

基于分散数据的深层网络通信有效学习



H、 布伦丹·麦克马汉·艾德·摩尔丹尼尔·拉马奇·塞思·汉普森·布莱斯·阿奎拉和阿尔卡斯——

谷歌公司，地址：651 N 34 St.，Seattle，WA 98103 USA

# 摘要

现代移动设备可以访问大量适合学习模型的数据，这反过来可以极大地改善用户在设备上的体验。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，这些丰富的数据往往对隐私敏感，数量庞大，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。我们提倡一种替代方案，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。我们称这种分散的方法为联合学习。

我们提出了一种基于迭代模型平均的深层网络联合学习方法，并对五种不同的模型结构和四种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，这是该设置的一个定义特征。通信成本是主要约束，与同步随机梯度下降相比，所需通信次数减少了10-100倍。

## 1简介

手机和平板电脑越来越成为许多人的主要计算设备[30,2]。这些设备（包括摄像头、麦克风和GPS）上强大的传感器，再加上它们经常被携带，意味着它们可以访问前所未有的大量数据，其中大部分数据本质上是私有的。从这些数据中学习到的模型

|  |
| --- |
| 2017年第20届国际人工智能与统计会议论文集，美国佛罗里达州劳德代尔堡。JMLR:W&CP卷54。作者版权所有2017。 |

它承诺通过支持更智能的应用程序来大大提高可用性，但数据的敏感特性意味着将数据存储在集中位置存在风险和责任。

我们研究了一种学习技术，它允许用户从这些丰富的数据中集体获得共享模型的好处，而无需集中存储这些数据。我们将我们的方法称为联合学习，因为学习任务是由一个由中央服务器协调的参与设备（我们称为客户端）的松散联盟来解决的。每个客户端都有一个本地训练数据集，该数据集从未上载到服务器。相反，每个客户机计算对服务器维护的当前全局模型的更新，并且只传递此更新。这是一个

直接应用2012年白宫消费者数据隐私报告提出的集中收集或数据最小化原则[39]。由于这些更新是特定于改进当前模型的，因此一旦应用了这些更新，就没有理由存储它们。

这种方法的一个主要优点是模型训练与直接访问原始训练数据的需要脱钩。显然，仍然需要对协调培训的服务器给予一些信任。但是，对于可以根据每个客户机上可用的数据指定培训目标的应用程序，联合学习可以通过将攻击面仅限于设备而不是设备和云来显著降低隐私和安全风险。

我们的主要贡献是：1）识别移动设备分散数据的训练问题作为一个重要的研究方向；2）选择了一种简单实用的算法，可以应用于这种情况；3）对所提出的方法进行了广泛的实证评估。更具体地说，我们介绍

联邦平均算法，将每个客户端上的局部随机梯度下降（SGD）与执行模型平均的服务器相结合。我们对该算法进行了大量的实验，证明了它是正确的

对不平衡和非IID数据分布具有鲁棒性，并且可以在数量级上减少训练深度网络所需的通信次数。

联合学习在移动设备上提供了一个明显的优势：1）联邦数据学习中心通常比真实数据培训具有更大的优势。2） 此数据对隐私敏感或较大（与模型的大小相比），因此最好不要将其记录到数据中心纯粹是为了模型培训（为集中收集原则服务）。3） 对于监督任务，数据上的标签可以从用户交互中自然推断出来。

许多支持移动设备智能行为的模型都符合上述标准。作为两个例子，我们考虑图像分类，例如预测哪些照片在未来最有可能被多次查看或共享；语言模型，可用于通过改进解码、下一个单词预测来改进语音识别和触摸屏键盘上的文本输入，甚至可以预测整个答案[10]。这两个任务的潜在训练数据（用户拍摄的所有照片以及他们在移动键盘上键入的所有内容，包括密码、url、消息等）可能对隐私敏感。从中提取这些示例的分布也可能与容易获得的代理数据集有很大的不同：聊天和文本消息中使用的语言通常与标准语言库（如Wikipedia和其他web文档）有很大不同；人们在手机上拍摄的照片可能与典型的Flickr照片。最后，这些问题的标签是直接可用的：输入的文本是为了学习语言模型而自我标记的，照片标签可以通过用户与照片应用程序（照片被删除、共享或查看）的自然交互来定义。

这两个任务都非常适合学习神经网络。对于图像分类，前馈深网络，特别是卷积网络，众所周知，它提供了最先进的结果[26，25]。对于语言建模任务，递归神经网络，尤其是LSTMs，已经取得了最先进的结果[20,5,22]。

与数据中心持久化数据培训相比，隐私联合学习具有明显的隐私优势。即使持有一个“匿名化”的数据集，通过与其他数据的连接，仍然会使用户隐私面临风险[37]。相比之下，为联合学习传输的信息是改进特定模型所需的最小更新（自然，隐私权益的强度取决于更新的内容）。[1]更新本身可以（而且应该）是短暂的。它们永远不会包含比原始训练数据更多的信息（通过数据处理不平等），并且通常包含的信息要少得多。此外，聚合算法不需要更新的源，因此可以在不识别元数据的情况下通过诸如Tor[7]的混合网络或通过可信的第三方来传输更新。最后，我们简要讨论了联合学习与安全多方计算和差分隐私相结合的可能性。

