IEEE神经网络与学习系统汇刊

# 基于非i.i.d.数据的健壮和高效的联合学习

费利克斯·萨特勒，Simon Wiedemann，Klaus Robert Müller，IEEE成员，和Wojciech Samek，IEEE成员

***摘要***

***索引术语***

稿件于2019年3月6日收到；2019年6月28日修订；2019年9月25日接受。这项工作部分由弗劳恩霍夫学会（Fraunhofer Society）通过MPI FhG合作项目“简化学习机器的理论与实践”提供支持，部分由德国教育和研究部（德国教育和研究部）作为第01IS14013A号拨款项下的柏林大数据中心提供支持，部分由柏林机器学习中心（Berlin Center for Machine Learning）根据第01IS18037I号拨款提供支持，部分由DFG根据赠款EXC 2046/1和赠款390685689提供，部分由韩国政府根据赠款2017-0-00451提供的信息和通信技术规划与评估（IITP）拨款提供。（通讯作者：克劳斯·罗伯特·穆勒；沃伊切赫·萨梅克）

F、 Sattler、S.Wiedemann和W.Samek在德国柏林10587号的Fraunhofer-Heinrich Hertz Institute工作（电子邮件：沃伊切赫.萨梅克@ 弗劳恩霍夫).

K、 -R.Müller在柏林理工大学工作，地址：10587柏林，德国，马克斯普朗克信息学研究所，66123 Saarbrucken，Germany，以及韩国首尔136-713韩国大学大脑和认知工程系（电子邮箱：klaus robert）。mueller@tu-berlin.de).

这篇文章的补充资料可从http://ieeexplore.ieee.org，作者提供。

本文中一个或多个图形的彩色版本可在http://ieeexplore.ieee.org。

数字对象标识符10.1109/TNNLS.2019.2944481

## 一、 简介

|  |
| --- |
| **T** |

目前有三大发展正在改变数据的创建和处理方式：首先，随着物联网（IoT）的出现，世界上智能设备的数量在过去几年里迅速增长。其中许多设备都配备了各种传感器和越来越强大的硬件，使它们能够以前所未有的规模收集和处理数据[1]–[3]。

在并行开发中，深度学习彻底改变了从数据资源中提取信息的方式，在计算机视觉、自然语言处理或语音识别等领域取得了突破性的成功[4]-[9]。随着数据量的不断增长，深度学习的规模越来越大，它在最近取得的惊人成功至少可以部分归因于为培训提供的非常大的数据集。因此，在利用物联网设备提供的丰富数据进行培训和改进深度学习模型方面有着巨大的潜力[10]。

与此同时，数据隐私问题也越来越受到许多用户的关注。近年来发生的多起数据泄露和误用案例表明，数据的集中处理给最终用户隐私带来了很大的风险。由于物联网设备通常在私有环境中收集数据，通常甚至没有用户的明确意识，这些关注点尤其强烈。因此，通常情况下，与能够进行深度学习模型培训的集中实体共享这些数据是不可取的。在其他情况下，由于其他原因（例如本地代理的自治性增强），可能需要对数据进行本地处理。

这让我们面临以下困境：如果这些数据不能存储在一个集中的位置，我们将如何利用数百万物联网设备的丰富组合数据来训练深度学习模型？

|  |
| --- |
| 本作品由Creative Commons Attribution 4.0许可证授权。有关详细信息，请参阅http://creativecommons.org/licenses/by/4.0/ |

联合学习解决了这个问题，因为它允许多个参与方在他们的组合数据上联合训练一个深度学习模型，而不需要任何参与者将他们的数据透露给一个集中的服务器[10]。这种形式的隐私保护协作学习是通过遵循图1所示的简单的三步协议来实现的。在第一步中，所有参与的客户机从服务器下载最新的主模型W。接下来，客户利用随机梯度下降（SGD）改进下载的模型，基于他们的局部训练数据。最后，所有参与的客户机将其本地改进的模型Wback上载到服务器，在服务器上收集和聚合这些模型以形成一个新的主模型*我*

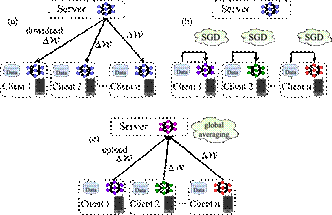


图1。带参数服务器的联合学习。文中给出了一种分布式SGD通信方案。（a） 客户端与服务器同步。（b） 客户机根据其本地数据独立计算权重更新。（c） 客户机将其本地权重更新上载到服务器，在服务器上对它们进行平均以生成新的主模型。

（实际上，权重更新ww可以传递，而不是完整的模型，只要所有客户端保持同步，这是等效的）。重复这些步骤，直到满足一定的收敛准则。请注意，当遵循此协议时，训练数据永远不会离开本地设备，因为只有模型更新被传递。尽管已经表明，在对抗性设置中，仍然可以从这些更新[11]中推断出关于训练数据的信息，但是可以应用诸如更新的同态加密[12]、[13]或差异私有训练[14]之类的附加机制来完全隐藏关于本地数据的任何信息。古老的

联合学习中的一个主要问题是发送模型更新时产生的大量通信开销。当天真地遵循前面描述的协议时，每个参与的客户机都必须在每次训练迭代期间通信完整的模型更新。每一次这样的更新都与经过训练的模型大小相同，对于具有数百万参数的现代架构来说，其大小可能在千兆字节的范围内[15]，[16]。在大数据集的数十万次训练迭代过程中，每个客户机的总通信量很容易增长到1 PB以上[17]。因此，如果通信带宽有限或通信成本高昂（天真），联合学习可能变得毫无成效，甚至完全不可行。

每个客户机在培训期间必须上传和下载的位的总数由

蟾蜍∈Oiter××|*/（N）f*|×上下+η）*/)*

|  |
| --- |
|  |

                                    #更新更新大小

（一）

其中，Niter是每个客户机执行的训练迭代（向前-向后传递）的总数，f是通信频率，| W|是模型的大小，Hup*/*向下*)*分别是上传和下载期间交换的权重更新的熵，以及*η*是编码的低效率，即真实更新大小和最小更新大小（由熵给出）之间的差异。如果我们假设模型的大小和训练迭代次数是固定的（例如，因为我们想在给定的任务上达到一定的精度），这就给我们留下了三个减少通信的选项：1）我们可以减少通信频率f；2）减少权重更新Hup的熵*/*向下*)*通过有损压缩方案；和/或3）使用更有效的编码来传达权重更新，从而减少。*η*

## 二。联合学习环境的挑战

在我们考虑减少通信量的方法之前，我们首先必须考虑到联合学习与其他分布式训练设置（如并行训练）的区别的独特特性（也可与[10]进行比较）。在联合学习中，训练数据和计算资源的分配是学习环境的一个基本的固定属性。这将带来以下挑战。

1） 由于单个客户机上的培训数据是由客户机根据其本地环境和使用模式收集的，因此本地数据集的大小和分布在不同的客户机之间通常会有很大的差异。*不平衡和非i.i.d.数据：*

2） 联合学习环境可能由数百万参与者组成[18]。此外，由于协作学习模型的质量取决于所有客户的综合可用数据，因此协作学习环境将具有自然增长的趋势。*大量客户：*

3） 一旦客户机数量增长超过某个阈值，权重更新的直接通信就变得不可行，因为通信和聚合更新的工作量都随着客户端的数量线性增长。因此，在联合学习中，通过中间参数服务器进行通信是不可避免的。这将每个客户端的通信量和通信轮数减少到一次本地权重更新上载和一次从服务器下载聚合更新，并将聚合工作负载从客户端移开。然而，通过参数服务器进行通信给高效的分布式训练带来了额外的挑战，因为现在上传到服务器和从服务器下载都需要压缩，以减少通信时间和能耗。*参数服务器：*

