据集 D_i。对于在任意一个本地数据集 D_i 中任意的一条数据样本 d，通常由数据样本的特征向量 $\boldsymbol{x}_d$ 和标签 y_d 两个部分组成。机器学习模型由模型参数 $\boldsymbol{w}$ 描述，而模型的准确度往往通过损失函数进行评估。针对模型参数 $\boldsymbol{w}$ 与数据样本 d 的特征向量 $\boldsymbol{x}_d$ 和标签 y_d，损失函数定义为 $f(\boldsymbol{w},\boldsymbol{x}_d,y_d)$，简写为 $f_d(\boldsymbol{w})$。本文主要研究平滑支持向量机（SSVM，squared support vector machine）这种机器学习模型，其损失函数如式(1)所示。

$$f_d(\boldsymbol{w})=\frac{\lambda}{2}\|\boldsymbol{w}\|+\frac{1}{2}\max(0,1-y_d\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_d)^2 \tag{1}$$

本文的研究方法也可以扩展到其他的机器学习模型中。在边缘节点 i 上，其本地损失函数定义为

$$F_i(\boldsymbol{w}_i)=\frac{\sum_{d\in D_i}f_d(\boldsymbol{w}_i)}{|D_i|} \tag{2}$$

其中，$\boldsymbol{w}_i$ 为边缘节点 i 的本地模型参数，$|D_i|$ 为数据集 D_i 的大小。

在边缘服务器上，全局损失函数定义为

$$F(\boldsymbol{w}_{\text{server}})=\frac{\sum_{i=1}^{K}|D_i|F_i(\boldsymbol{w}_i)}{\sum_{i=1}^{K}|D_i|} \tag{3}$$

其中，$\boldsymbol{w}_{\text{server}}$ 为全局模型参数，一般有

$$\boldsymbol{w}_{\text{server}}=\frac{\sum_{i=1}^{K}|D_i|\boldsymbol{w}_i}{\sum_{i=1}^{K}|D_i|} \tag{4}$$

联合学习模型的目标是找到使得全局损失函数 $F(\boldsymbol{w}_{\text{server}})$ 最小的全局模型参数 $\boldsymbol{w}^*_{\text{server}}$，即

$$\boldsymbol{w}^*_{\text{server}}=\arg\min F(\boldsymbol{w}_{\text{server}}) \tag{5}$$

为了使得全局损失函数 $F(\boldsymbol{w}_{\text{server}})$ 最小，式(5)一般使用同步的分布式梯度下降算法[9]。同步的分布式梯度下降算法共有全局模型参数下载、本地模型更新、本地模型参数上传和全局聚合 4 个步骤，联合学习模型训练过程是以上 4 个步骤的循环，直至模型训练完成。为了便于描述，本文将依次完成这 4 个步骤的一次过程称为联合学习的一帧，模型训练由若干个帧组成，联合学习的帧结构示意图如图 2 所示。

假设边缘节点与边缘服务器之间进行模型参数上传时使用 FDMA 接入协议，不同边缘节点使用不同的频段，因此，边缘节点之间不存在相互干扰。此外，模型参数传输的上/下行链路使用时分双工

（TDD，time division duplexing）技术。由于信道的互惠性，上/下行链路的信道状态信息是一致的。本文假设在联合学习模型训练过程中的无线信道是静态的，不会发生变化。

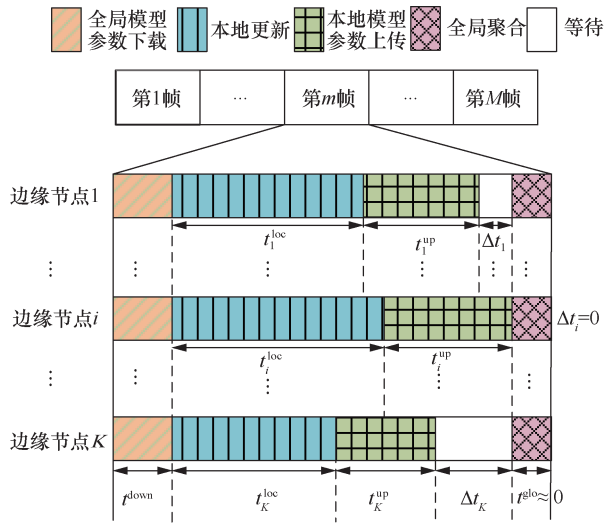

图 2　联合学习的帧结构示意图

1) 全局模型参数下载

在每一帧的开始，即联合学习模型训练开始时或边缘服务器在完成全局聚合操作后，边缘服务器需要将全局模型参数发送给各个边缘节点。全局模型参数下载的信息传输速率由信道增益最差的用户所决定[16]。令 P_{server} 为边缘服务器的发射功率，则下载全局模型参数的信息传输速率为