联邦优化我们将联邦学习中隐含的优化问题称为联邦优化，它与分布式优化建立了联系（和对比）。联邦优化有几个关键属性，使其与典型的分布式优化问题不同：

•非IID给定客户机上的培训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总体分布。

•不平衡类似地，一些用户会比其他用户更多地使用服务或应用程序，从而导致不同数量的本地培训数据。

•大规模分布我们期望参与优化的客户机数量远大于每个客户机的平均示例数。•有限的通信移动设备经常离线或连接速度慢或昂贵。

在这项工作中，我们的重点是优化的非IID和非平衡特性，以及通信约束的临界性质。部署的联邦优化系统还必须解决大量实际问题：随着数据的添加和删除而变化的客户端数据集；以复杂方式与本地数据分布相关的客户端可用性（例如，来自美语英语使用者的电话可能在不同的时间插入，而不是以英语为母语的人）以及从不响应或发送损坏更新的客户端。

这些问题已经超出了当前工作的范围；相反，我们使用了一个适合于实验的受控环境，但仍然解决了客户机可用性以及不平衡和非IID数据等关键问题。我们假设一个同步更新方案，在多轮通信中进行。有一组固定的客户机，每个客户机都有一个固定的本地数据集。在每轮开始时，随机选择一部分客户机，服务器将当前全局算法状态发送给每个客户机（例如，当前模型参数）。我们只选择了一小部分客户机以提高效率，因为我们的实验表明，在超过某一点的情况下，增加更多客户机的回报会递减。然后，每个选定的客户机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，然后重复该过程。*KC*

当我们关注非凸神经网络目标时，我们考虑的算法适用于任何形式的有限和目标，其中

对于机器学习问题，我们通常取（w）=*金融机构*

*`*（席席，易；W），即模型参数（Xi，Yi）的预测损失。我们假设有客户机对数据进行了分区，其中Pk是客户机上数据点的索引集，with=| Pk |。因此，我们可以将目标（1）改写为*wKknk公司*

哪里*.*

如果分区Pk是通过随机地将训练样本均匀地分布在客户机上形成的，那么我们将得到EPk[Fk（w）]=f（w），其中期望值超过分配给固定客户机的一组示例。这是典型的分布式优化算法所做的IID假设；我们将这种情况称为非IID设置（即，可能是任意错误的近似值）。*kFk公司f*

在数据中心优化中，通信成本相对较小，计算成本占主导地位，最近的重点是使用gpu来降低这些成本。相比之下，在联邦优化中，通信成本占主导地位——我们通常会受到1MB/s或更低的上传带宽的限制。此外，客户通常只会自愿参与优化，当他们充电，插上电源，并使用未测量的wi-fi连接。此外，我们预计每个客户每天只参与少量更新轮。另一方面，由于任何单个设备上的数据集相对于数据集的总大小来说都很小，而且现代智能手机的处理器（包括GPU）相对较快，因此与许多型号的通信成本相比，计算基本上是免费的。因此，我们的目标是使用额外的计算来减少训练模型所需的通信轮数。我们可以通过两种主要方式增加计算量：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机；2）在每个客户机上增加计算量，而不是执行像梯度计算这样的简单计算，每个客户端在每轮通信之间执行更复杂的计算。我们对这两种方法进行了研究，但是我们实现的加速主要是由于在每个客户机上增加了更多的计算，一旦在客户机上使用了最低级别的并行度。

McDonald等人研究了通过迭代平均局部训练模型进行的相关工作分布式训练。[28]对于感知器和Povey等人。[31]用于语音识别DNNs。Zhang等人。[42]研究了一种“软”平均的异步方法。这些工作只考虑集群/数据中心设置（最多16个工作人员，基于快速网络的挂钟时间），而不考虑非平衡和非IID的数据集，这些属性对于联邦学习设置至关重要。我们将这种算法风格应用于联邦设置，并执行适当的经验评估，这会提出与数据中心设置中相关的问题不同的问题，并且需要不同的方法。

使用与我们相似的动机，Neverova等人。[29]还讨论了将敏感用户数据保存在设备上的优点。Shokri和Shmatikov[35]的工作有几个方面的联系：他们专注于训练深层网络，强调隐私的重要性，通过在每一轮通信过程中只共享一个子集的参数来解决通信成本问题；但是，他们也不考虑非平衡和非IID数据，而实证评价是有限的。

在凸环境下，分布式优化和估计问题受到了极大的关注[4,15,33]，有些算法确实特别关注通信效率[45,34,40,27,43]。除了假设凸性外，这项现有的工作通常要求客户机的数量远远小于每个客户机的示例数，数据以IID的方式分布在客户机之间，并且每个节点都有相同数量的数据点-所有这些假设在联邦优化中都是违反的设置。异步分布形式的SGD也被应用于训练神经网络，如Dean等人。[12] ，但这些方法需要在联邦设置中进行大量更新。分布式一致性算法（例如[41]）放宽了IID假设，但仍然不适合在很多客户机上进行通信约束优化。

我们考虑的（参数化）算法族的一个端点是简单的一次平均法，其中每个客户机求解的模型将其局部数据的损失最小化（可能是正则化的），然后对这些模型进行平均以生成最终的全局模型。这种方法在IID数据的凸情况下得到了广泛的研究，众所周知，在最坏的情况下，生成的全局模型并不比在单个客户机上训练一个模型好[44，3，46]。

## 2联邦平均算法

最近，深度学习的大量成功应用几乎完全依赖于随机梯度下降（SGD）的变体进行优化；事实上，许多进展可以理解为通过简单的基于梯度的方法调整模型的结构（以及由此产生的损失函数），使其更易于优化[16]。因此，我们很自然地从SGD开始构建联邦优化算法。

SGD可以天真地应用于联邦优化问题，在这个问题中，每一轮通信只需进行一次批量的梯度计算（比如在随机选择的客户机上）。这种方法计算效率很高，但需要大量的训练才能产生好的模型（例如，即使使用诸如批量标准化、Ioffe和Szegedy[21]训练的MNIST等先进方法，在尺寸为60的小批量上进行50000步的训练）。我们在CIFAR-10实验中考虑了这个基线。

