《Communication-Efficient Learning of Deep Networks from Decentralized Data》论文阅读报告

1. 摘要

现代移动设备可以访问大量适合学习模型的数据，这反过来又可以极大地改善设备上的用户体验。例如，语言模型可以改进语音识别和文本输入，图像模型可以自动选择好的照片。

然而，这些丰富的数据通常对隐私敏感，数量大，或者两者兼而有之，这可能会妨碍使用传统方法登录到数据中心并在那里进行培训。**我们提倡另一种方法，将训练数据分布在移动设备上，并通过聚合本地计算的更新来学习共享模型，我们将这种分散的方法称为联合学习。**

我们提出了一种基于迭代模型平均的深度网络联合学习的实用方法，并结合五种不同的模型结构和四个数据集进行了广泛的实证评估。这些实验表明，该方法对不平衡和非IID数据分布具有鲁棒性，这是该设置的定义特征。通信成本是主要的约束条件，我们表明，与同步随机梯度下降相比，所需的通信轮次减少了10–100倍。

2. 介绍

手机和平板电脑越来越成为许多人的主要计算设备。这些设备（包括照相机、麦克风和GPS）上的强大传感器，加上它们经常被携带，意味着它们可以访问前所未有的数据量，其中大部分数据是私人性质的。

本文提出联邦学习机制，每个客户端都有一个从未上载到服务器的本地培训数据集。相反，每个客户端计算对服务器维护的当前全局模型的更新，并且仅通过通信来进行全局模型的更新。这种方法的一个主要优点是将模型训练与直接访问原始训练数据的需要分离开来。

本文的主要贡献为：

1）将移动设备分散数据的培训问题确定为一个重要的研究方向；

2） 选择可应用于此设置的简单实用的算法；

3）对所提出的方法进行了广泛的实证评估。

联邦学习联邦学习的理想问题具有以下特性：

1）对来自移动设备的真实数据进行培训，与对数据中心中普遍可用的代理数据进行培训相比，具有明显的优势。

2） 该数据对隐私敏感或大小较大（与模型大小相比），因此最好不要将其记录到数据中心，仅用于模型培训（为重点收集原则服务）。

3） 对于受监督的任务，可以从用户交互中自然推断出数据上的标签。

联合优化，联邦优化具有与典型的分布式优化问题不同的几个关键属性：：

1）非数据独立同分布：给定客户端上的训练数据通常基于特定用户对移动设备的使用情况，因此任何特定用户的本地数据集都不能代表人口分布。

2）不平衡：类似地，一些用户会比其他用户更频繁地使用该服务或应用程序，从而导致不同数量的本地培训数据。

3）大规模分布：我们期望参与优化的客户机数量远远大于每个客户机的平均示例数量。

4）有限的沟通：移动设备经常处于脱机状态，或者连接速度慢或费用高。

在本文工作中，我们的重点是优化的非IID和不平衡特性，以及通信约束的关键性质。部署的联邦优化系统还必须解决大量的实际问题：随着数据的添加和删除，客户端数据集会发生变化；客户端可用性以复杂的方式与本地数据分发相关（例如，美式英语使用者的电话可能会在与英式英语使用者不同的时间插入）；以及从不响应或发送损坏更新的客户端。

我们假设有一个同步更新方案，它会在多轮通信中进行。有一组固定的K个客户端，每个客户端都有一个固定的本地数据集。在每轮开始时，随机选择客户端的一部分C，服务器将当前全局算法状态发送给这些客户端中的每一个（例如，当前模型参数）。为了提高效率，我们只选择了一小部分客户机，因为我们的实验表明，在某一点之外添加更多客户机的回报会递减。然后，每个选定的客户端根据全局状态及其本地数据集执行本地计算，并向服务器发送更新。然后，服务器将这些更新应用于其全局状态，并重复该过程。

当我们关注非凸神经网络目标时，我们考虑的算法适用于以下形式的任何有限和目标：

$$\min_{w\in\mathbb{R}^d} f(w) \qquad \text{where} \qquad f(w) \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} f_i(w).$$

对于机器学习问题，我们通常取$f_i(w) = (x_i, y_i; w)$，即使用模型参数w对示例（xi，yi）进行预测的损失。我们假设有K个客户机对数据进行了分区，其中P_k是客户机K上数据点的索引集，$n_k = |P_k|$。

$$f(w) = \sum_{k=1}^{K} \frac{n_k}{n} F_k(w) \quad \text{where} \quad F_k(w) = \frac{1}{n_k}\sum_{i\in\mathcal{P}_k} f_i(w).$$

如果分区P_k是通过将训练示例**均匀随机分布**在客户机上形成的，那么我们将得到$E_{Pk}[F_k(w)] = f(w)$，其中期望值是分配给固定客户机k的示例集。这是分布式优化算法通常做出的IID假设；我们将这种情况称为非IID设置（即，F_k可能是f的任意错误近似值）。