$$r^{\text{down}} = B\log_2\left(1+\frac{\min_{i\in\mathcal{K}}(g_i)P_{\text{server}}}{\Gamma n_0 B}\right) \tag{6}$$

其中，g_i 为边缘节点 i 与边缘服务器之间的信道功率增益，n_0 为噪声功率谱密度，B 为信道带宽，$\Gamma \geqslant 1$ 为采用实际调制编码方式下的信噪比差距，是描述实际传输速率与信道容量差距的常数，为简单起见，本文假设 $\Gamma = 1$。

因此，边缘服务器将全局模型参数发送给各边缘节点所消耗的时间为

$$t^{\text{down}} = \frac{q}{r^{\text{down}}} \tag{7}$$

其中，q 为模型参数 $\boldsymbol{w}$ 所对应的比特数。

2) 本地模型更新

所有边缘节点在接收全局模型参数后，用全局模型参数覆盖原有的本地模型参数。边缘节点利用其本地数据集对本地模型参数使用梯度下降方法进行更新的过程，称为本地模型更新操作。所有边缘节点可进行一次或多次本地模型更新操作。本文设定所有边缘节点使用批量梯度下降（BGD，batch gradient descent）法来训练模型[17]，即在一次本地模型更新操作中使用所有数据样本进行梯度更新。

对于任意的边缘节点 i，在联合学习的第 $m\in\{1,2,\cdots,M\}$ 帧中进行第 $n\in\{1,2,\cdots,N\}$ 次本地模型更新时，其本地模型参数 $\boldsymbol{w}_i^{(m,n)}$ 按以下规则进行梯度更新

$$\boldsymbol{w}_i^{(m,n)} = \begin{cases} \boldsymbol{w}_{\text{server}}^{(m)} - \eta\nabla F_i\left(\boldsymbol{w}_{\text{server}}^{(m)}\right), n=1 \\ \boldsymbol{w}_i^{(m,n-1)} - \eta\nabla F_i\left(\boldsymbol{w}_i^{(m,n-1)}\right), n>1 \end{cases} \tag{8}$$

其中，η 为学习率。在联合学习的第 m 帧中，当 $n=1$ 时，各边缘节点对边缘服务器下发的全局模型参数 $\boldsymbol{w}_{\text{server}}^{(m)}$ 进行梯度更新，具体为当 $m=1$ 时，$\boldsymbol{w}_{\text{server}}^{(m)}$ 为系统初始化的全局模型参数；当 $m>1$ 时，$\boldsymbol{w}_{\text{server}}^{(m)}$ 为对所有边缘节点的本地模型参数进行加权平均操作所得到的全局模型参数；当 $n>1$ 时，各边缘节点对第 $n-1$ 次本地模型更新所得到的本地模型参数 $\boldsymbol{w}_i^{(m,n-1)}$ 进行梯度更新。

为分析本地模型更新的时延性能，假设任意一个边缘节点使用一条数据样本进行一次本地模型更新操作需要 a 次浮点运算操作，该浮点运算操作次数由训练样本的特征数量、机器学习模型及损失函数决定。本文使用 SSVM 这一机器模型，对 SSVM 的损失函数求梯度，可得

$$\nabla f(\boldsymbol{w}) = \begin{cases} \lambda\boldsymbol{w} + \left(1-y_d\boldsymbol{w}^{\mathrm{T}}x_d\right)x_d, y_d\boldsymbol{w}x_d < 1 \\ \lambda\boldsymbol{w}, y_d\boldsymbol{w}x_d > 1 \end{cases} \tag{9}$$