4） 在一般的物联网联合学习设置中，通常不能保证所有的客户端都参与到每一轮的通信中。设备可能会失去连接、电池耗尽或因其他原因无法参与协作培训。*部分参与：*

5） 移动设备和嵌入式设备通常不连接到电网。*电池和内存有限：*

## 表一

高效分布式通信的不同方法

文献中提出的深度学习。没有

现有方法满足所有要求–（R3）**（R1）**

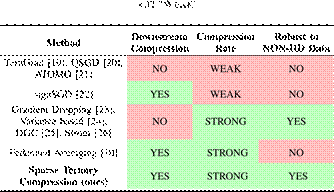
联邦学习环境。我们打电话

如果

联合训练的收敛与客户端数据的本地分布无关。

我们称压缩率更高

大于×32“强”且小于或等于



相反，它们运行计算的能力受到有限电池的限制。众所周知，对于深度神经网络来说，执行SGD迭代是非常昂贵的。因此，有必要保持每个客户梯度评估的数量尽可能少。移动设备和嵌入式设备通常也只有非常有限的内存。由于SGD的内存占用随着批处理大小呈线性增长，这可能会迫使设备在非常小的批处理大小上进行训练。

基于联邦学习环境的上述特征，我们得出结论：一个通信效率高的联邦学习分布式训练算法需要满足以下要求。

*（R1）*：它应该压缩上行和下行通信。

*（R2）*：它应该对非i.i.d.，小批量和不平衡数据具有鲁棒性。

*（R3）*：对于大量客户和部分客户参与，它应该是健壮的。

## 三、 贡献

在本文中，我们将演示为通信高效的联邦学习而提出的现有方法中没有一种能够满足所有这些需求（见表1）。更具体地说，我们将证明，能够压缩上游和下游通信的方法对非i.i.d.数据分布非常敏感，而对此类数据更稳健的方法不会压缩下游（见第五节）。然后，我们将着手为联邦学习构建一个新的高效通信协议，解决这些问题并满足所有要求（R1）–（R3）。我们对我们的方法进行了收敛性分析，并对四种不同的神经网络结构和数据集进行了大量的实验，结果表明稀疏三值压缩（STC）协议优于现有的压缩方案，因为它需要较少的梯度估计和通信比特来收敛达到给定的目标精度（见第九节）。这些结果也适用于i.i.d.制度。

## 四、 相关工作

在更广泛的通信高效分布式深度学习领域，人们提出了各种各样的方法来减少训练过程中的通信量。以（1）为参考，我们可以将现有的关于交流高效分布式深度学习的研究机构分成三个不同的小组。

1） 方法降低通信频率f。McMahan等人。[10] 提出联邦平均法，在这种情况下，每个客户机执行多个SGD迭代来计算权重更新，而不是在每次迭代后进行通信。McMahan等人。观察到在不同的卷积和递归神经网络结构中，只要数据以i.i.d.方式分布在客户机之间，通信就可以延迟100次迭代，而不会显著影响收敛速度。使用更长的延迟时间，通信量甚至可以进一步减少；然而，这是以增加梯度评估的数量为代价的。在后续研究中，Konecˇn`y等人。[27]将这种通信延迟与随机稀疏化和概率量化相结合。它们限制客户端学习随机稀疏权重更新，或者在之后强制随机稀疏更新（“结构化”与“草图”更新），并将这种稀疏化与概率量化相结合。然而，在SGD迭代方面，他们的方法显著降低了收敛速度。实践证明，该方法既能减少上游客户的参与，又能减少下游客户的通信延迟。**通信延迟**

（二）**稀疏化**方法通过限制参数的一小部分来减少更新的熵H。Strom[24]提出了一种方法（后来被[26]修改），其中只将幅度大于某个预定义阈值的梯度发送到服务器。所有其他梯度都是在残差中累积的。在声学建模任务中，这种方法可以实现高达三个数量级的上游压缩率。然而，在实践中，很难为阈值选择合适的值，因为对于不同的架构甚至不同的层，阈值可能会有很大的变化。为了克服这个问题，Aji和Heafield[23]反而修正了稀疏率，并且只与每个梯度的最大幅度的分数p条目进行通信，同时还收集了残差中的所有其他梯度。以p的稀疏率=0001，他们的方法只是稍微降低了训练模型的收敛速度和最终精度。Lin等人。[25]对Aji和Heafield[23]的工作进行了一些小的修改，甚至缩小了这个小的性能差距。稀疏化方法的提出主要是为了加速数据中心的并行训练。在联合学习环境中，他们的收敛性还没有得到更多的研究。稀疏化方法（以其现有形式）主要压缩上游通信，因为来自不同客户端的更新的稀疏模式通常会有所不同。如果参与的客户机数量大于反向稀疏率（在联邦学习中很容易出现这种情况），那么下游更新甚至根本不会被压缩。*.*

3） 方法通过将所有更新限制为一组减少的值来减少权重更新的熵。伯恩斯坦等人。[22]提出signSGD，一种对i.i.d.数据具有理论收敛性保证的压缩方法，它将每个梯度更新量化为其二进制符号，从而将每次更新的比特大小减少一倍×32。signSGD还合并了下载压缩，它通过多数投票的方式聚合来自所有客户端的二进制更新。其他作者建议以无偏的方式随机量化上传过程中的梯度（TernGrad[19]，量子化随机梯度下降（QSGD）[20]，ATOMO[21]）。这些方法在理论上很有吸引力，因为它们在相对温和的假设下继承了正则SGD的收敛特性。然而，它们的经验性能和压缩率与稀疏化方法不匹配。**密集量子化**

在上面列出的所有方法中，只有联邦平均和signSGD压缩了上行和下行通信。在第二节中定义的联邦学习设置中，所有其他方法的效用都是有限的，因为它们不压缩从服务器到客户机的通信。

*符号：*在下文中，手写体W表示神经网络的全部参数，而正则大写W表示W内参数的一个特定张量，小写表示网络的单个标量参数。神经网络参数之间的算术运算要基本理解。*w*

五、 现有压缩方法的局限性

关于高效分布式深度学习的相关工作几乎只考虑客户之间的i.i.d.数据分布，即他们假设局部梯度相对于整个批次梯度的无偏性，根据

                     E∼[∇WW]=∇W∀=1（2）*十圆周率我（十）,)R()我,..,n*

其中pi是第i个客户的数据分布，rw是组合培训数据的经验风险函数。*()*

而这种假设对于并行培训来说是合理的，因为客户之间的数据分布是由

0我们用VGG11\*表示[28]中描述的原始VGG11架构的简化版本，其中删除了所有丢失和批处理规范化层，卷积滤波器的数量和所有完全连接层的大小减少了2倍。

对于实践者来说，这在联合学习环境中通常是无效的，在这种环境中，我们通常只希望在平均值上是无偏的

*n*

*n*1            E*十我*∼*p我*[∇W*我（十）我,*W*)*] = ∇W*R(*W*)*（三）

*我*=1

而个别客户的梯度将偏向于根据

E*十**,..,n.*

（四）

由于它违反了假设（2），局部数据的非i.i.d.分布使得[19]-[21]和[29]中提出的现有收敛保证不适用，并对通信高效分布式训练算法的实际性能产生了显著影响，我们将在下面的实验中证明。

