IEEE ICASSP 2020 Tutorial on Distributed and Efficient Deep Learning

Outline of tutorial

1. Introduction (WS)
2. Model Compression & Efficient Deep Learning (WS)
3. (Virtual) Coffee Break
4. Distributed & Federated Learning (FS)
Part III: Distributed & Federated Learning

Wojciech Samek & Felix Sattler
Distributed Learning

Server

Data

Data

Data

Data

Data

Data

Data
Traditional Centralized ("Cloud") ML

→ Data is gathered Centrally

Problems

- Privacy
- Ownership (→ who owns the data?)
- Security (→ single point of failure)
- Efficiency (→ need to move data around)

Server

train model
Distributed (/ “Embedded”) ML

→ Data never leaves the local Devices

→ Instead model Updates are communicated

Problems

- Privacy
- Ownership (→ who owns the data?)
- Security (→ single point of failure)
- Efficiency (→ need to move data around)

Solved
Distributed ("Embedded") ML: Settings

Federated Learning

Peer-to-Peer Learning

Distributed Training

On-Device Inference

Detailed Comparison: Sattler, Wiegand, Samek. "Trends and Advancements in Deep Neural Network Communication."
Federated Learning

“Federated Learning is a machine learning setting where multiple entities collaborate in solving a learning problem, without directly exchanging data. The Federated training process is coordinated by a central server.”

Kairouz, Peter, et al. "Advances and open problems in federated learning."
Federated Learning

Nvidia uses federated learning to create medical imaging AI

Federated learning technique predicts hospital stay and patient mortality

PUBLICATIONS:
Federated Learning for Mobile Keyboard Prediction

How Apple personalizes Siri without hoovering up your data

Fraunhofer Heinrich Hertz Institute
IEEE ICASSP 2020 Tutorial on Distributed and Efficient Deep Learning
Federated Learning

Server

Data

Data

Data

Fraunhofer Heinrich Hertz Institute

IEEE ICASSP 2020 Tutorial on Distributed and Efficient Deep Learning
Federated Learning

Server

SGD

Data

SGD

Data

SGD

Data
Federated Learning

Server

averaging
Federated Learning

IEEE ICASSP 2020 Tutorial on Distributed and Efficient Deep Learning
Federated Learning - Settings

**Cross Device**
- Large Number of Clients
- Only fraction of Clients available at any given time
- Few data points per Client
- Limited computational resources

**Cross Silo**
- Small number of Clients
- Clients are always available
- Large local data sets
- Strong computational resources
Federated Learning - Challenges

Challenges in Federated Learning

- Privacy
- Robustness
- Personalization
- Heterogeneity
- Convergence
- Communication
Federated Learning - Communication

Download

Expensive communication!

Upload

Expensive communication!
Federated Learning - Communication

Total Communication = [#Communication Rounds] x [#Parameters] x [Avg. Codeword length]

Case Study: VGG16 on ImageNet

- Number of Iterations until Convergence: 900,000
- Number of Parameters: 138,000,000
- Bits per Parameter: 32

→ Total Communication = **496.8 Terabyte** (Upload+Download)
Federated Learning - Compression Methods

Total Communication = [#Communication Rounds] x [#Parameters] x [Avg. Codeword length]

Compression Methods

- Communication Delay
- Lossy Compression: Unbiased
- Lossy Compression: Biased
- Efficient Encoding
Communication Delay

**Distributed SGD:**
For $t=1,..,[$Communication Rounds$]:

For $i=1,..,[$Participating Clients$]:

Client does:

$$g_i \leftarrow \nabla_{\theta} l(\theta_t, D_i^b)$$

Server does:

$$\theta_{t+1} = \theta_t - \eta \frac{1}{M} \sum_i g_i$$

**Federated Averaging:**
For $t=1,..,[$Communication Rounds$]:

For $i=1,..,[$Participating Clients$]:

Client does:

$$\theta_i = \text{SGD}_K(\theta_t, D_i)$$

Server does:

$$\theta_{t+1} = \frac{1}{M} \sum_i \Delta \theta_i$$
Communication Delay

**Advantages:**

- Simple
- Reduces Communication Frequency (advantageous in on-device FL)
- Reduces both Upstream and Downstream communication
- Easy to integrate with Privacy mechanisms

**Disadvantages:**
Communication Delay

Convergence Analysis for Convex Objectives:

\[ E[F(x_T) - F(x^*)] \in O\left(\frac{HM}{T} + \frac{\sigma}{\sqrt{TKM}}\right) \]

- \( M \) – Clients Participating per Round
- \( T \) – Total communication Rounds
- \( K \) – Local Iterations per Round
- \( L \) – Lipschitz Parameter of the Loss Function
- \( \sigma \) – Bound on the Variance of the Stochastic Gradients

IID-Assumption:
\[ D_i \sim \varphi(x, y) \]

Statistical Heterogeneity

Convergence speed drastically decreases with increasing heterogeneity in the data.

→ This effect aggravates if the number of participating clients (“reporting fraction”) is low.