联合学习模型训练过程中很难统计式(9)中的分段函数的浮点运算操作次数。因此，本文将式(9)中一条数据样本进行一次本地模型更新操作所需的浮点运算操作最大次数作为其实际执行的浮点运算操作次数。两个维度均为 u 的向量进行内积所需的浮点运算次数为 $2u-1$。设数据样本特征的维度为 u，因此，模型参数的维度也为 u。对于任意一个边缘节点的任意一条数据样本，其模型参数向量的任意一个元素求导所需最多的浮点运算操作次数为 $(2u-1)+4$，因此，对于维度为 u 的模型参数求导所需的浮点运算操作次数为 $u((2u-1)+4)=2u^2+3u$。所以，任意一个边缘节点使用任意一条数据样本进行一次本地模型更新操作所需的浮点

运算操作次数 a 为 $2u^2+3u+2u=\ \ 2u^2+5u$，当 u 足够大时，$a\approx 2u^2$。

边缘节点 i 在一个 CPU 时钟周期可进行 c_i 次浮点运算操作，其 CPU 时钟频率为 f_i。设所有边缘节点每进行 $N\geqslant 1$ 次本地模型更新操作后将本地模型参数上传到边缘服务器上，则边缘节点 i 使用批量梯度下降法进行 N 次本地模型更新操作所消耗的时间为

$$t_i^{\mathrm{loc}}\left(S_i\right)=N\frac{aS_i}{c_i}\cdot\frac{1}{f_i} \tag{10}$$

其中，S_i 为边缘节点 i 本地存储数据样本的数量。

3) 本地模型参数上传

当各边缘节点完成 N 次本地模型更新操作后，需要将其本地模型参数上传至边缘服务器。为了避免不同边缘节点之间的相互干扰，本地模型参数上传过程使用 FDMA 接入技术，将系统带宽平均分配给参与联合学习的所有边缘节点，每个边缘节点在被分配到的频带中进行本地模型参数传输。

各边缘节点依据 FDMA 接入协议进行本地模型参数上传，则所有边缘节点所分配到的传输带宽为 B/K。因此，边缘服务器接收边缘节点 i 上传的本地模型参数的信息传输速率为

$$r_i^{\mathrm{up}}=\frac{B}{K}\log_2\left(1+\frac{g_iP_i}{n_0B/K}\right) \tag{11}$$

其中，边缘节点 i 的发射功率为 P_i。

由于本地模型参数的比特数与全局模型参数的比特数相同，边缘节点 i 上传本地模型参数到边缘服务器所消耗的时间为

$$t_i^{\mathrm{up}}=\frac{q}{r_i^{\mathrm{up}}} \tag{12}$$

4) 全局聚合

边缘服务器在接收所有参与联合学习的边缘节点上传的本地模型参数后，对所有本地模型参数进行加权平均操作，得到新的全局模型参数的过程称为全局聚合。

由于边缘服务器的计算能力足够强，并且全局聚合操作只是将接收的所有本地模型参数进行加权平均操作，其计算量较少，全局聚合所消耗的时间 t^{glo} 会非常小。因此，全局聚合所消耗的时间可以忽略不计，即 $t^{\mathrm{glo}}\approx 0$。

完成 M 帧联合学习的模型训练后，系统所消耗的总时间为

$$t^{\mathrm{total}}=M\left(t^{\mathrm{down}}+\max_{i\in\mathcal{K}}\left\{t_i^{\mathrm{loc}}(S_i)+t_i^{\mathrm{up}}\right\}\right) \tag{13}$$

其中，M 为联合学习模型训练的帧数。

由于边缘节点的异构性，即不同边缘节点的 CPU 时钟频率差异、在一个 CPU 时钟周期所能执行的浮点运算操作次数差异、存储的数据样本数量差异以及发射功率差异，因此，各边缘节点在一帧中完成其本地模型更新操作过程和进行本地模型参数上传过程所消耗的时间存在差异。

由式(13)可以看出，完成一帧联合学习模型训练所消耗的时间受最后完成本地模型参数上传的边缘节点限制。由于边缘服务器需要接收所有边缘节点上传的本地模型参数才能进行全局聚合操作，而率先完成本地模型参数上传的边缘节点需要等待最后完成本地模型参数上传的边缘节点，才能得到边缘服务器全局聚合后发送新的全局模型参数，以开始下一帧。如图 2 所示，边缘节点 i 是最后完成本地模型参数上传的边缘节点，因此，边缘节点 i 的等待时间 Δt_i 为零，而此时其他先于边缘节点 i 完成本地模型参数上传的边缘节点 j 的等待时间 $\Delta t_j(j\neq i)$ 将大于或等于零。