在联邦环境下，涉及更多客户机的挂钟时间成本很小，因此对于基线，我们使用了大批量同步SGD；Chen等人的实验。[8] 说明这种方法在数据中心环境中是最先进的，它比异步方法的性能要好。为了在联邦环境中应用这种方法，我们在每一轮中选择一小部分客户机，并计算这些客户机持有的所有数据的损失梯度。因此，控制全局批量大小，与=1对应的全批次（非随机）梯度下降。[2]我们将此基线算法称为联邦SGD（或FedSGD）。*CCC*

FedSGD的一个典型实现是=1和固定的学习率，每个客户端计算=O（wt），即当前模型下其本地数据的平均梯度*Cηk吉克Fk公司重量*，中央服务器聚合这些渐变并应用更新*,* 自从

. 由和给出等效更新。

也就是说，每个客户端在本地使用其本地数据对当前模型进行一步梯度下降，然后服务器对得到的模型进行加权平均。一旦以这种方式编写算法，我们可以通过在平均步骤之前多次迭代局部更新←wk−ηO（wk），为每个客户机增加更多的计算量。我们称这种方法为联邦平均（FedAvg）。计算量由三个关键参数控制：每轮执行计算的客户端的分数；然后每个客户端在每轮上对其本地数据集进行的训练通过数；以及用于客户端更新的本地小批量大小。我们写下=∞来表示整个局部数据集被视为一个小批量。因此，在该算法族的一个端点处，我们可以取与FedSGD完全对应的∞和=1。对于一个客户*周Fk公司CEBBBEnk公司*局部示例，每轮的本地更新次数由给出；算法1中给出了完整的伪代码。

对于一般的非凸目标，参数空间中的平均模型会产生任意坏的模型。

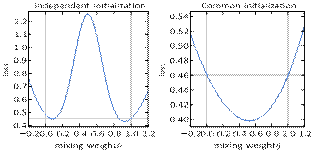


图1：对两个模型的参数取平均值并使用+（1−θ）w0对50个均匀间隔值∈[-0.2,1.2]生成的模型的完整MNIST训练集的损失。模型和在不同的小数据集上使用SGD进行训练。对于左绘图，和使用不同的随机种子初始化；对于右绘图，使用共享种子。注意不同的轴刻度。水平线给出了或实现的最佳损失*ww*0 *θwθww*0 *ww*0 *是的ww*0

（相当接近，对应于=0和=1处的垂直线）。在共享初始化的情况下，平均模型可以显著减少总训练集上的损失（比任何一个父模型的损失要好得多）。*θθ*

遵循古德费罗等人的方法。[17] ，当我们平均两个从不同初始条件训练出来的MNIST数字识别模型[3]时，我们正好看到了这种不良行为（图1，左图）。对于这个图，父模型和每个都接受了来自MNIST训练集的600个样本的非重叠IID样本的训练。培训是通过SGD进行的，固定学习率为0.1，在大小为50的小批量中进行240次更新（或者在600大小的小数据集上通过20次）。这大约是模型开始过度拟合其本地数据集的训练量。*ww*0 *E*

最近的工作表明，在实践中，充分过参数化NNs的损失面令人惊讶地良好地保存，特别是不容易出现坏的局部极小值，这比以前认为的要小[11,17,9]。事实上，当我们从相同的随机初始化开始两个模型，然后再次在不同的数据子集上独立地训练每个模型时（如上所述），我们发现朴素的参数平均效果出奇地好（图1，右图）：这两个模型的平均值在整个MNIST训练集上的损失显著降低比单独在两个小数据集上训练得到的最佳模型要好。图1从随机初始化开始，注意一个共享的启动模型*重量*用于每一轮的FedAvg，所以同样的直觉也适用。

算法1联邦平均。客户机按索引；是本地小批量大小，是本地时间段的数量，是学习率。*KkBEη*



服务器执行：初始化*w*0

每轮=1,2，。。。do←最大值（C·K，1）*t米*

*St*←（随机客户集）每个客户∈St*米k*并行执行ClientUpdate（克，重量）



ClientUpdate（）：//在客户端B上运行←（将Pk拆分为批量大小），从1到do的每个本地epoch*k、 wkB我E*

对于批处理∈B do←w−ηO（w；B）返回服务器*bw`w*



## 3实验结果

我们受到图像分类和语言建模任务的激励，在这些任务中，好的模型可以极大地提高移动设备的可用性。对于每一个任务，我们首先选择一个足够小的代理数据集，这样我们就可以彻底研究FedAvg算法的超参数。虽然每个单独的训练运行相对较小，但我们为这些实验训练了超过2000个单独的模型。然后，我们给出了基准CIFAR-10图像分类任务的结果。最后，为了证明FedAvg在实际问题上的有效性，我们对一个大型语言建模任务进行了评估。

|  |
| --- |
|  |
|  |  |

我们的初步研究包括两个数据集上的三个模型族。前两个用于MNIST数字识别任务[26]：1）一个简单的多层感知器，具有两个隐藏层，每个层使用ReLu激活（199210个总参数），我们称之为mnist2nn。2） CNN有两个5x5卷积层（第一个有32个通道，第二个有64个通道，每个都有2x2个最大池），一个有512个单元和ReLu激活的完全连接层，以及一个最终的softmax输出层（总参数1663370）。为了研究联邦优化，我们还需要指定数据在客户机上的分布方式。我们研究了两种在客户端上划分MNIST数据的方法：IID，其中的数据被洗牌，然后划分为100个客户端，每个客户端接收600个示例；而非IID，我们首先根据数字标签对数据进行排序，将其分为200个300大小的碎片，并为每个100个客户端分配2个碎片。这是一个病态的非IID数据分区，因为大多数客户端只有两位数的示例。因此，这让我们探索我们的算法在高度非IID数据上的破坏程度。这两个分区是平衡的，表1：客户端分数对mnist2nn=1和CNN=5的影响。注意=0.0对应于每轮一个客户机；因为我们使用100个客户机作为MNIST数据，所以行对应于1、10、20、50和100个客户机。每个表条目给出了第2NN和CNN分别达到97%和99%的测试集精度所需的通信轮数，以及相对于=0基线的加速。在允许的时间内，五次大批量运行未达到目标精度。*CEECC*