通过两种主要方式添加计算：

1）增加并行性，即在每轮通信之间使用更多独立工作的客户端；

2）增加了每个客户端上的计算，其中，每个客户端在每个通信回合之间执行更复杂的计算，而不是执行像梯度计算这样的简单计算。

我们研究了这两种方法，但我们实现的加速主要是由于在每个客户端上添加了更多的计算，一旦使用了客户端的最低并行级别。

3. FedAvg算法

许多成功的深度学习应用几乎完全依赖于随机梯度下降（SGD）的变体进行优化；许多进步可以理解为使模型的结构（以及损失函数）更易于通过简单的基于梯度的方法进行优化。

SGD可以简单地应用于联邦优化问题，在该问题中，每轮通信都要进行一次批量梯度计算（比如在随机选择的客户机上）。这种方法计算效率很高，但需要进行大量的训练才能生成好的模型。

在联邦设置中，涉及更多客户机的挂钟时间成本很低，因此对于我们的基线，我们使用大批量同步SGD；为了在联邦设置中应用这种方法，我们在每一轮中选择一组客户端，并计算这组客户端所持有的所有数据的丢失梯度。因此，C控制全局批次大小，C=1对应于完整批次（非随机）梯度下降。我们将此基线算法称为FederatedSGD（或FedSGD）。

一个典型的FedSGD实现，C=1，固定学习率η，每个客户端k计算$g_k = \bigtriangledown F_k(w_t)$，当前模型$w_t$下其本地数据的平均梯度，中央服务器聚合这些梯度并应用更新，FedSGD如下所示：

$$w_{t+1} \leftarrow w_t - \eta \sum_{k=1}^{K} \frac{n_k}{n} g_k$$

$$\sum_{k=1}^{K} \frac{n_k}{n} g_k = \nabla f(w_t)$$

FedAvg等效更新如下所示，每个客户端使用其本地数据在当前模型上进行一步梯度下降，然后服务器对得到的模型进行加权平均。我们称这种方法为FedAvg，如下所示：

$$\forall k,\ w_{t+1}^{k} \leftarrow w_t - \eta g_k$$

$$w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^{k}$$

计算量由三个关键参数控制：C在每一轮上执行计算的客户端的分数；E每个客户在每一轮中对其本地数据集进行的培训通过次数；B用于客户端更新的本地小批量大小。

我们写B=∞ 指示将完整的本地数据集视为单个小批量。因此，在该算法族的一个端点处，我们可以取B=∞ ，E=1，这正好对应于FedSGD。具体算法伪代码如下：

Algorithm 1 `FederatedAveraging`. The K clients are indexed by k; B is the local minibatch size, E is the number of local epochs, and η is the learning rate.

Server executes:

initialize w_0

for each round $t = 1, 2, \dots$ **do**

$m \leftarrow \max(C \cdot K, 1)$

$S_t \leftarrow$ (random set of m clients)

for each client $k \in S_t$ **in parallel do**

$w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$

$w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$

ClientUpdate(k, w): *// Run on client k*

$\mathcal{B} \leftarrow$ (split $\mathcal{P}_k$ into batches of size B)

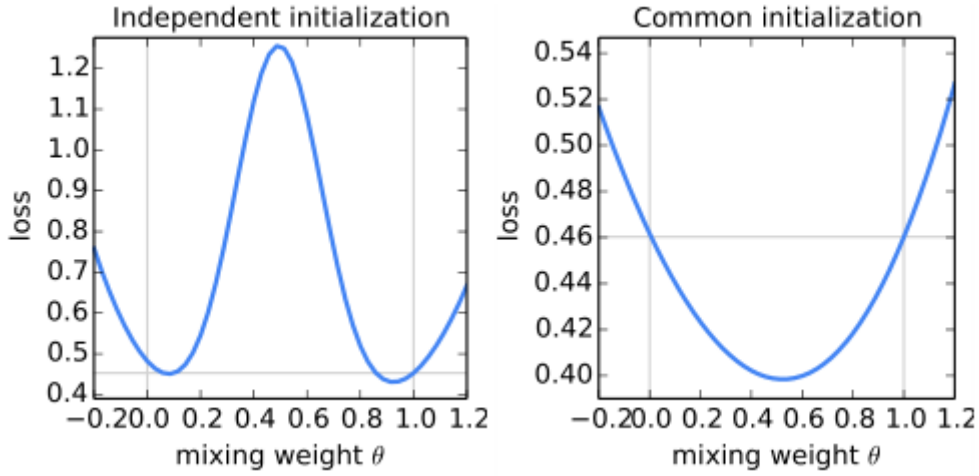
for each local epoch i from 1 to E **do**

for batch $b \in \mathcal{B}$ **do**

$w \leftarrow w - \eta \nabla \ell(w; b)$

return w to server

通过共享初始化，平均模型可以显著减少总训练集上的损失（比任何父模型的损失要好得多），实验对比结果如下：



4. 实验结果

我们研究了在客户端上划分MNIST数据的两种方法：IID，其中数据被洗牌，然后划分为100个客户端，每个客户端接收600个示例；非IID，其中我们首先按数字标签对数据进行排序，将其划分为200个大小为300的分片，并为100个客户端中的每个客户端分配2个分片。