不同边缘节点的异构性越大，则计算与通信能力越强（或数据样本较小）的边缘节点需要等待的时间越长，这将造成计算和通信能力强的边缘节点计算和通信资源的浪费，同时，拉长了联合学习模型训练过程所消耗的时间，降低了联合学习模型的训练效率。

3 问题建模

3.1 面向联合学习的 D2D 计算任务卸载

为了减少由于边缘节点计算和通信能力的异构性对联合学习模型训练效率的影响，本文提出一种面向联合学习的 D2D 计算任务卸载方案。定义执行 D2D 计算任务卸载的过程为联合学习的第 0 帧，面向联合学习的 D2D 计算任务卸载如图 3 所示。D2D 计算任务卸载利用 D2D 通信技术实现，边缘节点之间直接建立 D2D 通信链路，不需要通过基站服务进行通信。通过对参与联合学习的边缘节点的计算任务量进行分配，实现了计算任务卸载所消耗的时间与进行联合学习模型训练所消耗的时间两者的最优折中，减少了边缘节点计算和通信能力的异构性对联合学习模型训练效率的影响。

图 3　面向联合学习的 D2D 计算任务卸载

在机器学习中，数据样本数量从侧面反映了模型训练所需的计算任务量，因此，进行 D2D 的计算任务卸载实际是进行边缘节点之间的数据样本卸载。

边缘节点 i 卸载到边缘节点 j 的数据样本数量为 s_{ij}。则在完成 D2D 的计算任务卸载后，边缘节点 i 拥有的数据样本数量为

$$S_i^{'} = S_i + \sum_{j \neq i} s_{ji} - \sum_{j \neq i} s_{ij} \tag{14}$$

边缘节点之间进行计算任务卸载时使用 FDMA 接入协议，以避免不同链路之间的相互干扰。由于边缘节点间进行计算任务卸载过程中，不存在两个边缘节点间的任务互传，因此，分配给任意两个边缘节点间的通信链路的带宽为

$$\frac{B}{K(K-1)/2} \tag{15}$$

令 P_{ij} 为边缘节点 i 卸载数据样本到边缘节点 j 的发射功率，则边缘节点 i 卸载数据样本到边缘节点 j 的信息传输速率为

$$r_{ij}^{\text{offl}} = \frac{B}{K(K-1)/2} \log_2 \left(1 + \frac{h_{ij} P_{ij}}{n_0 \dfrac{B}{K(K-1)/2}} \right) \tag{16}$$

其中，h_{ij} 为边缘节点 i 与边缘节点 j 之间的信道增益。

因此，边缘节点 i 卸载数据样本到边缘节点 j 所消耗的时间为

$$t_{ij}^{\text{offl}}(s_{ij}) = \frac{b s_{ij}}{r_{ij}^{\text{offl}}} \tag{17}$$

其中，b 为一条训练数据样本的比特数。

完成 D2D 计算任务卸载后，边缘节点 i 进行本地模型更新操作所消耗的时间为

$$t_i^{\text{loc}}(S_i') = N \frac{a S_i'}{c_i} \cdot \frac{1}{f_i} \tag{18}$$

将式(14)代入式(18)中，可得

$$t_i^{\text{loc}}\left(\{s_{ij}\}\right) = N \frac{a\left(S_i + \sum_{j \neq i} s_{ji} - \sum_{j \neq i} s_{ij}\right)}{c_i} \cdot \frac{1}{f_i} \tag{19}$$