      2NN IID非IID

      0.5-（-）75（4.2×）—（-）443（7.4×）

|  |  |  |  |
| --- | --- | --- | --- |
| 1.0-（-） | 70 （4.5倍） | — (—) | 380 （8.6×） |
| CNN，=5*E* |  |  |  |
| 0.0            387 | 50 | 1181 | 956 |
| 0.1             339 （1.1×） | 18 （2.8×） | 1100 （1.1×） | 206 （4.6×） |
| 0.2             337 （1.1×） | 18 （2.8×） | 978 （1.2×） | 200 （4.8×） |
| 0.5             164 （2.4×） | 18 （2.8×） | 1067 （1.1×） | 261 （3.7×） |
| 1.0             246 （1.6×） | 16 （3.1×） | — (—) | 97 （9.9×） |

然而[4]

对于语言建模，我们从莎士比亚的全集中建立了一个数据集[32]。我们为每个角色构建一个客户端数据集，每个角色至少有两行。这产生了一个包含1146个客户机的数据集。对于每个客户机，我们将数据分成一组训练行

（角色的前80%行）和测试行（最后20%，四舍五入到至少一行）。得到的数据集在训练集中有3564579个字符，在测试集中有870014个字符[5]。这些数据基本上是不平衡的，许多角色只有几行，少数角色有大量的行。此外，观察测试集不是一个随机的线路样本，而是由每个剧本的时间顺序进行时间间隔。使用相同的训练/测试分割，我们还形成了一个平衡的IID版本的数据集，同样有1146个客户机。

在这些数据上，我们训练了一个堆叠的字符级LSTM语言模型，它在读取一行中的每个字符后，预测下一个字符[22]。该模型以一系列字符作为输入，并将这些字符嵌入到一个学习的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LSTM层的输出被发送到每个字符有一个节点的softmax输出层。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。

表2：达到目标的通信轮数

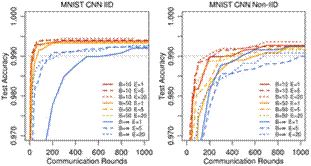
FedAvg与FedSGD（第一行，=1且=∞）的精确度。该列给出=En/（KB），即每轮更新的预期数量。*EB美国美国*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 美国有线电视新闻网 | 美国有线电视新闻网  *E* | *B* | ，99%*美国* | | 精度IID | |  | 非IID | |
| 联邦政府 | 1 | ∞ | 1 | | 626 | | 483 | |  |
| 费达夫 | 5 | ∞ | 5 | | 179 （3.5倍） | | 1000 | | （0.5×） |
| 费达夫 | 1 | 50 | 12 | | 65 （9.6倍） | | 600 | | （0.8×） |
| 费达夫 | 20 | ∞ | 20 | | 234 （2.7×） | | 672 | | （0.7×） |
| 费达夫 | 1 | 10 | 60 | | 34 （18.4×） | | 350 | | （1.4×） |
| 费达夫 | 5 | 50 | 60 | | 29 （21.6×） | | 334 | | （1.4×） |
| 费达夫 | 20 | 50 | 240 | | 32 （19.6倍） | | 426 | | （1.1×） |
| 费达夫 | 5 | 10 | 300 | | 20 （31.3×） | | 229 | | （2.1×） |
| 费达夫 | 20 | 10 | 1200 | | 18 （34.8×） | | 173 | | （2.8×） |
| S  LSTM | 哈克斯  *E* | 皮尔  *B* | 第一季度，*美国* | 54%准确度  IID公司 | | | 不 | | -IID公司 |
| 联邦政府 | 1 | ∞ | 1.0 | 2488 | |  | 3906 | | |
| 费达夫 | 1 | 50 | 1.5 | 1635 | | （1.5倍） | 549 （7.1×） | | |
| 费达夫 | 5 | ∞ | 5.0 | 613 | | （4.1×） | 597 （6.5倍） | | |
| 费达夫 | 1 | 10 | 7.4 | 460 | | （5.4×） | 164 （23.8×） | | |
| 达夫格 | 5 | 50 | 7.4 | 401 | | （6.2×） | 152 （25.7×） | | |
| 费达夫 | 5 | 10 | 37.1 | 192 （13.0×） | | | 41 （95.3倍） | | |
|  |  |  |  |  |  |  |  |  |  |

SGD对学习速率参数的调整非常敏感。这里报告的结果是基于在足够宽的学习率网格上的训练（在分辨率为10的乘法网格上通常为11-13个值）*ηη*或者10个). 我们检查以确保最佳学习率在我们的表格中间，并且最佳学习率之间没有显著差异。除非另有说明，否则我们将为每个轴值单独选择的最佳执行率绘制度量。我们发现，最佳学习速率与其他参数的函数变化不大。*十*

