Communication-Efficient Learning of Deep Networks from Decentralized Data

1. Abstract

- Federated Learning : 训练数据分布在移动设备上，通过**聚合**本地计算的**更新**来学习共享模型
- 考虑5种模型架构和4个数据集，提出的Iterative model averaging**迭代模型平均**
- 证明了对于non-balanced, non-IID 的数据具有**鲁棒性**
- **通信成本**是主要限制因素，比同步的SGD descent相比，通信次数减少10-100倍。

2. Introduction

- Federated Learning 的 Learning tasks是被选中的参与设备，即clients完成，由central server来提供协调

  每个client都有一个local dataset，这个dataset是不会上传到server的

  每个client对global model计算一个update，对这个update进行communication。

  这些updates是针对特定model，当被应用后，就没有必要存储他们

- 基本过程：server把global model发送到clients，clients利用本地数据集训练，将训练后的权重上传到server，实现global model 的更新

优点：模型训练与直接访问原始数据的解耦。也可将供给面限制在设备，而不是设备和云，来降低风险

2.1 Federated Averaging

提出了**Federated Averaging**的算法，结合了本地的SGD，在Client和Server之间执行model averaging

Federated Learning的属性：

1）与数据中心提供的针对代理数据的培训相比，来自mobile devices的real-world的数据训练更有优势（**数据真实**）

2）数据都是privacy sensitive或者large in size，因此不要将其记录在data center

3）对于监督学习任务，数据的labels可以从与users的交互中推断出

2.2 Privacy

data center情况下即使拥有一个"匿名"的数据集，通过连接其他用户的数据危及隐私

而federated learning传输数据是一些model的最小的updates，隐私的强度也取决于更新的内容

aggregation algorithm可以在不识别原数据来源的情况下完成，因此updates可以在直接传输

2.3 Federated Optimization

（区别于典型的分布式系统的优化）

- **Non-IID**

clients的数据都是基于mobile devices的使用情况，特定用户的local dataset不会代表什么分布（**非独立同分布**）

- **Unbalanced**

一些users对于service或者app的使用会更多，导致local training data的数量是变化较大（**数据量不同**）

- **Massively distributed**

参与optimization的用户数量远大于每个用户的示例数量（**用户分布大规模**）

- **Limited communication**

mobile devices通常是offline（**通信资源有限**）

Federated optimization需要考虑实际的问题：client的dataset随着数据的增添删除会变化

主要针对*non-iid*和不平衡的问题进行研究优化

2.3 Experiment Background

固定K个clients,每个都有一个固定的local dataset

每一轮开始时，随机选中比例C的clients(超过某个点的clients数量会导致效果变差)，server发送当前的global algorithm state给每个clients(当前global model的parameters)

每个选中的clients基于global state和local dataset进行本地计算，发送update到server

server把这些updates应用到global state，并重复上述过程

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

$f_i(w) = loss(x_i, y_i; w)$是在样例（$x_i, y_i$）上的基于$global \quad parameters \quad w$的预测的$loss$值
假设现在有K个$client$，第k个$client$的数据点是P_k, 对应数据量是$n_k = |P_k|$，（加权平均）公式：

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

随机均匀采样时候，就是IID:

$$\mathbb{E}_{\mathcal{P}_\mathrm{k}}\left[\mathrm{F_k(w)}\right] = \mathrm{f(w)}$$

2.4 Datacenter vs FedAvg

目标：使用额外的计算来减小通信的轮数

1）*increased parallelism* 增加更多的clients独立工作

2）*increased computation on each client*（更复杂的计算）

2.5 Related work

1. 通过分布式的*iteratively averaging locally training*已经有进展
2. 联邦学习通常不考虑*datasets*的*unbalanced*和*non-IID*
3. 也可关注训练的深度网络，强调privacy的重要性，在每一轮通信中共享参数的子集
4. 每个*model*在本地找到*minimize loss*的*parameters*，然后发送到*server*去*average*

3. The FederatedAveraging Algorithm

在深度学习中，最常用的用于优化的方法就是SGD（随机梯度下降），基于梯度来找到最优。

SGD应用于**联邦优化**问题，

3.1 Baseline-FedSGD

在每一轮中随机选择一组客户端，并计算损失梯度，***(只计算一次)***

每轮通信中进行单个批处理的梯度计算， 每一轮选择C分之一的clients，计算这些clients拥有数据的gradient的 loss值

缺点：但需要大量的训练才能得到好的model（通信轮数多）

3.2 Parameters

三个重要的参数：

> C：每轮参与的clients 的小数占比。控制了global batch 的size，C = 1 对应的是full-batch
>
> E：每个client在每一轮重的训练的次数,**本地的epochs**，在FedSGD中，E=1；