3.2　问题建模与优化

基于上述 D2D 计算任务卸载方案，本文考虑在不同边缘节点之间进行计算任务量分配的问题，最小化计算任务卸载过程与联合学习模型训练过程所消耗的总时间，实现任务卸载过程时间消耗与模型训练过程时间消耗两者的最优折中。因此，本文的目标是通过优化数据样本卸载量 $\{s_{ij}\}$，以最小化联合学习训练过程所消耗的总时延。该问题可以建模为(P1)

$$\begin{aligned} (\text{P1}): & \min_{\{s_{ij}\}} \max_{i,j \in \mathcal{K}} \left\{ t_{ij}^{\text{offl}}\left(s_{ij}\right) \right\} + \\ & M\left(t^{\text{down}} + \max_{i \in \mathcal{K}} \left\{ t_i^{\text{loc}}(\{s_{ij}\}) + t_i^{\text{up}} \right\} \right) \end{aligned} \tag{20}$$

$$\text{s.t.} \sum_{j \neq i} s_{ij} \leqslant S_i, \forall i \in \mathcal{K}$$

$$s_{ij} \geqslant 0, \forall i, j \in \mathcal{K}, j \neq i \tag{21}$$

其中，式(20)表示边缘节点 i 卸载数据样本的总量不能超过其所拥有的数据样本数量，式(21)表示数据样本卸载量为非负数。

面向联合学习的 D2D 计算任务卸载方案虽然增加了 D2D 计算任务卸载过程所消耗的时间，但能够有效降低由不同边缘节点之间计算和通信能力的异构性带来的影响，使得各边缘节点的计算任务量与其计算能力相匹配，减少联合学习模型训练过程中所消耗的时间，进而达到最小化整个系统所消耗的总时间的目标。该方案能实现计算任务卸载过程时间消耗与模型训练过程时间消耗两者的最优折中。

由于变量 $\{s_{ij}\}$ 是离散变量，问题(P1)不是凸优化问题，这使问题(P1)变得难以求解。

为了便于求解，首先把 $\{s_{ij}\}$ 放松为连续变量。同时，引入辅助变量 T^{offl} 和 $T^{\text{loc\_up}}$。于是，可将问题(P1)转化为凸优化问题，即(P1.1)。

$$(\text{P1.1}): \min_{\{s_{ij}\}, T^{\text{offl}}, T^{\text{loc\_up}}} T^{\text{offl}} + M(T^{\text{loc\_up}} + t^{\text{down}})$$

$$\text{s.t.}\ t_{ij}^{\text{offl}}\left(s_{ij}\right) \leqslant T^{\text{offl}}, \forall i, j \in \mathcal{K}, j \neq i \tag{22}$$

$$t_i^{\text{loc}}\left(\{s_{ij}\}\right) + t_i^{\text{up}} \leqslant T^{\text{loc\_up}}, \forall i, j \in \mathcal{K}, j \neq i \tag{23}$$

$$\sum_{j \neq i} s_{ij} \leqslant S_i, \forall i \in \mathcal{K} \tag{24}$$

$$s_{ij} \geqslant 0, \forall i, j \in \mathcal{K}, j \neq i \tag{25}$$

由于问题(P1.1)是一个凸优化问题，因此，可

以利用成熟的凸优化工具 CVX 对问题(P1.1)进行求解，可得连续解$\{s_{ij}\}$。在此基础上，对求解出的连续解$\{s_{ij}\}$同时进行向上取整和向下取整，通过遍历对比所有$\{s_{ij}\}$的取整组合，找出使得所求问题的值最小的解，相对应的$\{s_{ij}\}$即为求得的问题(P1)的整数解。

4　仿真实验

本节通过仿真实验来验证本文所提出的 D2D 计算任务卸载方案对联合学习过程时间消耗实现了较大的性能增益。

4.1　实验设置

本文在由一个边缘服务器和若干个边缘设备组成的蜂窝网络上模拟了一个 MEC 环境。边缘节点在 MEC 系统的分布情况如图 4 所示，其中，边缘服务器的发射功率为 50 W。系统的总带宽 B 为 100 MHz，噪声功率谱密度 n_0 为−174 dBm/Hz。