增加并行性我们首先尝试使用客户机部分，它控制多客户机并行性的数量。表1显示了变化对两个MNIST模型的影响。我们报告实现目标测试集精度所需的通信轮数。为了计算这一点，我们为每个参数设置组合构建一条学习曲线，按照上述方法进行优化，然后通过取所有先前轮次中获得的测试集精度的最佳值，使每个曲线单调地改进。然后，我们使用形成曲线的离散点之间的线性插值计算曲线与目标精度相交的圆数。通过参考图2可以更好地理解这一点，其中灰色线显示目标。*CCη*

使用=∞（对于MNIST将所有600个客户示例作为每轮的单个批处理），在增加客户机分数方面只有一个小小的优势。使用较小的批大小=10表明使用≥0.1的显著改进，尤其是在非IID情况下。基于这些结果，在我们剩下的大部分实验中，我们确定=0.1，这在计算效率和收敛速度之间取得了很好的平衡。比较中=∞和=10列的轮数*BBCCBB*



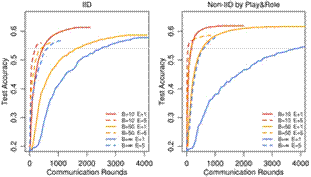


图2:MNIST CNN（IID，然后是病理非IID）和Shakespeare LSTM（IID，然后按游戏和角色）的测试集精确度与沟通轮数的比较，结果为=0.1并经过优化。灰色线表示表2中使用的目标精度。附录A中的图7给出了2NN的曲线图。*Cη*

表1显示了一个戏剧性的加速，我们接下来将对此进行研究。

在本节中，增加每个客户机的计算量，我们固定为0.1，并在每轮计算中为每个客户机增加更多的计算量，可以减少、增加，也可以两者兼而有之。图2表明，每轮增加更多的本地SGD更新可以显著降低通信成本，表2对这些加速进行了量化。每轮每个客户端的预期更新次数为=（E[nk]/B）E=nE/（KB），其中期望值超过随机客户端的绘制量。我们根据这个统计数据对表2中每个部分的行进行排序。我们看到通过改变两者来增加和是有效的。只要足够大，可以充分利用客户端硬件上可用的并行性，就基本上不需要为降低并行度而在计算时间上付出任何代价，因此在实践中，这应该是首先调整的参数。*CBE美国k美国EBB*

对于MNIST数据的IID划分，使用每个客户端更多的计算量，CNN和2NN分别减少35倍和46倍（2NN详见附录A表4）。病理分区的非IID数据的加速比较小，但仍然相当可观（2.8–3.7×）。令人印象深刻

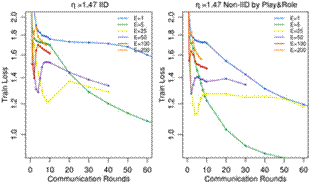


图3：莎士比亚LSTM在固定学习率为1.47的情况下，平均步数、固定值=10和=0.1之间的许多局部时期的训练效果。*EBCη*

当我们天真地对训练在完全不同的数字对上的模型参数进行平均时，这种平均方法提供了任何优势（与实际的发散相比）。因此，我们认为这是这种方法稳健性的有力证据。

莎士比亚作品的非均衡性和非个人化分布

（按角色划分）更能代表我们期望在现实世界中应用的那种数据分布。令人鼓舞的是，对于这个问题，在非IID和非平衡数据上学习实际上要容易得多（95倍的加速比13倍的平衡IID数据）；我们推测这主要是由于某些角色具有相对较大的本地数据集，这使得增加局部训练特别有价值。

对于所有三个模型类，FedAvg收敛到比基线FedSGD模型更高的测试集精度。即使这些线超出了绘制的范围，这种趋势也会继续。例如，对于CNN，=∞，E=1的FedSGD模型在1200轮后最终达到99.22%的准确率（并且在6000轮后没有进一步提高），而在300轮后，=10，E=20 FedAvg模型的准确率达到99.44%。我们推测，除了降低通信成本外，模型平均还会产生与辍学相似的正则化好处[36]。*BB*

我们主要关注的是在模拟测试的准确性方面，我们主要关注的是测试的准确性。我们观察到所有三个模型类的相似行为，并在附录A的图6中展示了MNISTNN的曲线图。

我们可以对客户端数据集进行过度优化吗？当前模型参数只影响通过初始化在每个ClientUpdate中执行的优化。因此，作为→∞，至少对于一个凸问题，初始条件应该是无关的，并且不管初始化如何，都会达到全局最小值。即使对于一个非凸问题，人们也可以猜测算法-*E*

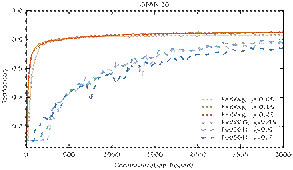


图4:CIFAR10实验的测试精度与通信。FedSGD使用每轮0.9934的学习率衰减；FedAvg使用=50，每轮学习率衰减0.99，并且=5。*BE*

只要初始化在同一盆地内，rithm收敛到相同的局部极小值。也就是说，我们预计，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。

图3显示了初始训练期间大容量对莎士比亚LSTM问题的影响。事实上，对于大量的局部时代，FedAvg可以停滞或发散。[6]这一结果表明，对于某些模型，尤其是在收敛的后期阶段，衰减每轮局部计算量（向更小或更大的方向移动）可能是有用的，同样，衰减学习速率也是有用的。附录A中的图8给出了MNIST CNN的类似实验。有趣的是，对于这个模型，我们没有看到大值的收敛速度有明显的下降。然而，对于下面描述的大规模语言建模任务，我们看到在=1和=5时的性能稍好一些（参见附录A中的图10）。*EEBEEE*