### A、 初步实验

我们对经过充分研究的11层VGG11网络的简化版本进行了初步实验[28]，我们在联邦学习设置中使用10个客户机对CIFAR-10[30]数据集进行培训。对于i.i.d.设置，我们将训练数据随机分成大小相等的碎片，并为每个客户分配一个碎片。对于“非i.i.d.（m）”设置，我们从数据集的m个类中分配每个客户机样本。数据分割是非重叠和平衡的，这样每个客户机最终都有相同数量的数据点。生成数据分割的详细程序在补充资料附录的B节中描述。我们还使用一个简单的logistic回归分类器进行了实验，我们在相同的联邦学习环境下对MNIST数据集进行训练[31]。两个模型都是使用动量SGD训练的。为了使结果具有可比性，所有压缩方法都使用相同的学习速率和批处理大小。

### B、 结果

图2示出了当使用不同的通信有效联邦学习方法训练时，两个模型在梯度评估方面的收敛速度。我们观察到，虽然所有压缩方法在i.i.d.数据的梯度评估方面实现了相当快的收敛，与未压缩的基线（黑线）非常匹配，但它们在非i.i.d.训练设置下会受到很大的影响。由于在logistic回归模型中也可以观察到这种趋势，我们可以得出结论，潜在的现象并不是深层神经网络所独有的，而且还会传导到凸目标上。现在我们将针对不同的压缩方法详细分析这些结果。

*（一）联合平均：*最值得注意的是，联邦平均[10]（参见图2中的橙色线）虽然是专门针对联邦学习设置而提出的，但在很大程度上受到非i.i.d.数据的影响。这一观察结果与赵等一致。[32]世卫组织证明，在非i.i.d.学习环境中，模型精度可下降高达55%

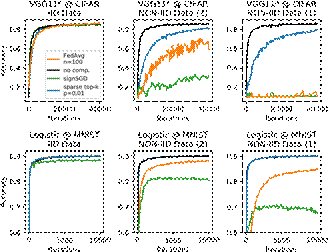


图2。使用不同压缩方法训练VGG11\*在CIFAR-10上的收敛速度，以及在MNIST和时尚MNIST上使用逻辑回归的过程中使用不同的压缩方法时的收敛速度，该环境下有10个客户用于i.i.d.和非i.i.d.数据。在非i.i.d.的情况下，每个客户机只保存数据集中10个类中的两个分别对应的示例。在非i.i.d.情况下，所有压缩方法的收敛速度都会下降，而稀疏top-k算法受影响最小。2

与身份证号码比较。他们将准确性的下降归因于客户之间权重差异的增加，并建议通过为所有客户分配一个共享的公共i.i.d.数据集来回避这个问题。虽然这种方法确实可以创建更精确的模型，但它也有多个缺点，最关键的一个缺点是，我们一般不能假设这样一个公共数据集的可用性。如果存在一个公共数据集，可以使用它在服务器上预训练模型，这与联邦学习中通常所做的假设不一致。此外，如果所有客户机共享（部分）同一个公共数据集，则过度适应此共享数据可能会成为一个严重问题。这种影响在高度分布式的环境中尤其严重，因为每个客户端上的数据点数量很少。最后，即使在客户之间共享一个相对较大的数据集，也无法完全恢复在i.i.d.情况下获得的原始精度。基于这些原因，我们认为[32]提出的数据共享策略对于解决联邦平均在非i.i.d.数据上存在收敛问题的根本问题是不够的。

*（二）标牌：*量化方法signSGD[29]（参见图2中的绿线）在非i.i.d.学习环境中遭受更糟糕的稳定性问题。该方法完全不能收敛于CIFAR基准，即使对于凸logistic回归目标，训练平台的精度也大大降低。

为了理解这些收敛性问题的原因，我们必须研究单个批次梯度出现“正确”符号的可能性有多大。让

*k*

### g=wk1 WL（席，W）（5）K

*我*=1

是在参数Dk={}⊂大小为k的特定小批量数据上的批渐变。让，更进一步，g*十*1*,..., xk公司Dww*

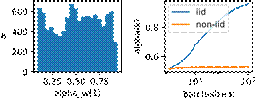


图3。左：MNIST数据集上逻辑回归权重层1的值分布。右图：用于增加批量大小的开发。在i.i.d.案例中，批次是从培训数据中随机抽取的，而在非i.i.d.案例中，每个批次只包含一个班级的样本。对于i.i.d.批次，随着批次大小的增加，梯度符号变得越来越精确。对于非i.i.d.批量数据，情况并非如此。无论批次有多大，梯度符号与整个批次的梯度高度不一致。*αw()α(k)*

是整个训练数据的梯度，然后，我们可以通过

|  |  |
| --- | --- |
| 我们也可以计算平均统计量*.* | （六） |

（七）

估计网络所有参数的平均同余。

图3（左）示例性示出了训练开始时MNIST上logistic回归权重内1的值的分布。如我们所见，在批大小为1的情况下，gis是一个非常糟糕的预测真梯度符号，方差非常高，平均同余为1=051，略高于随机数。一旦我们检查了增加批量的梯度符号同余的发展，signSGD对非i.i.d.数据的敏感性变得明显。图3（右）显示了从i.i.d.和noni.i.d.分布中取样的成批增加的情况。对于后者，每个采样批次只包含来自一个类的数据。随着数据量的增加，我们可以看到，随着数据量的不断增加，i.d。然而，对于非i.i.d.数据，一致性保持较低，与批次大小无关。这意味着，如果客户持有高度非i.i.d.数据子集，则signSGD更新只会与最陡下降方向弱相关，无论选择多大的批量进行培训。*αw()w*1 *α().α*

*3） Top-k稀疏化：*在所有现有的压缩方法中，top-k稀疏化（见图2中的蓝线）受非i.i.d.数据的影响最小。对于CIFAR上的VGG11，即使每个客户机只保存一个类的数据，训练仍然可靠地收敛；对于在MNIST上训练的logistic回归分类器，收敛速度丝毫没有减慢。我们假设这种对非i.i.d.数据的鲁棒性主要是由于两个原因。首先，客户端之间频繁的权重更新通信阻止了

它们彼此相差太远，因此top-k稀疏化不会像联邦平均那样受到权重发散的影响[32]。其次，稀疏化不会像signSGD那样破坏训练的稳定性，因为随机梯度中的噪声不会被量化放大。尽管top-k稀疏化在非i.i.d.数据上表现出了很好的性能，但是它在联邦学习环境中的效用有限，因为它只直接压缩上游通信。

表一总结了我们的发现。现有的压缩方法都不支持下载压缩和非i.i.d.数据。

## 六、 稀疏三值压缩

Top-k稀疏化在使用非i.i.d.客户端数据的分布式学习环境中显示了最有前途的性能。我们将以此为出发点，为联合学习构建一个有效的通信协议。为了达成这个协议，我们将解决三个阻碍top-k稀疏化直接应用于联合学习的开放性问题。

1） 我们将通过量化和最佳无损编码进一步提高我们的方法的效率

权重更新。

2） 我们将把下游压缩合并到方法中，以允许从服务器到客户端的有效通信。

3） 我们将实现一个缓存机制，以在部分客户端参与的情况下保持客户端同步。

### A、 三元化权重更新

如[23]和[25]所建议的，常规top-k稀疏化以完全精确的方式传达最大元素的分数，而其他元素则完全不进行通信。在我们之前的工作中（Sattler等人。[17] ），我们已经证明，这种更新精度的不平衡在分布式训练环境中是浪费的，并且当稀疏化与非零元素的量化相结合时，可以获得更高的压缩增益。

我们采用[17]中描述的方法来进行联邦学习设置，并将稀疏化更新的剩余top-k元素量化为平均总体大小，留下一个包含值{−µ，0}的三元张量。量化方法在算法1中被形式化。*,µ*