对于语言建模，我们根据威廉·莎士比亚的全集构建了一个数据集。我们为每个剧本中的每个说话人角色构建一个客户端数据集，其中至少有两行。这产生了一个有1146个客户端的数据集。

根据这些数据，我们训练了一个堆叠的字符级LSTM语言模型，该模型在读取一行中的每个字符后，预测下一个字符。该模型将一系列字符作为输入，并将每个字符嵌入到学习的8维空间中。然后通过2个LSTM层处理嵌入的字符，每个层有256个节点。

最后，第二个LSTM层的输出被发送到softmax输出层，每个字符有一个节点。整个模型有866578个参数，我们使用80个字符的展开长度进行训练。

SHAKESPEARE LSTM, 54% ACCURACY

LSTM	E	B	u	IID	NON-IID
FEDSGD	1	∞	1.0	2488	3906
FEDAVG	1	50	1.5	1635 (1.5×)	549 (7.1×)
FEDAVG	5	∞	5.0	613 (4.1×)	597 (6.5×)
FEDAVG	1	10	7.4	460 (5.4×)	164 (23.8×)
FEDAVG	5	50	7.4	401 (6.2×)	152 (25.7×)
FEDAVG	5	10	37.1	192 (13.0×)	41 (95.3×)

提高并行性。我们首先使用客户端部分C进行实验，它控制多客户端并行性的数量。

增加每个客户端的计算量。我们将C=0.1，并在每一轮中为每个客户端添加更多计算，要么减少B，要么增加E，要么两者兼而有之。

我们可以过度优化客户端数据集吗？当前模型参数仅影响通过初始化在每个ClientUpdate中执行的优化。因此，作为E→ ∞, 至少对于凸问题，初始条件最终应该是无关的，并且无论初始化如何，都会达到全局最小值。虽然一轮平均可能会产生一个合理的模型，但额外的几轮沟通（和平均）不会产生进一步的改进。

CIFAR 实验。我们还对CIFAR-10数据集进行了实验，以进一步验证FedAvg。该数据集由10类32x32图像和三个RGB通道组成。有50000个培训示例和10000个测试示例，我们将其划分为100个客户端，每个客户端包含500个培训和100个测试示例；

SGD使用的小批量大小为100。FedSGD和FedAvg使用C=0.1，FedAvg使用E=5和B=50。

ACC.	80%	82%	85%
SGD	18000 (—)	31000 (—)	99000 (—)
FEDSGD	3750 (4.8×)	6600 (4.7×)	N/A (—)
FEDAVG	280 (64.3×)	630 (49.2×)	2000 (49.5×)

5. 结论和未来工作

我们的实验表明，联邦学习是可行的，因为FedAvg使用相对较少的几轮通信来训练高质量的模型，在各种模型体系结构上的结果表明了这一点：多层感知器、两个不同的卷积NNs、两层字符LSTM和大规模字级LSTM。虽然联邦学习提供了许多实用的隐私好处，但通过差异隐私、安全多方计算或两者的结合提供了更有力的保障，这是未来工作的一个有趣方向。请注意，这两类技术最自然地应用于FedAvg之类的同步算法。

其它学习记录

独立同分布

独立：每次抽样之间没有关系，不会相互影响。比如你在随便丢骰子，每次抛到的数字是几就是几，是独立的。 但如果我要求你要两次抛到的数字和大于等于9，第一次和第二次抛就不独立，因为他们相互关联。

同分布：你丢骰子，每次丢骰子到任何一个数字的概率都是1/6，是相等概率。或者说，在概率空间里面， 你不论进行几次抽样实验，他们都服从同样一个分布。又比如说，现在一个大小为pi的圆放在大小为4的正方形，你丢一根针进去，结果分为在圆里面和圆外面。每次丢进这个圆的概率都是pi/4，你重复+∞次会 无限接近于这个pi/4.

随机梯度下降

在每次更新时用1个样本，可以看到多了随机两个字，随机也就是说我们用样本中的一个例子来近似我所有的样本，来调整θ，因而随机梯度下降是会带来一定的问题，因为计算得到的并不是准确的一个梯度，**对于最优化问题，凸问题**，虽然**不是每次迭代**得到的损失函数都向着**全局最优**方向， 但是大的整体的方向是向全局最优解的，最终的**结果**往往是在**全局最优解附近。**但是相比于批量梯度，这样的方法更快，更快收敛，虽然不是全局最优，但很多时候是我们可以接受的，所以这个方法用的也比梯度下降方法多。

参考学习视频

https://www.youtube.com/watch?v=STxtRucv_zo&t=6s