CIFAR实验我们还对CIFAR-10数据集进行了实验[24]，以进一步验证FedAvg。该数据集由10类32x32幅图像和3幅图像组成

RGB通道。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；由于这些数据没有自然的用户分区，因此我们考虑了平衡和IID设置。模型结构取自TensorFlow教程[38]，该教程包括两个卷积层，接着是两个完全连接的层，然后是一个线性转换层，用于生成逻辑图，总共大约有106个参数。注：表3:CIFAR10上达到目标测试集精度相对于基线SGD的回合数和加速。SGD使用了100个小批量。使用FedSGD和FedAvg=0.1，FedAvg使用=5和=50。*CEB*

|  |  |  |  |
| --- | --- | --- | --- |
| 根据。 | 80% | 82% | 85% |
| 新加坡元 | 18000（-） | 31000（-） | 99000（-） |
| 联邦政府 | 3750（4.8×） | 6600（4.7×） | 不适用（-） |
| 费达夫 | 280（64.3×） | 630（49.2×） | 2000年（49.5×） |

最先进的方法对CIFAR的测试准确率为96.5%[19]；然而，我们使用的标准模型足以满足我们的需求，因为我们的目标是评估我们的优化方法，而不是在这项任务上达到最佳的精确度。图像作为训练输入管道的一部分进行预处理，包括将图像裁剪为24x24，随机左右翻转，调整对比度、亮度和白化。

对于这些实验，我们考虑了一个额外的基线，标准的SGD训练在完整的训练集中（没有用户分区），使用大小为100的小批量。在197500次小批量更新之后，我们达到了86%的测试准确率（每个小批量更新都需要在联邦设置中进行一轮通信）。FedAvg在仅进行2000次通信后就达到了85%的类似测试精度。对于所有的算法，除了初始学习速率外，我们还调整了学习速率衰减参数。表3给出了基线SGD、FedSGD和FedAvg达到三个不同精度目标的通信轮数，图4给出了FedAvg与FedSGD的学习速率曲线。

通过对SGD和FedAvg的大小为50的小批量进行实验，我们还可以将精度视为此类小批量梯度计算数量的函数。我们期望SGD在这里做得更好，因为在每次小批量计算之后都会采取一个连续的步骤。然而，正如附录中的图9所示，对于和的适度值，FedAvg在每一个小批量计算中的进度是相似的。此外，我们还发现，标准SGD和FedAvg每轮只有一个客户（C=0）在准确性上都有显著的波动，而平均更多的客户可以消除这一点。*BCE*

大规模LSTM实验我们在一个大规模的下一个单词预测任务上进行了实验，以证明我们的方法在实际问题上的有效性。我们的培训数据集包括来自一个大型社交网络的1000万个公共帖子。我们根据作者对帖子进行了分组，总共有超过500000名客户。此数据集是用户移动设备上显示的文本输入数据类型的实际代理。我们将每个客户数据集限制在最多5000个单词，并在一个由不同（非培训）作者撰写的1e5个帖子组成的测试集中报告准确率（在10000个可能性中，预测概率最高的是正确的下一个单词的数据部分）。我们的

下一个字预测LSTM，非IID数据

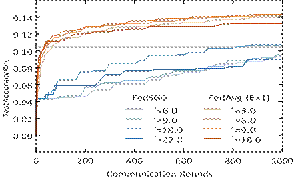


图5：大规模语言模型词LSTM的单调学习曲线。

模型是一个256节点的LSTM，词汇表为10000个单词。每个词的输入和输出嵌入维数为192，并与模型共同训练，共有4950544个参数。我们用10个单词展开。

这些实验需要大量的计算资源，因此我们没有对超参数进行彻底的研究：所有的运行都在每轮200个客户机上进行训练；FedAvg使用了=8和=1。我们研究了FedAvg和基线FedSGD的各种学习率。图5显示了最佳学习率的单调学习曲线。当FedSGD=18.0时，需要820发子弹才能达到10.5%的准确率，而FedAvg=9.0时，只有35发通讯弹（比FedSGD少23倍）就达到了10.5%的准确率。我们观察到FedAvg的测试准确度较低，见附录A中的图10。该图还包括了=5的结果，其表现略差于=1。*BEηηEE*

## 4结论和今后的工作

我们的实验表明，联邦学习是可行的，因为FedAvg使用相对较少的通信轮次来训练高质量的模型，正如在各种模型体系结构上的结果所证明的：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。

虽然联合学习提供了许多实际的隐私好处，但通过差异隐私[14，13，1]，安全多方计算[18]，或它们的组合来提供更强大的保证，是未来工作的一个有趣的方向。注意，这两类技术最自然地应用于FedAvg之类的同步算法

# 工具书类

[1] 马丁·阿巴迪、朱棣文、伊恩·古德费罗、布伦丹·麦克马汉、伊利亚·米罗诺夫、库纳尔·塔尔瓦尔和李章。

深度学习与差异隐私。在2016年第23届ACM计算机和通信安全会议（ACM CCS）上。

[2] 莫妮卡·安德森。技术设备所有权：2015年。http://www.pewinternet.org/2015/10/29/技术-设备-所有权-2015/，

2015.