**算法1**STC公司



**1输入：**平坦张量T∈R，稀疏p*n*

**2输出：**稀疏三元张量T∈{−µ，0}←max1←topk |）∗*,µnk（np）,)五(T*

**5**·掩模←（| |≥v）∈{01}*T,n*

**6** ·←掩模*T蒙面的T*

**7** ·µ←|\*←=1 |×符号|*k*1*镍µT我假装（吨）蒙面的)*

**8返回***T*

此三元化步骤减少了

*H*稀疏= −*p*日志2*（p)*− (1 −*p)*日志2*（p)*+ 32便士（8）

到

*H*STC公司= −*p*日志2*（p)*− (1 −*p)*日志2*（p)*+ *p*（九）

与常规稀疏化相比。在p=001的稀疏率下，通过三元化实现的额外压缩是Hsparse=4414。为了通过纯稀疏化获得相同的压缩增益，必须将稀疏率提高大约相同的因子。*./H*STC公司*.*

使用Stich等人开发的理论框架。[33]，在损失函数的标准假设下，我们可以证明STC的收敛性。这个证明依赖于限制由压缩算子引起的扰动的影响。这在下面的定义中被正式化。

*定义1（k-收缩）[33]：*对于参数

0 *k<k*≤*d*，k-收缩是一个算子comp：R*d*→右侧*d*

满足收缩性质

E*十*补偿         ∀∈R（10）k d*十*2 *十d.*

我们可以证明STC确实是一个k-收缩。

*引理2：*算法1中定义的STCk是

*k*●收缩，带

                                  0 *<k*˜ =托普克*(十*2*)*12 *d*≤*d.* （十一）

*克朗*2

证据可在补充材料的附录E中找到。然后直接从[33，Th。2.4]对于任何L-smooth，*µ*-梯度有界的强凸目标函数f，更新规则

（十二）

                           := +*它*−

（十三）

根据

[ ]− + *d*2 *G*2升+ *d*33 *G*2 *k*˜*. 微特微特*2 µ*T*3

（十四）

按比例这意味着forO，这与普通新加坡元是一样的！∈O≯）（（，STC收敛*（克*2*/微特))T（d）/k我/µ))*1*/*2*)*

初步实验符合我们的理论

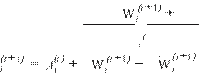
调查结果。图4显示了VGG11\*模型在不同稀疏度水平上训练时的最终精度，有无国际化。正如我们所看到的，额外的国际化对收敛速度的影响很小，有时甚至会提高训练模型的最终精度。显然，稀疏性和量化的结合比纯粹的稀疏化更有效地利用预算的通信。

### B、 延伸至下游压缩

现有的针对分布式训练的压缩框架（见[19]、[20]、[23]、[25]）只压缩从客户端到服务器的通信，这对于通过all-reduce操作实现聚合的应用来说已经足够了。然而，在联邦学习设置中，客户端必须从服务器下载聚合权重更新，这种方法不可行，因为它会导致通信瓶颈。

作为映射（压平）权重的压缩运算符，为了说明这一点，让STCk:→→W update W to a sparsized and ternalized weight update？W根据算法1。对于局部权重更新，STC的更新规则可以写为

*n*

W*（吨）*STCk（15）=1瓦∮+1*t A(我t)我)*

*A*（十六）

从一个空的剩余Ai=0开始∈Ron所有客户机。*(*0*)n*

虽然从客户端发送到服务器的更新W+总是稀疏的，但是在最坏的情况下，更新W+1中的非零元素的数量与参与的客户端的数量有关。如果参与率超过逆稀疏度1，则更新˜*(它*1*)(t)*向下游输送的水呈线性增长*/p*

为了解决这个问题，我们建议在客户端和服务器端使用相同的com-+1压缩机制来压缩下游通信。这会将更新规则修改为*(t)*本质上变得稠密。

        ˜ = *n*

*(t*+1*)*STC公司*k*1个STC*k                                   A（吨）)*

### 西北

                                       =1瓦∮+1*我我(t)*

（十七）

有客户端和服务器端的剩余更新

*A*W˜*我(t*+1*)*（十八）

*A**.* （十九）

我们可以将上传和下载压缩（17）的这个新更新规则表示为带有广义滤波器掩码的纯上传压缩（15）的一个特例。设Mi，i=1b为上传期间各客户端使用的稀疏过滤器掩码，M为服务器在下载期间使用的过滤掩码。然后，我们可以到达*,..,n*

相同的稀疏更新m∮=®Wis是Hadamard产品。因此，如果所有客户机都使用过滤器掩码*我米我(t)*

*米*，其中预测使用这种新更新规则的训练模型的行为应该类似于常规的仅上游稀疏化，但稀疏率略有增加。我们通过实验验证了这一预测：

图5显示了VGG11在CIFAR10上实现的精度，当在联邦学习环境中训练时，五个客户端以不同的上传和下载压缩速率进行10000次迭代。可以看出，只要下载稀疏度和上传稀疏度是同一个阶数，在i.i.d.和非i.i.d.两种情况下，稀疏化下载对收敛性的影响并不会太大，而且会使准确度降低至多2%。

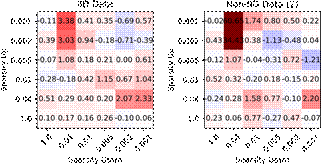


图4。不同上传和下载稀疏度的国际化效果。显示的是使用稀疏更新训练的模型和使用稀疏二进制更新训练的模型之间的最终精度差（以百分比为单位）。正数表示用纯稀疏性训练的模型有更好的性能。VGG11在CIFAR10上接受了16000次迭代的培训，其中五个客户持有身份证和非身份证数据。

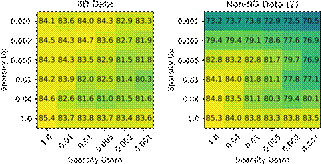


图5。VGG11\*在分布式环境中接受CIFAR培训，在不同级别的上传和下载稀疏度下进行16000次迭代，可获得精确性。与仅使用上载稀疏性相比，为下游通信稀疏化更新最多可降低3%的最终精度。

### C、 部分客户端参与的权重更新缓存

到目前为止，我们只考虑了所有客户参与整个培训过程的场景。然而，正如第二节所阐述的，在联合学习中，通常只有整个客户群体的一小部分会参与任何特定的沟通回合。由于客户机不下载完整的型号W，而只下载压缩型号更新W∮（；这在保持所有客户机同步方面带来了新的挑战。*（吨）)t)*

为了解决同步问题，减少客户端的工作负载，我们建议在服务器端使用缓存机制。假设最后*τ*通讯回合已经产生了更新*,..., T*−τ}. 服务器可以将这些更新的所有部分和与全局模型一起缓存到某个点{=}。*P（第)*˜*(Tt)*

每个希望参与下一轮通信的客户机必须首先通过下载Por W与服务器同步，这取决于它跳过了多少次前一轮通信。对于一般稀疏更新，熵的界*（第)（吨）)*

*H*（二十）

可以达到。这意味着下载的大小将随着客户端跳过的轮数线性增长

培训。平均跳过轮数等于反向参与分数1。这通常是可以容忍的，因为下行链路通常比上行链路更便宜并且具有更高的带宽，如[10]和[19]中所述。本质上，所有只传递参数更新而不是完整模型的压缩方法都会遇到同样的问题。signSGD也是如此，尽管在这里，下游更新的大小只随延迟时间对数增长，根据*/η*

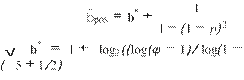
*H（P标牌GD(τ))*≤日志2*(*2*τ*+ 1*).*（二十一）

部分客户参与也会影响联合训练的收敛速度，包括延迟更新和稀疏更新。我们将在第VII-C节详细研究这些影响。

### D、 无损编码

为了传递STC产生的一组稀疏三元张量，我们只需要传输平坦张量中非零元素的位置，以及每次非零更新一位来指示平均符号或−µ。与其传达非零元素的绝对位置，不如传达它们之间的距离。假设一个随机稀疏模式，我们知道对于| |和k=| |的大值，距离近似几何分布，成功概率等于稀疏率p。因此，我们可以使用Golomb代码对距离进行最佳编码[34]。Golomb编码将平均位置位数减少到*µWpW*