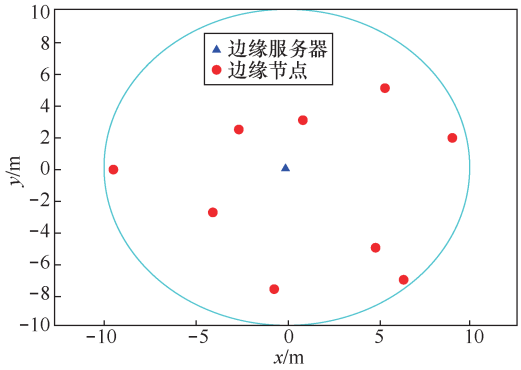

图 4　边缘节点在 MEC 系统的分布情况

设定在 MEC 系统中，有 3 种异构的边缘节点分别为 3 种不同型号的智能手机参与联合学习模型训练，分别命名为边缘节点 I、边缘节点 II 和边缘节点 III。边缘节点 I 的 CPU 时钟频率 f_{I} 为 1.5 GHz，在一个 CPU 时钟周期内所能执行的浮点运算次数 c_{I} 为 8，发射功率 P_{I} 为 2 W；边缘节点 II 的 CPU 时钟频率 f_{II} 为 1.95 GHz，在一个 CPU 时钟周期内所能执行的浮点运算次数 c_{II} 为 12，发射功率 P_{II} 为 2 W；边缘节点 III 的 CPU 时钟频率 f_{III} 为 2.6 GHz，在一个 CPU 时钟周期内所能执行的浮点运算次数 c_{III} 为 16，发射功率 P_{III} 为 2 W。本文设定边缘节点 I、边缘节点 II 和边缘节点 III 各有 3 个边缘节点参与联合学习，即共有 9 个边缘节点参与联合学习。

进行联合学习模型训练使用的公共数据集为 MNIST 数据集，共有 70 000 张黑底白字的手写数字图片，其中，60 000 张图片为训练数据样本，10 000 张图片为测试数据样本[17]。一条数据样本所拥有的比特数 b 为 6 136 bit。每条数据样本共有 784 个像素点作为数据样本的特征，因此，模型参数共有 784 个。设每个模型参数拥有 8 bit，则所有模型参数所拥有的总比特数 q 为 6 272 bit。MNIST 数据集中共有 10 个标签，分别为数字 0～9，本实验利用联合学习模型对手写数字是奇数还是偶数进行分类。

实验开始前，所有边缘节点均拥有 4 500 条数据样本，设定每个边缘节点所拥有的数据集是非独立同分布的，即每个边缘节点所拥有的数据样本的标签是只拥有 MNIST 数据集的部分标签。

本文实验使用的训练模型为 SSVM，可得一条数据样本进行一次本地模型更新操作需要进行 $2\times 784^2=1\,229\,312$ 次浮点运算操作，即 $a=1\,229\,312$。

4.2　实验结果与分析

在给定全局聚合次数 $M=40$、本地模型更新次数 $N=5$ 的条件下，对比进行 D2D 计算任务卸载前后，联合学习模型训练随时间变化的全局损失函数与模型训练准确度，D2D 计算任务卸载前后，联合学习模型的全局损失函数如图 5 所示。

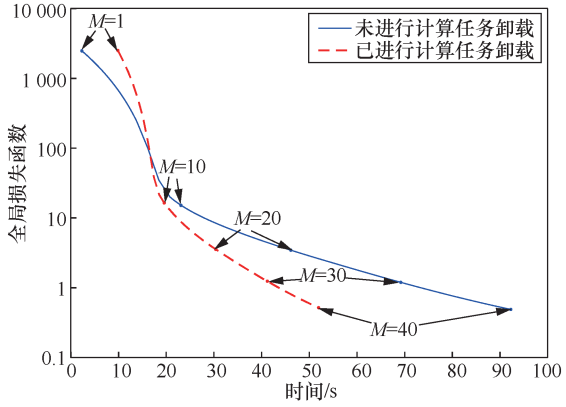

图 5　D2D 计算任务卸载前后，联合学习模型的全局损失函数

由图 5 可以看出，进行 D2D 计算任务卸载前后的联合学习模型的全局损失函数在训练过程中均呈明显下降趋势。虽然在联合学习模型训练初期，进行 D2D 计算任务卸载后的联合学习需要在计算任务卸载过程中消耗额外的时间，但在模型训练过程中，其在单位时间内全局损失函数下降率大于未进行 D2D 计算任务卸载的联合学习的单位时间内全局损失函数下降率，说明了面向联合学习的