[3] Yossi Arjevani和Ohad Shamir。分布式凸学习与优化的通信复杂性。神经信息处理系统进展28。2015

[4] Maria Florina Balcan、Avrim Blum、Shai Fine和Yishay Mansour。分布式学习、通信复杂性和隐私性。arXiv预印本arXiv:1204.35142012年。

[5] Yoshua Bengio、Rejean Ducharme、Pascal Vincent和Christian Janvin。一种神经概率语言模型。J、 马赫。学习。2003年。

[6] 基思·博纳维茨、弗拉基米尔·伊万诺夫、本·克鲁特、安东尼奥·马塞东、H.布伦丹·麦克马汉、萨瓦尔·帕特尔、丹尼尔·拉马奇、亚伦·西格尔和卡恩·塞思。基于用户持有数据的联合学习的实用安全聚合。在NIPS私人多方机器学习研讨会，2016年。

[7] 大卫·L·乔姆。无法追踪的电子邮件、回信地址和数字化名。公社。ACM，24（2），1981年。

[8] 陈建民，拉贾特·蒙加，萨米·本吉奥和拉法尔·乔泽福维奇。再论分布式同步sgd。2016年ICLR车间轨道。

[9] 安娜·乔罗曼斯卡、米凯尔·海纳夫、迈克尔·马修、杰拉德·本·阿沃斯和扬·莱昆。多层网络的损耗面。艾斯塔茨，2015年。

[10] 格雷格·科拉多。电脑，回复这封邮件。http地址：//

googleresearch.blogspot.com/2015/

11/电脑回复此邮件。html，2015年11月。

[11] Yann N. Dauphin, Razvan Pascanu, C¸aglar Gulc¸ehre,¨ KyungHyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In NIPS, 2014.

[12] Jeffrey Dean, Greg S. Corrado, Rajat Monga, Kai

Chen, Matthieu Devin, Quoc V. Le, Mark Z. Mao, Marc&apos;Aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, and Andrew Y. Ng. Large scale distributed deep networks. In NIPS, 2012.

[13] John Duchi, Michael I. Jordan, and Martin J. Wainwright. Privacy aware learning. Journal of the Association for Computing Machinery, 2014.

[14] Cynthia Dwork and Aaron Roth. The Algorithmic Foundations of Differential Privacy. Foundations and Trends in Theoretical Computer Science. Now Publishers, 2014.

[15] Olivier Fercoq, Zheng Qu, Peter Richtarik, and Martin´ Takac. Fast distributed coordinate descent for non-´ strongly convex losses. In Machine Learning for Signal Processing (MLSP), 2014 IEEE International Workshop on, 2014.

[16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning. Book in preparation for MIT Press, 2016.

[17] Ian J. Goodfellow, Oriol Vinyals, and Andrew M. Saxe. Qualitatively characterizing neural network optimization problems. In ICLR, 2015.

[18] Slawomir Goryczka, Li Xiong, and Vaidy Sunderam. Secure multiparty aggregation with differential privacy: A comparative study. In Proceedings of the Joint EDBT/ICDT 2013 Workshops, 2013.

[19] Benjamin Graham. Fractional max-pooling. CoRR, abs/1412.6071, 2014. URL http://arxiv.org/ abs/1412.6071.

[20] Sepp Hochreiter and Jurgen Schmidhuber. Long short-¨ term memory. Neural Computation, 9(8), November 1997.

[21] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In ICML, 2015.

[22] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. Character-aware neural language models. CoRR, abs/1508.06615, 2015.

[23] Jakub Konecnˇ y, H. Brendan McMahan, Felix X. Yu,´

Peter Richtarik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. In NIPS Workshop on Private Multi-Party Machine Learning, 2016.

[24] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS. 2012.

[26] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 1998.

[27] Chenxin Ma, Virginia Smith, Martin Jaggi, Michael I Jordan, Peter Richtarik, and Martin Tak´ a´c. Adding vs.ˇ averaging in distributed primal-dual optimization. In ICML, 2015.

[28] Ryan McDonald, Keith Hall, and Gideon Mann. Distributed training strategies for the structured perceptron. In NAACL HLT, 2010.

[29] Natalia Neverova, Christian Wolf, Griffin Lacey, Lex

Fridman, Deepak Chandra, Brandon Barbello, and Graham W. Taylor. Learning human identity from motion patterns. IEEE Access, 4:1810–1820, 2016.

[30] Jacob Poushter. Smartphone ownership and internet usage continues to climb in emerging economies. Pew Research Center Report, 2016.

[31] Daniel Povey, Xiaohui Zhang, and Sanjeev Khudanpur. Parallel training of deep neural networks with natural gradient and parameter averaging. In ICLR Workshop Track, 2015.

[32] William Shakespeare. The Complete Works of William Shakespeare. Publically available at https:

//www.gutenberg.org/ebooks/100.

[33] Ohad Shamir and Nathan Srebro. Distributed stochastic optimization and learning. In Communication, Control, and Computing (Allerton), 2014.

[34] Ohad Shamir, Nathan Srebro, and Tong Zhang. Communication efficient distributed optimization using an approximate newton-type method. arXiv preprint arXiv:1312.7853, 2013.

[35] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In Proceedings of the 22Nd ACM

*SIGSAC Conference on Computer and Communications Security*, CCS &apos;15, 2015.

[36] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. 15, 2014.

[37] Latanya Sweeney. Simple demographics often identify people uniquely. 2000.

[38] TensorFlow team. Tensorflow convolutional neural networks tutorial, 2016. http://www.tensorflow. org/tutorials/deep\_cnn.

[39] White House Report. Consumer data privacy in a networked world: A framework for protecting privacy and promoting innovation in the global digital economy. Journal of Privacy and Confidentiality, 2013.

[40] Tianbao Yang. Trading computation for communication: Distributed stochastic dual coordinate ascent. In Advances in Neural Information Processing Systems, 2013.

[41] Ruiliang Zhang and James Kwok. Asynchronous distributed admm for consensus optimization. In ICML. JMLR Workshop and Conference Proceedings, 2014.

[42] Sixin Zhang, Anna E Choromanska, and Yann LeCun. Deep learning with elastic averaging sgd. In NIPS. 2015.