> B：对于clients的updates而言的 local batch size，在FedSGD中，B=∞
>
> （B是无穷时候，表示full local dataset 作为了一个single minibatch）

对于一个*E= 1 ,B*= ∞就是其中的一个极端，*FedAvg*等价于*FedSGD*。

3.2 FederatedAveraging （FedAvg）

> 每个client在本地使用当前的model使用local data做gradient descent
>
> 然后server对于result model 做一次加权的average

因此，可以对每个client通过本地**多次迭代update**来更多的增大计算量。

对于一个有n_k个local examples的client，每轮local updates的数量是u_k = E * n_k/ B

算法细节：

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**

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

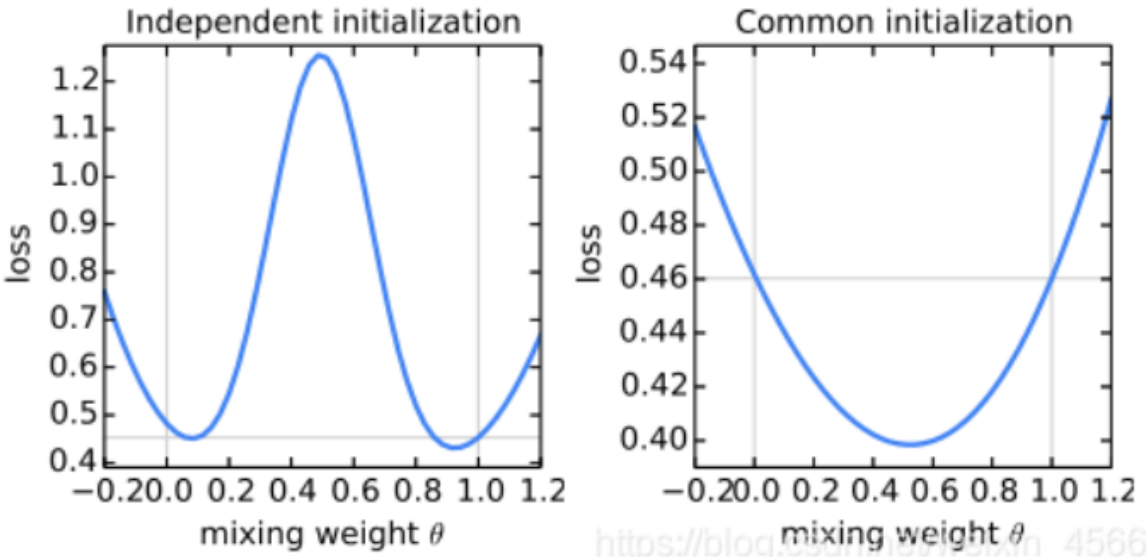
return w to server

如下图是两种**初始化参数对model avg的影响**：

1. 不同的模型使用相互独立的初始化模型
2. 不同的模型使用相同的初始化模型

得出结论：相同的初始化参数对模型平均，可以显著降低loss

（证明了联邦学习时，需要server发布global model，每个client使用相同参数的model训练效果更好）



4. Experimental Results

4.1 MNIST的digit recognition task

- **MINIST数据集——CNN和2NN**

1）一个多层感知机，有2层的hidden layers，使用ReLu激活函数，用200个units，也就是MNIST 2NN

2）一个CNN有两个5*5的卷积层，第一个是32channels，第二个是64个，每个后面是2*2的max pooling，一个有512个units的全连接层，和ReLu的激活函数，以及最后的softmax的output layer

考虑了两种MNIST的数据的分布：
1）IID,数据是被shuffled，之后划分到100clients上，每个接收600examples

2）Non-IID,根据digit的label去sort，然后划分为200个碎片，每个shards的size是300，分配给每100个clients 2个shards