D2D 计算任务卸载方案不仅不影响联合学习模型训练，而且能显著提高联合学习模型的训练效率。

D2D 计算任务卸载前后，联合学习模型的训练准确度如图 6 所示，同样验证了面向联合学习的 D2D 计算任务卸载方案能显著提高联合学习模型的训练效率。完成 40 次全局聚合后，进行 D2D 计算任务卸载前后的联合学习模型训练准确度分别约为 0.85 和 0.86，进行 D2D 计算任务卸载后的联合学习模型训练准确度比未进行 D2D 计算任务卸载的联合学习模型训练准确度提高了 0.01，这是因为 D2D 进行了计算任务卸载（实际为 D2D 的数据样本卸载），减弱了数据样本的非独立同分布特性，进而提高了联合学习模型训练的准确度。

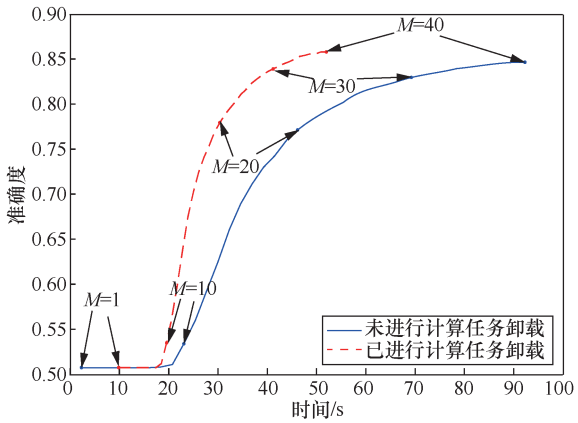

图 6　D2D 计算任务卸载前后，联合学习模型的训练准确度

图 7 为 D2D 计算任务卸载前后，不同全局聚合次数下系统的时间消耗与模型准确度的关系。每条线上共有 5 个点，分别是 M 为 10、20、30、40 和 50 时，系统的时间消耗与模型准确度对应的点。

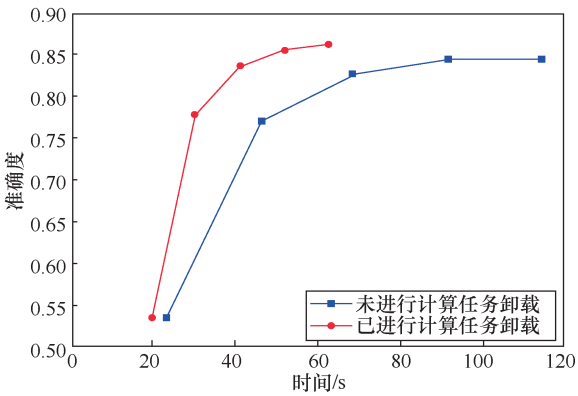

图 7　D2D 计算任务卸载前后，不同全局聚合次数下系统的时间消耗与模型准确度的关系

由图 7 可以看出，当 M 较少时，进行 D2D 计算任务卸载前后的系统所消耗的总时间差距不大，但其准确度较低。然而，随着 M 的增加，进行 D2D 计算任务卸载前后的联合学习模型训练的准确度均不断提高，但系统所消耗的总时间差距不断增大，进行 D2D 计算任务卸载的系统所消耗的总时间明显少于未进行 D2D 计算任务卸载的系统所消耗的总时间。在实际中的联合学习模型训练往往需要较多次数全局聚合才能使得联合学习模型的性能较好。因此，在实际应用中，本文提出的面向联合学习的 D2D 计算任务卸载方案能有效减少联合学习模型训练所消耗的时间，显著提高联合学习模型的训练效率。在给定相同 M 的条件下，进行 D2D 计算任务卸载后的联合学习模型训练的准确度高于未进行 D2D 计算任务卸载的联合学习模型训练的准确度，于是，本文提出的面向联合学习的 D2D 计算任务卸载方案也能减弱数据的非独立同分布特性带来的影响，提高模型训练的准确度。

5　结束语

本文考虑在 MEC 系统中联合边缘服务器与多个边缘节点的联合学习模型，研究边缘节点通信和计算的异构性对联合学习模型的训练效率的影响，并提出一种面向联合学习的 D2D 计算任务卸载方案，通过调配参与联合学习的边缘节点的数据样本数量，实现数据样本卸载所消耗的时间与进行联合学习模型训练所消耗的时间两者的最优折中，以最小化系统所消耗的时间。仿真结果验证了本文提出的面向联合学习的 D2D 计算任务卸载方案能有效降低边缘节点计算和通信能力异构性带来的影响，显著提高联合学习模型的训练效率，同时，能够减弱数据的非独立同分布特性带来的影响，提高联合学习模型训练的准确度。

参考文献：

[1] MAO Y Y, YOU C S, ZHANG J, et al. A survey on mobile edge computing: the communication perspective[J]. IEEE Communications Surveys & Tutorials, 2017, 19(4): 2322-2358.