**b**∗（二十二）

p和=是黄金比例。对于稀疏率，例如p=001，我们得到b´pos=838，这转换成×19压缩，与16个固定比特的朴素距离编码相比。编码和解码方案可在补充材料附录（算法A1和A2）的A节中找到。更新在上传和下载之前都会被编码。*)))φ...*

算法2描述了一个完整的压缩框架，该框架通过稀疏化、三元化和更新的最佳编码实现了上游和下游的压缩。

## 七。实验

我们在四种不同的学习任务上评估了我们提出的通信协议，并将其与联邦平均和signSGD在各种不同的联邦学习环境中的性能进行了比较。

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

*模型和数据集：*为了涵盖广泛的学习问题，我们对不同大小的卷积和递归神经网络进行评估，用于图像分类和语音识别的相关联邦学习任务：

**算法2**基于STC的参数服务器联合学习算法



**1输入：**初始参数W

**2输出：**改进参数W

**3初始：**所有客户机Ci，i=1[客户机数量]都使用相同的参数W←W.Every初始化*,..,我*

客户端持有不同的数据集Di，其中

|{：（∈}|=[每个客户的班级数]，大小为| |=ψ|∪|。残差初始化为零*是的十, 是的)Di公司Di公司我j Dj*

WRR←0。*,我,*

**4个待办事项***t*=1*,.., T*

**5平行do5平行do***我*∈⊆{1[客户数量]}*它,..,*

**6**客户机Ci：

**7**·消息←下载→消息*Ci()*

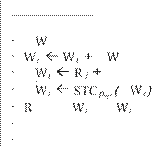
**8**←解码消息*()*

**9**

**10**新加坡元*(我, Di公司,b)我*

**11**

**12***我*← − ˜

**13**msgi公司←编码*(*W˜*我)*

**14**上传→消息*C我S(我)*

**15结束15结束**

**16**服务器可以：

**1718** ··聚集W R1 W？呢*Ci*→*S*˜*我)，我我*∈*t It我*

**19** ·←（W*)*

**20           21**·← − ˜ · ← + ˜

**22**·消息←编码∮W*()*

**23**·广播→消息=1*Ci()，我,.., 米*

**24结束**

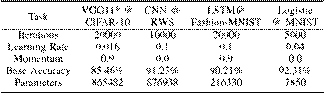
**25返回**W

*CIFAR上的VGG11\**: We train a modified version of the popular 11-layer VGG11 network [28] on the CIFAR [30] data set. We simplify the VGG11 architecture by reducing the number of convolutional filters to [32, 64, 128, 128, 128, 128, 128, 128] in the respective convolutional layers and reducing the size of the hidden fully-connected layers to 128. We also remove all dropout layers and batch-normalization layers as the regularization is no longer required. Batch normalization has been observed to perform very poorly with both small batch sizes and non-i.i.d. data [35], and we do not want this effect to obscure the investigated behavior. The resulting VGG11\* network still achieves 85.46% accuracy on the validation set after 20000 iterations of training with a constant learning rate of 0.16 and contains 865482 parameters.

*CNN on KWS*: We train the four-layer convolutional neural network (CNN) from [27] on the speech commands data set [36]. The speech commands data set consists of 51088 different speech samples of specific keywords. There are 30 different keywords in total, and every speech sample is of 1-s duration. Like [32], we restrict us to the subset of the ten most common keywords. For every speech command, we extract the Mel spectrogram from the short-time Fourier transform,

## TABLE II

MODELS AND HYPERPARAMETERS. THE LEARNING RATE IS KEPT CONSTANT THROUGHOUT TRAINING



which results in a 32×32 feature map. The CNN architecture achieves 89.12% accuracy after 10000 training iterations and has 876938 parameters in total.

*LSTM on Fashion-MNIST*: We also train an Long ShortTerm Memory (LSTM) network with two hidden layers of size 128 on the Fashion-MNIST data set [37]. The FashionMNIST data set contains 60000 train and 10000 validation greyscale images of ten different fashion items. Every 28×28 image is treated as a sequence of 28 features of dimensionality 28 and fed as such in the many-to-one LSTM network. After 20000 training iterations with a learning rate of 0.04, the LSTM model achieves 90.21% accuracy on the validation set. The model contains 216330 parameters.

*Logistic Regression on MNIST*: Finally, we also train a simple logistic regression classifier on the MNIST [31] data set. The MNIST data set contains 60000 training and 10000 test greyscale images of handwritten digits of size 28 × 28. The trained logistic regression classifier achieves 92.31% accuracy on the test set and contains 7850 parameters.

The different learning tasks are summarized in Table II. In the following, we will primarily discuss the results for VGG11\* trained on CIFAR; however, the described phenomena carry over to all other benchmarks and the supporting experimental results can be found in the Appendix in the Supplementary Material.

*Compression Methods*: We compare our proposed STC method at a sparsity rate of p = 1400 with federated averaging at an “equivalent” delay period of n = 400 iterations and signSGD with a coordinatewise step size of = 00002. At a sparsity rate of p = 1400, STC compresses updates both during upload and download by roughly a factor of ×1050. A delay period of n = 400 iterations for federated averaging results in a slightly smaller compression rate of ×400. Further analysis on the effects of the sparsity rate p and delay period n on the convergence speed of STC and federated averaging can be found in Section C of the Appendix in the Supplementary Material. During our experiments, we keep all training related hyperparameters constant for the different compression methods. To be able to compare the different methods in a fair way, all methods are given the same budged of training iterations in the following experiments (one communication round of federated averaging uses up n iterations, where n is the number of local iterations).*/δ ./*

*Learning Environment:* The federated learning environment described in Algorithm 2 can be fully characterized by five parameters. For the base configuration, we set the number of clients to 100, the participation ratio to 10%, and the

## TABLE III

BASE CONFIGURATION OF THE FEDERATED LEARNING

ENVIRONMENT IN OUR EXPERIMENTS



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

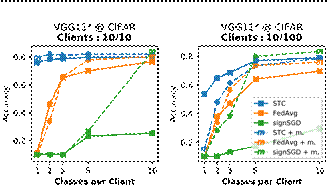


Fig. 6. Robustness of different compression methods to the non-i.i.d.-ness of client data on four different benchmarks. VGG11\* trained on CIFAR. STC distinctively outperforms federated averaging on non-i.i.d. data. The learning environment is configured as described in Table III. Dashed lines signify that a momentum of m = 09 was used.*.*

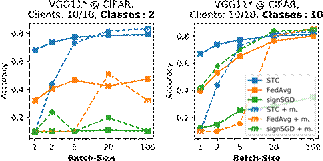


Fig. 7. Maximum accuracy achieved by the different compression methods when training VGG11\* on CIFAR for 20000 iterations at varying batch sizes in a federated learning environment with ten clients and full participation. Left: Every client holds data from exactly two different classes. Right: Every client holds an i.i.d. subset of data.

local batch size to 20 and assign every client an equally sized subset of the training data containing samples from ten different classes. In the following experiments, if not explicitly signified otherwise, all hyperparameters will default to this base configuration summarized in Table III. We will use the short notations “Clients: /N” and “Classes: c” to refer to a setup of the federated learning environment in which a random subset of out of a total of N clients participates in every communication round and every client is holding data from exactly c different classes.*ηNηN*

### A. Momentum in Federated Optimization

We start out by investigating the effects of momentum optimization on the convergence behavior of the different compression methods. Figs. 6–9 show the final accuracy achieved by federated averaging (n = 400), STC (p = 1400), and signSGD after 20000 training iterations in a variety of different federated learning environments. In Figs. 6–9, dashed*/*

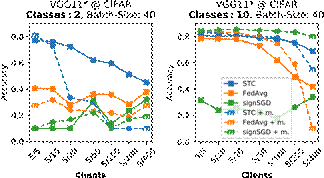


Fig. 8. Validation accuracy achieved by VGG11\* on CIFAR after 20000 iterations of communication-efficient federated training with different compression methods. The relative client participation fraction is varied between 100% (5/5) and 5% (5/100). Left: Every client holds data from exactly two different classes. Right: Every client holds an i.i.d. subset of data.