4.2 Language的modeling

- **莎士比亚数据集——LSTM**

a stacked character-level LSTM language model

在读入一行的character后，预测下一个character

以一系列的字符作为input，将每个character嵌入到一个学习到的8维的空间中

嵌入的字符之后会被一个2层的LSTM model处理，每层有256个nodes

之后第二层的LSTM的输出进入到一个softmax的output layer，每个character只有一个node

4.3 Increasing parallelism

增加并行其实就是clients数量变多

C控制了多个clients的并行度，调整C来找到合适的并行

这里是**固定了参数E，对C和B进行调整**，分别在iid和non-iid下得到结果：

2NN	IID		NON-IID	
C	$B = \infty$	$B = 10$	$B = \infty$	$B = 10$
0.0	1455	316	4278	3275
0.1	1474 (1.0×)	87 (3.6×)	1796 (2.4×)	664 (4.9×)
0.2	1658 (0.9×)	77 (4.1×)	1528 (2.8×)	619 (5.3×)
0.5	— (—)	75 (4.2×)	— (—)	443 (7.4×)
1.0	— (—)	70 (4.5×)	— (—)	380 (8.6×)
CNN, $E = 5$				
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×)

C变大，clients数量变多，迭代的轮数就减少，收敛越快

> （但是**不能一直增加用户数量**，当clients数量达到一定数量，收敛速度就不再明显增加，这时候增加用户数量对于收敛速度也没有效果）

从图中可以看出：

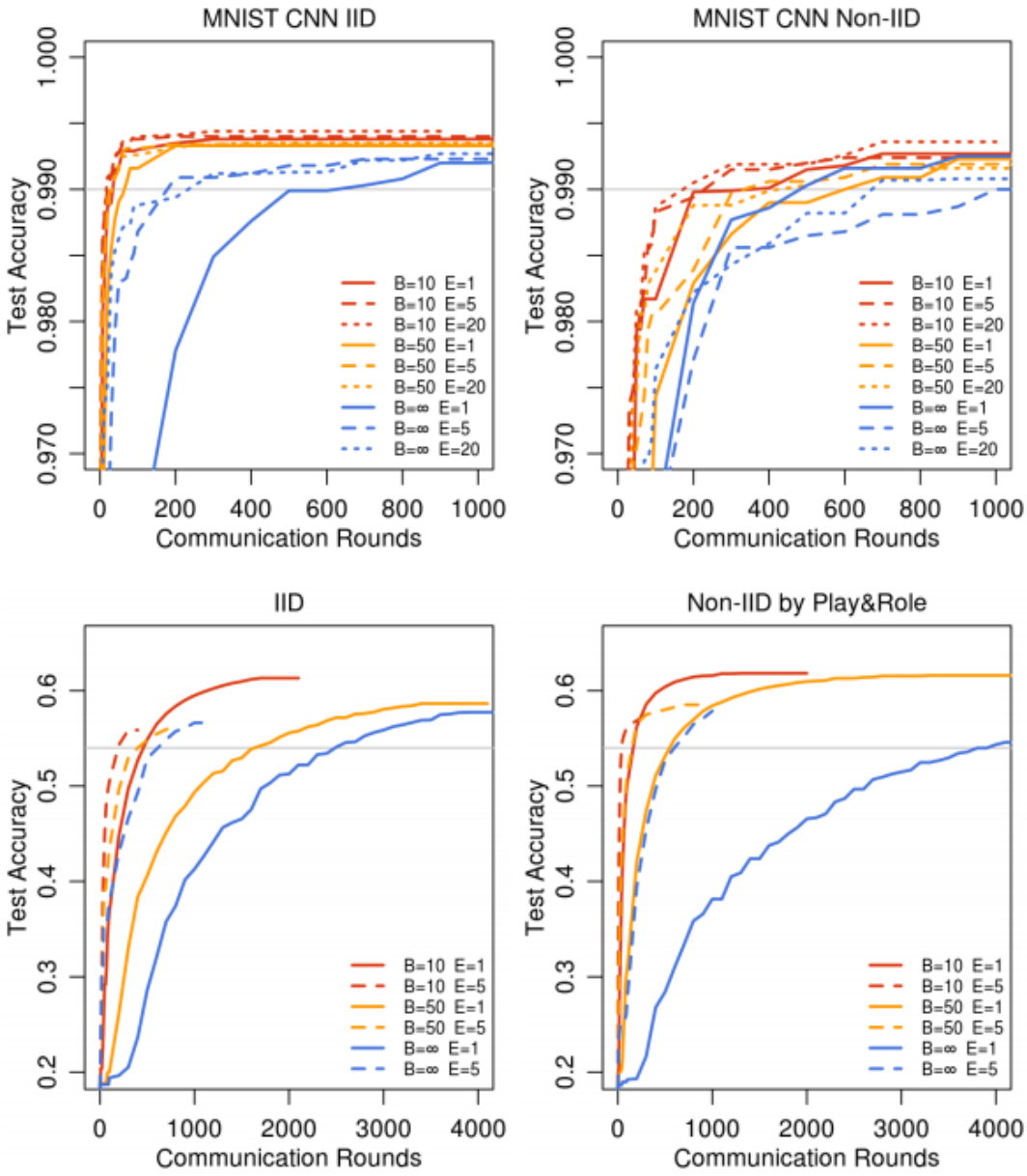
1. B=∞，C变大时候的优势并不明显；
2. B=10,C= 0.1的时候效果是最好的，计算效率和收敛速度达到平衡