[2] CAO X W, WANG F, XU J, et al. Joint computation and communication cooperation for energy-efficient mobile edge computing[J]. IEEE Internet of Things Journal, 2019, 6(3): 4188-4200.

[3] WANG F, XU J, WANG X, et al. Joint offloading and computing optimization in wireless powered mobile-edge computing systems[J]. IEEE Transactions on Wireless Communications, 2018, 17(3): 1784-1797.

[4] LETAIEF B K, CHEN W, SHI Y M, et al. The roadmap to 6G: AI

empowered wireless networks[J]. IEEE Communications Magazine, 2019, 57(8): 84-90.

[5] ELGAZAR A, HARRAS K, AAZAM M, et al. Towards intelligent edge storage management: determining and predicting mobile file popularity[C]//IEEE International Conference on Mobile Cloud Computing, Services, and Engineering. IEEE, 2018.

[6] KONECNY J, BRENDAN M, RAMAGE D, et al. Federated machine learning: concept and applications[J]. ACM Transactions on Intelligent Systems and Technology, 2019, 10(2): 12.

[7] ZHANG J L, TU H D, REN Y J, et al. An adaptive synchronous parallel strategy for distributed machine learning[J]. IEEE Access, 2018(6): 19222-19230.

[8] SPRAGUE M R, JALALIRAD A, SCAVUZZO M, et al. Asynchronous federated learning for geospatial applications[C]//Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, 2018.

[9] CHEN J M, MONGA R, BENGIO S, et al. Revisiting distributed synchronous SGD[C]//International Conference on Learning Representations Workshop Track. Arxiv, 2016.

[10] WANG S Q, TUOR T, SALONIDIS T, et al. Adaptive federated learning in resource constrained edge computing systems[J]. IEEE Journal on Selected Areas in Communications, 2019, 37(6): 1205-1221.

[11] WEN W, XU C, YAN F, et al. TernGrad: ternary gradients to reduce communication in distributed deep learning[C]//Advances in Neural Information Processing Systems. IEEE, 2017.

[12] LI X Y, ZHU G X, GONG Y, et al. Wirelessly powered data aggregation for IoT via over-the-air function computation: beamforming and power control[J]. IEEE Transactions on Wireless Communications, 2019, 18(7): 3437-3452.

[13] HU Y C, PATEL M, SABELLA D, et al. Mobile edge computing—a key technology towards 5G[J]. ETSI White Paper, 2015, 11(11): 1-16.

[14] XU J, CHEN L X, LIU K, et al. Designing security-aware incentives for computation offloading via device-to-device communication[J]. IEEE Transactions on Wireless Communications, 2018, 17(9): 6053-6066.

[15] LEE S J, TCHA Y, SEO S Y, et al. Efficient use of multicast and unicast channels for multicast service transmission[J]. IEEE Transactions on Communications, 2011, 59(5): 1264-1267.

[16] HUNGER R. Floating point operations in matrix-vector calculus[M]. Munich: Institute for Circuit Theory and Signal Processing, 2005.

[17] LECUN Y, BOTTOU L, BENGIO Y, et al. Gradient-based learning applied to document recognition[J]. IEEE, 1998, 86(11): 2278-2324.

[作者简介]

蔡晓然（1996– ），女，广东阳江人，广东工业大学硕士生，主要研究方向为无线通信、机器学习和移动边缘计算。

莫小鹏（1996– ），男，广东湛江人，广东工业大学硕士生，主要研究方向为无线通信、机器学习、无人机通信和移动边缘计算。

许杰（1985– ），男，四川内江人，广东工业大学教授、博士生导师，主要研究方向为无线通信、机器学习、无线能量传输、无人机通信以及移动边缘计算。