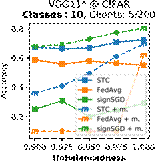


Fig. 9. Validation accuracy achieved by VGG11\* on CIFAR after 20000 iterations of communication-efficient federated training with different compression methods. The training data are split among the client at different degrees of unbalancedness with varying between 0.9 and 1.0.*γ*

lines refer to experiments where the momentum of m = 09 was used during training, while solid lines signify that classical SGD was used. As we can see, momentum has a significant influence on the convergence behavior of the different methods. While signSGD always performs distinctively better if momentum is turned on during the optimization, the picture is less clear for STC and federated averaging. We can make out three different parameters of the learning environment that determine whether momentum is beneficial or harmful to the performance of STC. If the participation rate is high and the batch size used during training is sufficiently large (see Fig. 7 left), momentum improves the performance of STC. Conversely, momentum will deteriorate the training performance in situations where training is carried out on small batches and with low client participation. The latter effect is increasingly strong if clients hold non-i.i.d. subsets of data [see Fig. 6 (right)]. These results are not surprising, as the issues with stale momentum described in [25] are enhanced in these situations. Similar relationships can be observed for federated averaging where again the size (see Fig. 7) and the heterogeneity (see Fig. 6) of the local minibatches determine whether the momentum will have a positive effect on the training performance or not.*.*

When we compare federated averaging, signSGD and STC in the following, we will ignore whichever version of these methods (momentum “on” or “off”) performs worse.

### B. Non-i.i.d.-ness of the Data

Our preliminary experiments in Section V have already demonstrated that the convergence behavior of both federated averaging and signSGD is very sensitive to the degree of i.i.d.ness of the local client data, whereas sparse communication seems to be more robust. We will now investigate this behavior in some more detail. Fig. 6 shows the maximum achieved generalization accuracy after a fixed number of iterations for VGG11\* trained on CIFAR at different levels of non-i.i.d.ness. Additional results on all other benchmarks can be found in Fig. A2 in the Appendix in the Supplementary Material. Both at full (left plot) and partial (right plot) client participations, STC outperforms federated averaging across all levels of i.i.d.-ness. The most distinct difference can be observed in the non-i.i.d. regime, where the individual clients hold less than five different classes. Here, STC (without momentum) outperforms both federated averaging and signSGD by a wide margin. In the extreme case where every client only holds data from exactly one class, STC still achieves 79.5% and 53.2% accuracy at full and partial client participations, respectively, while both federated averaging and signSGD fail to converge at all.

### C. Robustness to Other Parameters of the Learning Environment

We will now proceed to investigate the effects of other parameters of the learning environment on the convergence behavior of the different compression methods. Figs. 7–9 show the maximum achieved accuracy after training VGG11\* on CIFAR for 20000 iterations in different federated learning environments. Additional results on the three other benchmarks can be found in Section D in the Appendix in the

Supplementary Material.

We observe that STC (without momentum) consistently dominates federated averaging on all benchmarks and learning environments.

*1) Local Batch Size:* The memory capacity of mobile and IoT devices is typically very limited. As the memory footprint of SGD is proportional to the batch size used during training, clients might be restricted to train on small minibatches only. Fig. 7 shows the influence of the local batch size on the performance of different communication-efficient federated learning techniques exemplary for VGG11\* trained on CIFAR. First of all, we notice that using momentum significantly slows down the convergence speed of both STC and federated averaging at batch sizes smaller than 20 independent of the distribution of data among the clients. As we can see, even if the training data is distributed among the clients in an i.i.d. manner (see Fig. 7 right) and all clients participate in every training iteration, federated averaging suffers considerably from small batch sizes. STC, on the other hand, demonstrates to be far more robust to this type of constraint. At an extreme batch size of one, the model trained with STC still achieves an accuracy of 63.8%, while the federated averaging model only reaches 39.2% after 20000 training iterations.

*2) Client Participation Fraction:* Fig. 8 shows the convergence speed of VGG11\* trained on CIFAR10 in a federated learning environment with different degrees of client participation. To isolate the effects of reduced participation, we keep the absolute number of participating clients and the local batch sizes at constant values of 5 and 40, respectively, throughout all experiments and vary only the total number of clients (and thus the relative participation ). As we can see, reducing the participation rate has negative effects on both federated averaging and STC. The causes for these negative effects, however, are different. In federated averaging, the participation rate is proportional to the effective amount of data that the training is conducted on in any individual communication round. If a nonrepresentative subset of clients is selected to participate in a particular communication round of federated averaging, this can steer the optimization process away from the minimum and might even cause catastrophic forgetting [38] of previously learned concepts. On the other hand, partial participation reduces the convergence speed of STC by causing the clients residuals to go out sync and increasing the gradient staleness [25]. The more rounds a client has to wait before it is selected to participate during training again, the more outdated its accumulated gradients become. We can observe this behavior for STC most strongly in the non-i.i.d. situation (see Fig. 8 left), where the accuracy steadily decreases with the participation rate. However, even in the extreme case where only 5 out of 400 clients participate in every round of training, STC still achieves higher accuracy than federated averaging and signSGD. If the clients hold i.i.d. data (see Fig. 8 right), STC suffers much less from a reduced participation rate than federated averaging. If only 5 out of 400 clients participate in every round, STC (without momentum) still manages to achieve an accuracy of 68.2% while federated averaging stagnates at 42.3% accuracy. signSGD is affected the least by reduced participation, which is unsurprising, as only the absolute number of participating clients would have a direct influence on its performance. Similar behavior can be observed on all other benchmarks, and the results can be found in Fig. A3 in the Appendix in the Supplementary Material. It is noteworthy that in federated learning, it is usually possible for the server to exercise some control over the rate of client participation. For instance, it is typically possible to increase the participation ratio at the cost of a long waiting time for all clients to finish.*η*

*3) Unbalancedness:* Up until now, all experiments were performed with a balanced split of data in which every client was assigned the same amount of data points. In practice, however, the data sets on different clients will typically vary heavily in size. To simulate different degrees of unbalancedness, we split the data among the clients in a way such that the ith out of n clients is assigned a fraction *α γ i*

*ϕi(α,γ)* = *n* + (1 − α)*nj*=1 *γ j* (23)

of the total data. The parameter controls the minimum amount of data on every client, while the parameter controls the concentration of data. We fix = 01 and vary between 0.9 and 1.0 in our experiments. To amplify the effects of unbalanced client data, we also set the client participation to a low value of only 5 out of 200 clients. Fig. 9 shows the final accuracy achieved after 20000 iterations for different values of . Interestingly, the unbalancedness of the data does not seem to have a significant effect on the performance of either of the compression methods. Even if the data are highly concentrated on a few clients (as is the case for = 09), all methods converge reliably, and for federated averaging, the accuracy even slightly goes down with increased balancedness. Apparently, the rare participation of large clients can balance out several communication rounds with much smaller clients. These results also carry over to all other benchmarks (see Fig. A5 in the Appendix in the Supplementary Material).*α γ α .γ γγ .*