[43] Yuchen Zhang and Lin Xiao. Communication-efficient distributed optimization of self-concordant empirical loss. arXiv preprint arXiv:1501.00263, 2015.

[44] Yuchen Zhang, Martin J Wainwright, and John C

Duchi. Communication-efficient algorithms for statistical optimization. In NIPS, 2012.

[45] Yuchen Zhang, John Duchi, Michael I Jordan, and Martin J Wainwright. Information-theoretic lower bounds for distributed statistical estimation with communication constraints. In Advances in Neural Information Processing Systems, 2013.

[46] Martin Zinkevich, Markus Weimer, Lihong Li, and Alex J. Smola. Parallelized stochastic gradient descent. In NIPS. 2010.

# A Supplemental Figures and Tables

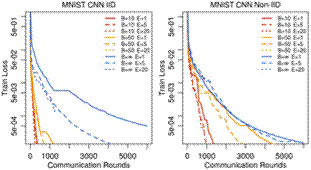


Figure 6: Training set convergence for the MNIST CNN. Note the -axis is on a log scale, and the -axis covers more training than Figure 2. These plots fix = 0.1.*yxC*

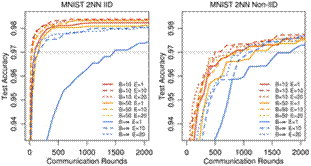


Figure 7: Test set accuracy vs. communication rounds for MNIST 2NN with = 0.1 and optimized . The left column is the IID dataset, and right is the pathological 2digits-per-client non-IID data.*C η*

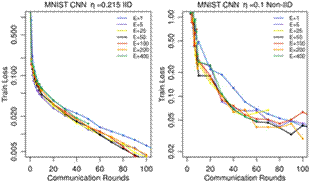


Figure 8: The effect of training for many local epochs (large

*E*) between averaging steps, fixing = 10 and = 0.1. Training loss for the MNIST CNN. Note different learning rates and -axis scales are used due to the difficulty of our pathological non-IID MNIST dataset.*B C y*

Table 4: Speedups in the number of communication rounds to reach a target accuracy of 97% for FedAvg, versus FedSGD (first row) on the MNIST 2NN model.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MNIST 2NN | *E* | *B* | *u* | IID | NON-IID |
| FEDSGD | 1 | ∞ | 1 | 1468 | 1817 |
| FEDAVG | 10 | ∞ | 10 | 156 (9.4×) | 1100 (1.7×) |
| FEDAVG | 1 | 50 | 12 | 144 (10.2×) | 1183 (1.5×) |
| FEDAVG | 20 | ∞ | 20 | 92 (16.0×) | 957 (1.9×) |
| FEDAVG | 1 | 10 | 60 | 92 (16.0×) | 831 (2.2×) |
| FEDAVG | 10 | 50 | 120 | 45 (32.6×) | 881 (2.1×) |
| FEDAVG | 20 | 50 | 240 | 39 (37.6×) | 835 (2.2×) |
| FEDAVG | 10 | 10 | 600 | 34 (43.2×) | 497 (3.7×) |
| FEDAVG | 20 | 10 | 1200 | 32 (45.9×) | 738 (2.5×) |

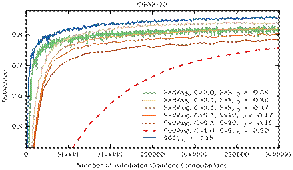


Figure 9: Test accuracy versus number of minibatch gradient computations (B = 50). The baseline is standard sequential SGD, as compared to FedAvg with different client fractions (recall = 0 means one client per round), and different numbers of local epochs .*C C E*

Next Word Prediction LSTM, Non-IID Data

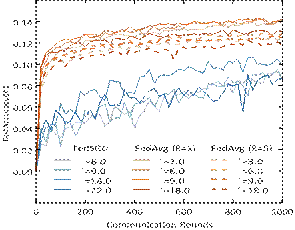


Figure 10: Learning curves for the large-scale language model word LSTM, with evaluation computed every 20 rounds. FedAvg actually performs better with fewer local epochs (1 vs 5), and also has lower variance in accuracy across evaluation rounds compared to FedSGD.*E*

[[1]](" \l "_ftnref1" \o ") For example, if the update is the total gradient of the loss on all of the local data, and the features are a sparse bag-of-words, then the non-zero gradients reveal exactly which words the user has entered on the device. In contrast, the sum of many gradients for a dense model such as a CNN offers a harder target for attackers seeking information about individual training instances (though attacks are still possible).

[[2]](" \l "_ftnref2" \o ") While the batch selection mechanism is different than selecting a batch by choosing individual examples uniformly at random, the batch gradients computed by FedSGD still satisfy E[g] = O(w).*g f*

[[3]](" \l "_ftnref3" \o ") We use the “2NN” multi-layer perceptron described in Section 3.

[[4]](" \l "_ftnref4" \o ") We performed additional experiments on unbalanced versions of these datasets, and found them to in fact be slightly easier for

FedAvg.

[[5]](" \l "_ftnref5" \o ") We always use character to refer to a one byte string, and use role to refer to a part in the play.

[[6]](" \l "_ftnref6" \o ") Note that due to this behavior and because for large not all experiments for all learning rates were run for the full number of rounds, we report results for a fixed learning rate (which perhaps surprisingly was near-optimal across the range of parameters) and without forcing the lines to be monotonic.*E E*

[[7]](" \l "_ftnref7" \o ") Subsequent to this work, Bonawitz et al. [6] introduced an efficient secure aggregation protocol for federated learning, and Konecnˇ y et al.´ [23] presented algorithms for further decreasing communication costs.