4.4 Increasing computation per client

- 固定C = 0.1（实验得到这个点的收敛速度比较好），在每一轮给每个client加入更多的computation
- 增加更多的local updates可以很大程度降低通信开销

每个client的预期的updates的数量是**u_k = E * n_k / B** （要么减小B,增大E)

u: 每个客户端每回合的预期更新次数


MNIST CNN IID
Test Accuracy
1.000
0.990
0.980
0.970
0
200
400
600
800
1000
Communication Rounds
B=10 E=1
B=10 E=5
B=10 E=20
B=50 E=1
B=50 E=5
B=50 E=20
B=∞ E=1
B=∞ E=5
B=∞ E=20
MNIST CNN Non-IID
Test Accuracy
1.000
0.990
0.980
0.970
0
200
400
600
800
1000
Communication Rounds
B=10 E=1
B=10 E=5
B=10 E=20
B=50 E=1
B=50 E=5
B=50 E=20
B=∞ E=1
B=∞ E=5
B=∞ E=20
IID
Test Accuracy
0.6
0.5
0.4
0.3
0.2
0
1000
2000
3000
4000
Communication Rounds
B=10 E=1
B=10 E=5
B=50 E=1
B=50 E=5
B=∞ E=1
B=∞ E=5
Non-IID by Play&Role
Test Accuracy
0.6
0.5
0.4
0.3
0.2
0
1000
2000
3000
4000
Communication Rounds
B=10 E=1
B=10 E=5
B=50 E=1
B=50 E=5
B=∞ E=1
B=∞ E=5

MNIST CNN, 99% ACCURACY					
CNN	E	B	u	IID	NON-IID
FEDSGD	1	∞	1	626	483
FEDAVG	5	∞	5	179 (3.5×)	1000 (0.5×)
FEDAVG	1	50	12	65 (9.6×)	600 (0.8×)
FEDAVG	20	∞	20	234 (2.7×)	672 (0.7×)
FEDAVG	1	10	60	34 (18.4×)	350 (1.4×)
FEDAVG	5	50	60	29 (21.6×)	334 (1.4×)
FEDAVG	20	50	240	32 (19.6×)	426 (1.1×)
FEDAVG	5	10	300	20 (31.3×)	229 (2.1×)
FEDAVG	20	10	1200	18 (34.8×)	173 (2.8×)
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×)

> CNN: 在IID下效果明显，提升达到34.8倍，而Non-IID下提升不明显，只有2.8倍
>
> LSTM: 在Non-IID下效果更加明显，95.3倍，在IID下是13.0倍
>
> （按角色分，某些角色数据集较大，对于增加本地训练有价值）