### D. Communication Efficiency

Finally, we compare the different compression methods with respect to the number of iterations and communicated bits they require to achieve a certain target accuracy on a federated learning task. As we saw in Section V, both federated averaging and signSGD perform considerably worse if clients hold non-i.i.d. data or use small batch sizes. To still have a meaningful comparison, we, therefore, choose to evaluate this time on an i.i.d. environment where every client holds ten different classes and uses a moderate batch size of 20 during training. This setup favors federated averaging and signSGD to the maximum degree possible! All other parameters of the learning environment are set to the base configuration given in Table III. We train until the target accuracy is achieved or a maximum amount of iterations is exceeded and measure the amount of communicated bits both for upload and download. Fig. 10 shows the results for VGG11\* trained on CIFAR, CNN trained on keyword spotting (KWS), and the LSTM model trained on Fashion-MNIST. We can see that even if all clients hold i.i.d. data, STC still manages to achieve the desired target accuracy within the smallest communication budget out of all methods. STC also converges faster in terms of training iterations than the versions of federated averaging with comparable compression rate. Unsurprisingly, we see that both for federated averaging and STC, we face a tradeoff between the number of training iterations (“computation”) and the number of communicated bits (“communication”). On all investigated benchmarks, however, STC is Pareto-superior to federated averaging in the sense for any fixed iteration complexity, it achieves a lower (upload) communication complexity.

Table IV shows the amount of upstream and downstream communications required to achieve the target accuracy for the different methods in megabytes. On the CIFAR learning task, STC at a sparsity rate of p = 00025 only communicates 183.9 MB worth of data, which is a reduction in communication by a factor of ×1995 as compared to the baseline with requires 36696 MB and federated averaging (n = 100), which still requires 1606 MB. Federated averaging with a delay period of 1000 steps does not achieve the target accuracy within the given iteration budget.*..*

## VIII. LESSONS LEARNED

We will now summarize the findings of this article and give general suggestions on how to approach communicationconstrained federated learning problems (see our summarizing Fig. 11).

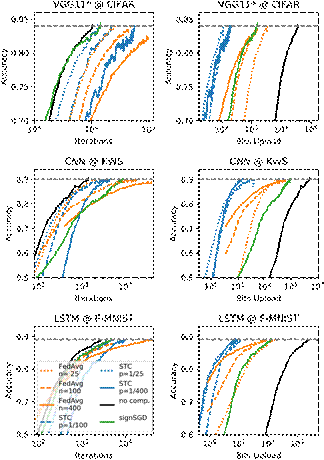


Fig. 10. Convergence speed of federated learning with compressed communication in terms of training iterations (left) and uploaded bits (right) on three different benchmarks (top to bottom) in an i.i.d. federated learning environment with 100 clients and 10% participation fraction. For better readability, the validation error curves are average-smoothed with a step size of five. On all benchmarks, STC requires the least amount of bits to converge to the target accuracy.

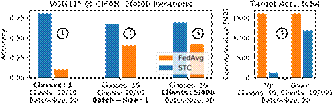


Fig. 11. Left: accuracy achieved by VGG11\* on CIFAR after 20000 iterations of federated training with federated averaging and STC for three different configurations of the learning environment. Right: upstream and downstream communication necessary to achieve a validation accuracy of 84% with federated averaging and STC on the CIFAR benchmark under i.i.d. data and a moderate batch-size.

1) If clients hold non-i.i.d. data, sparse communication protocols such as STC distinctively outperform federated averaging across all federated learning environments [see Figs. 6, 7 (left), and 8 (left)].

2) The same holds true if clients are forced to train on small minibatches (e.g., because the hardware is memory constrained). In these situations, STC outperforms federated averaging even if the client&apos;s data are i.i.d. [see Fig. 7 (right)].

## TABLE IV

BITS REQUIRED FOR TO ACHIEVE A CERTAIN*Upload and/ Download*

TARGET ACCURACY ON DIFFERENT LEARNING TASKS IN AN I.I.D.

LEARNING ENVIRONMENT. A VALUE OF “n.a.” IN THE TABLE SIGNIFIES THAT THE METHOD HAS NOT ACHIEVED THE TARGET ACCURACY WITHIN THE ITERATION BUDGET.

THE LEARNING ENVIRONMENT IS CONFIGURED AS DESCRIBED IN TABLE III



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



3) STC should also be preferred over federated averaging if the client participation rate is expected to be low, as it converges more stable and quickly in both the i.i.d. and non-i.i.d. regime [see Fig. 8 (right)].

4) STC is generally most advantageous in situations where the communication is bandwidth-constrained or costly (metered network, limited battery), as it does achieve a certain target accuracy within the minimum amount of communicated bits even on i.i.d. data (see Fig. 10 and Table IV).

5) Federated averaging in return should be used if the communication is latency-constrained or if the client participation is expected to be very low (and 1–3 do not hold).

6) Momentum optimization should be avoided in federated learning whenever either clients are training with small batch sizes or the client data are non-i.i.d. and the participation rate is low (see Figs. 6–8).

## IX. CONCLUSION

Federated learning for mobile and IoT applications is a challenging task, as generally little to no control can be exerted over the properties of the learning environment.

In this article, we demonstrated that the convergence behavior of current methods for communication-efficient federated learning is very sensitive to these properties. On a variety of different data sets and model architectures, we observe that the convergence speed of federated averaging drastically decreases in learning environments where the clients either hold non-i.i.d. subsets of data are forced to train on small minibatches or where only a small fraction of clients participates in every communication round. To address these issues, we propose STC, a communication protocol that compresses both the upstream and downstream communications via sparsification, ternarization, error accumulation, and optimal Golomb encoding. Our experiments show that STC is far more robust to the above-mentioned peculiarities of the learning environment than federated averaging. Moreover, STC converges faster than federated averaging both with respect to the number of training iterations and the amount of communicated bits even if the clients hold i.i.d. data and use moderate batch sizes during training.

Our approach can be understood as an alternative paradigm for communication-efficient federated optimization that relies on high-frequent low-volume instead of low-frequent highvolume communication. As such, it is particularly well suited for federated learning environments that are characterized by low latency and low bandwidth channels between clients and server.

## REFERENCES

[1] R. Taylor, D. Baron, and D. Schmidt, “The world in 2025: 8 Predictions for the next 10 years,” in Proc. 10th Int. Microsyst., Packag., Assembly Circuits Technol. Conf. (IMPACT), 2015, pp. 192–195.

[2] S. Wiedemann, K.-R. Müller, and W. Samek, “Compact and computationally efficient representation of deep neural networks,” IEEE Trans. Neural Netw. Learn. Syst., to be published. doi: 10.1109/TNNLS. 2019.2910073.

[3] S. Wiedemann, A. Marban, K.-R. Müller, and W. Samek, “Entropyconstrained training of deep neural networks,” in Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN), 2019, pp. 1–8.

[4] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, pp. 436–444, May 2015.

[5] A. Karpathy and L. Fei-Fei, “Deep visual-semantic alignments for generating image descriptions,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2015, pp. 3128–3137.

[6] S. Bosse, D. Maniry, K.-R. Müller, T. Wiegand, and W. Samek,

“Deep neural networks for no-reference and full-reference image quality assessment,” IEEE Trans. Image Process., vol. 27, no. 1, pp. 206–219, Jan. 2018.

[7] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, “Large-scale video classification with convolutional neural networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2014, pp. 1725–1732.

[8] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in Proc. Adv. Neural Inf. Process. Syst., 2014, pp. 3104–3112.

[9] W. Samek, T. Wiegand, and K.-R. Müller, “Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models,” ITU J., ICT Discoveries, vol. 1, no. 1, pp. 39–48, 2018.

[10] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” 2016, arXiv:1602.05629. [Online]. Available: https://arxiv.org/abs/1602.05629

[11] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, “How to backdoor federated learning,” 2018, arXiv:1807.00459. [Online]. Available: https://arxiv.org/abs/1807.00459

[12] K. Bonawitz et al., “Practical secure aggregation for privacy-preserving machine learning,” in Proc. ACM SIGSAC Conf. Comput. Commun. Secur., 2017, pp. 1175–1191.

[13] S. Hardy et al., “Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption,” 2017, arXiv:1711.10677. [Online]. Available: https://arxiv.org/abs/1711.10677

[14] M. Abadi et al., “Deep learning with differential privacy,” in Proc. ACM SIGSAC Conf. Comput. Commun. Secur., 2016, pp. 308–318.

[15] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770–778.

[16] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, “Densely connected convolutional networks,” in Proc. IEEE CVPR, vol. 1, Jun. 2017, no. 2, p. 3.

[17] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, “Sparse binary compression: Towards distributed deep learning with minimal communication,” in Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN), 2019, pp. 1–8.

[18] K. Bonawitz et al., “Towards federated learning at scale: System design,” 2019, arXiv:1902.01046. [Online]. Available: https://arxiv. org/abs/1902.01046

[19] W. Wen et al., “TernGrad: Ternary gradients to reduce communication in distributed deep learning,” 2017, arXiv:1705.07878. [Online]. Available:

https://arxiv.org/abs/1705.07878

[20] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, “QSGD: Communication-efficient SGD via gradient quantization and encoding,” in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 1707–1718.

[21] H. Wang, S. Sievert, Z. Charles, D. Papailiopoulos, S. Liu, and S. Wright, “ATOMO: Communication-efficient learning via atomic sparsification,” 2018, arXiv:1806.04090. [Online]. Available: https://arxiv. org/abs/1806.04090

[22] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, “signSGD: Compressed optimisation for non-convex problems,” 2018, arXiv:1802.04434. [Online]. Available: https://arxiv.org/abs/1802.04434

[23] A. F. Aji and K. Heafield, “Sparse communication for distributed gradient descent,” 2017, arXiv:1704.05021. [Online]. Available: https://arxiv. org/abs/1704.05021

[24] N. Strom, “Scalable distributed DNN training using commodity GPU cloud computing,” in Proc. 16th Annu. Conf. Int. Speech Commun. Assoc., 2015, pp. 1488–1492.

[25] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally, “Deep gradient compression: Reducing the communication bandwidth for distributed training,” 2017, arXiv:1712.01887. [Online]. Available: https://arxiv.org/abs/1712.01887

[26] Y. Tsuzuku, H. Imachi, and T. Akiba, “Variance-based gradient compression for efficient distributed deep learning,” 2018, arXiv:1802.06058. [Online]. Available: https://arxiv.org/abs/1802.06058

[27] J. Konecˇný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, “Federated learning: Strategies for improving communication efficiency,” 2016, arXiv:1610.05492. [Online]. Available: https://arxiv.org/abs/1610.05492

[28] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 2014, arXiv:1409.1556. [Online]. Available: https://arxiv.org/abs/1409.1556

[29] J. Bernstein, J. Zhao, K. Azizzadenesheli, and A. Anandkumar, “signSGD with majority vote is communication efficient and byzantine fault tolerant,” 2018, arXiv:1810.05291. [Online]. Available: https://arxiv.org/abs/1810.05291

[30] A. Krizhevsky, V. Nair, and G. Hinton. (2014). The CIFAR-10 Dataset. [Online]. Available: http://www.cs.toronto.edu/kriz/cifar.html

[31] Y. LeCun. (1998). The MNIST Database of Handwritten Digits. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[32] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, “Federated learning with non-IID data,” 2018, arXiv:1806.00582. [Online]. Available: https://arxiv.org/abs/1806.00582

[33] S. U. Stich, J.-B. Cordonnier, and M. Jaggi, “Sparsified SGD with memory,” in Proc. Adv. Neural Inf. Process. Syst., 2018, pp. 4447–4458.

[34] S. Golomb, “Run-length encodings (corresp.),” IEEE Trans. Inf. Theory, vol. 12, no. 3, pp. 399–401, Jul. 1966.

[35] S. Ioffe, “Batch renormalization: Towards reducing minibatch dependence in batch-normalized models,” in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 1945–1953.

[36] P. Warden, “Speech commands: A dataset for limited-vocabulary speech recognition,” 2018, arXiv:1804.03209. [Online]. Available: https://arxiv.org/abs/1804.03209

[37] H. Xiao, K. Rasul, and R. Vollgraf, “Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms,” 2017, arXiv:1708.07747. [Online]. Available: https://arxiv.org/abs/1708.07747

[38] I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville, and Y. Bengio, “An empirical investigation of catastrophic forgetting in gradient-based neural networks,” 2013, arXiv:1312.6211. [Online]. Available: https://arxiv.org/abs/1312.6211

**Felix Sattler** received the B.Sc. degree in mathematics, the M.Sc. degree in computer science, and the M.Sc. degree in applied mathematics from the Technische Universität Berlin, Berlin, Germany, in 2016, 2018, and 2018, respectively. He is currently with the Machine Learning Group, Fraunhofer Heinrich Hertz Institute, Berlin. His current research interests include distributed machine learning, neural networks, and multitask learning.

**Simon Wiedemann** received the M.Sc. degree in applied mathematics from the Technische Universität Berlin, Berlin, Germany, in 2017.

He is currently with the Machine Learning Group, Fraunhofer Heinrich Hertz Institute, Berlin. His current research interests include machine learning, neural networks, and information theory.

**Klaus-Robert Müller** (M&apos;12) received the Ph.D. degree in computer science from the Technische Universität Karlsruhe, Karlsruhe, Germany, in 1992, where he studied physics from 1984 to 1989.

He has been a Professor of computer science with the Technische Universität Berlin, Berlin, Germany, since 2006, where he is currently co-directing the Berlin Big Data Center. After completing a postdoctoral position at GMD FIRST, Berlin, he was a Research Fellow with The University of Tokyo,

Tokyo, Japan, from 1994 to 1995. In 1995, he

founded the Intelligent Data Analysis Group, GMD-FIRST (later Fraunhofer FIRST), and directed it until 2008. From 1999 to 2006, he was a Professor with the University of Potsdam, Potsdam, Germany. His current research interests include intelligent data analysis, machine learning, signal processing, and brain–computer interfaces.

Dr. Müller was elected to be a member of the German National Academy of Sciences-Leopoldina in 2012, the Berlin Brandenburg Academy of sciences in 2017, and an External Scientific Member of the Max Planck Society in 2017. He received the 1999 Olympus Prize by the German Pattern Recognition Society, DAGM. He received the SEL Alcatel Communication Award in 2006, the Science Prize of Berlin awarded by the Governing Mayor of Berlin in 2014, and the Vodafone Innovation Award in 2017.

**Wojciech Samek** (M&apos;13) received the Diploma degree in computer science from the Humboldt University of Berlin, Berlin, Germany, in 2010, and the Ph.D. degree in machine learning from the Technische Universität Berlin, Berlin, in 2014.

In 2014, he founded the Machine Learning Group, Fraunhofer Heinrich Hertz Institute, where he is currently the Director. He was a Scholar of the German National Academic Foundation and a Ph.D.

Fellow with the Bernstein Center for Computational

Neuroscience Berlin, Berlin, where he is also with

the Berlin Big Data Center. He was visiting with Heriot-Watt University, Edinburgh, U.K., and The University of Edinburgh, Edinburgh, from 2007 to 2008. In 2009, he was with the Intelligent Robotics Group, NASA Ames Research Center, Mountain View, CA, USA. His current research interests include interpretable machine learning, neural networks, federated learning, and computer vision.