通过改变E和B来增大u是有效的，只要B足够大，就可以充分利用client硬件的并行性

4.5 Other comparisons

进一步验证FedAvg的效果

4.5.1 FedSGD vs FedAvg Learning rate (CIFAR-10)

固定参数，**学习率变化**

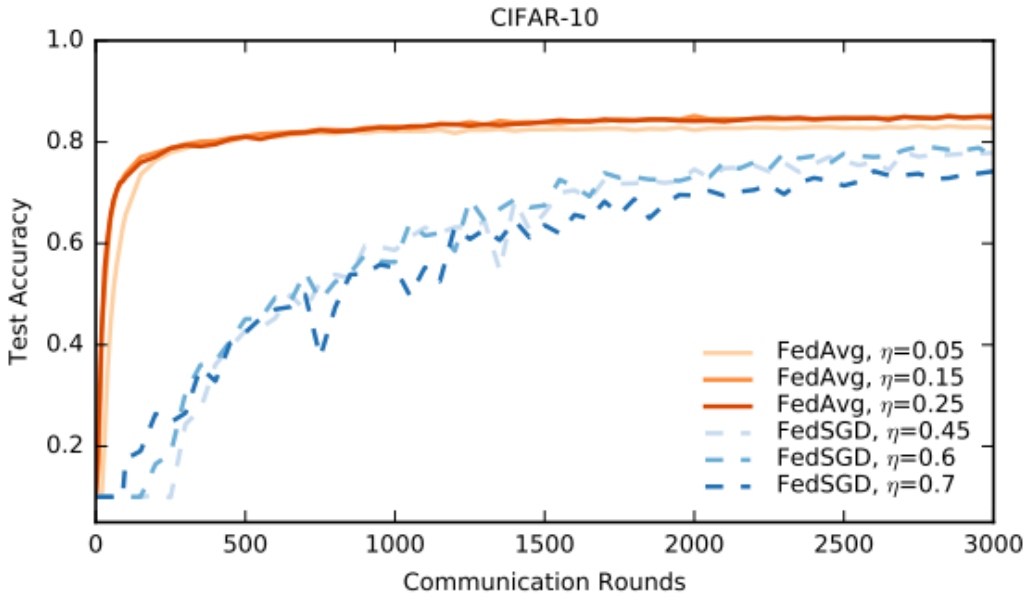


Figure 4: Test accuracy versus communication for the CIFAR10 experiments. FedSGD uses a learning-rate decay of 0.9934 per round; FedAvg uses $B = 50$, learning-rate decay of 0.99 per round, and $E = 5$.

4.5.2 SGD、FedSGD、FedAvg Rounds（CIFAR-10）

模型在达到某一个Accuracy时候需要迭代的**轮数**

Table 3: Number of rounds and speedup relative to baseline SGD to reach a target test-set accuracy on CIFAR10. SGD used a minibatch size of 100. FedSGD and FedAvg used $C = 0.1$, with FedAvg using $E = 5$ and $B = 50$.

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×)

4.5.3 FedSGD vs FedAvg (large-level word LSTM)

对于大规模的语言模型，比较**不同学习率**对Accuracy的影响

训练数据来自大型社交网络的1000万个帖子。 将帖子按作者分组，共有超过500,000个客户。

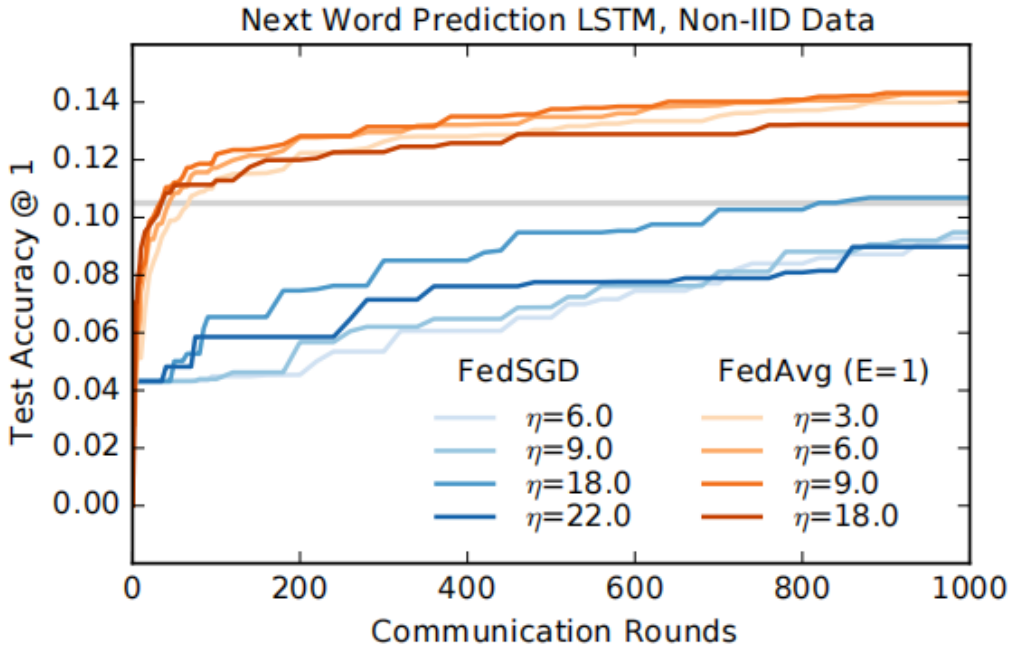


Figure 5: Monotonic learning curves for the large-scale language model word LSTM.

5. Conclusions and Future Work

- Federated Learning 对于在较少通信轮数下训练高质量的模型是可行的
- 在Federated Learning上通过DP、MPC or their combination可以提供更强的隐私保证