Hsu, Tzu-Ming Harry, Hang Qi, and Matthew Brown. "Measuring the effects of non-identical data distribution for federated visual classification."
Communication Delay

<table>
<thead>
<tr>
<th>Method</th>
<th>Non-IID</th>
<th>Other assumptions</th>
<th>Variant</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lian et al. [266]</td>
<td>BCGV</td>
<td>BLGV</td>
<td>Dec; AC; 1step</td>
<td>$O(1/T) + O(1/\sqrt{NT})$</td>
</tr>
<tr>
<td>PD-SGD [265]</td>
<td>BCGV</td>
<td>BLGV</td>
<td>Dec; AC</td>
<td>$O(N/T) + O(1/\sqrt{NT})$</td>
</tr>
<tr>
<td>MATCHA [401]</td>
<td>BCGV</td>
<td>BLGV</td>
<td>Dec</td>
<td>$O(1/\sqrt{TKM}) + O(M/KT)$</td>
</tr>
<tr>
<td>Khaled et al. [232]</td>
<td>BOGV</td>
<td>CVX</td>
<td>AC; LBG</td>
<td>$O(N/T) + O(1/\sqrt{NT})$</td>
</tr>
<tr>
<td>Li et al. [264]</td>
<td>BOBD</td>
<td>SCVX; BLGV; BLGN</td>
<td>-</td>
<td>$O(1/\sqrt{T})$</td>
</tr>
<tr>
<td>FedProx [261]</td>
<td>BGV</td>
<td>BNCVX</td>
<td>Prox</td>
<td>$O(1/\sqrt{T})$</td>
</tr>
<tr>
<td>SCAFFOLD [227]</td>
<td>-</td>
<td>SCVX; BLGV</td>
<td>VR</td>
<td>$O(1/TKM) + O(e^{-T})$</td>
</tr>
</tbody>
</table>

Kairouz, Peter, et al. "Advances and open problems in federated learning."
Communication Delay

Advantages:

- Simple
- Reduces Communication Frequency (more practical in on-device FL)
- Reduces Upstream + Downstream communication
- Easy to integrate with Privacy mechanisms

Disadvantages:

- Bad performance on non-iid data
- Low sample efficiency
Federated Learning - Compression Methods

Total Communication = \([#\text{Communication Rounds}] \times [#\text{Parameters}] \times [\text{Avg. Codeword length}]\)

Compression Methods

- Communication Delay
- Lossy Compression: Unbiased
- Lossy Compression: Biased
- Efficient Encoding
Update Compression

Distributed SGD:
For $t=1,\ldots,[\text{Communication Rounds}]$:
For $i=1,\ldots,[\text{Participating Clients}]$:

Client does:

\[ g_i \leftarrow \nabla_{\theta} l(\theta_t, D_i^b) \]

Server does:

\[ \theta_{t+1} = \theta_t - \eta \frac{1}{M} \sum_i g_i \]

Distributed SGD with Compression:
For $t=1,\ldots,[\text{Communication Rounds}]$:
For $i=1,\ldots,[\text{Participating Clients}]$:

Client does:

\[ g_i \leftarrow \nabla_{\theta} l(\theta_t, D_i^b) \]
\[ \tilde{g}_i \leftarrow \text{comp}(g_i) \]

Server does:

\[ \theta_{t+1} = \theta_t - \eta \frac{1}{M} \sum_i \tilde{g}_i \]
Update Compression

--- Unbiased ---

---------- Biased ----------
Unbiased Compression

**Definition:** A compression operator \( \text{comp} : \mathbb{R}^d \rightarrow \mathbb{R}^d \) is called **unbiased** iff,

\[
E[\text{comp}(x)] = x \quad \forall x \in \mathbb{R}^d
\]

**Pros:**

- “Straight forward” Convergence Analysis (Stochastic Gradients with increased variance)
- Variance reduction (uncorrelated noise)

**Strongly convex bound:**

\[
E[f(x_T) - f^*] \leq O\left(\frac{\sigma^2}{\mu T}\right)
\]

Variance of the gradient estimator

\[
\sigma^2\left[\frac{1}{n} \sum_{i=1}^{n} \text{comp}(x_i)\right] = \frac{1}{n^2} \sum_{i=1}^{n} \sigma^2[\text{comp}(x_i)]
\]
Unbiased Compression

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Test accuracy</th>
<th>Data/epoch</th>
<th>Time per batch</th>
</tr>
</thead>
<tbody>
<tr>
<td>SGD</td>
<td>94.3%</td>
<td>1023 MB</td>
<td>312 ms</td>
</tr>
<tr>
<td>Atomo</td>
<td>92.6%</td>
<td>113 MB</td>
<td>948 ms</td>
</tr>
<tr>
<td>Signum</td>
<td>93.6%</td>
<td>32 MB</td>
<td>301 ms</td>
</tr>
<tr>
<td>Rank 2</td>
<td>94.4%</td>
<td>8 MB</td>
<td>239 ms</td>
</tr>
</tbody>
</table>

- **Pros**: “Straight forward” Convergence Analysis (Stochastic Gradients, increased variance)
- **Cons**: Variance blow-up leads to poor empirical performance
Biased Compression

**Definition:** A compression operator \( \text{comp} : \mathbb{R}^d \rightarrow \mathbb{R}^d \) is called **biased** iff,

\[
E[\text{comp}(x)] \neq x \quad \forall x \in \mathbb{R}^d
\]

Biased compression methods do not necessarily converge!

→ Can be turned into convergent methods via error accumulation.

Karimireddy, et al. "Error feedback fixes signsgd and other gradient compression schemes."
Stich, Cordonnier, Jaggi. "Sparsified SGD with memory."
Error Accumulation

Distributed SGD:
For $t=1,\ldots,[\text{Communication Rounds}]$:
For $i=1,\ldots,[\text{Participating Clients}]$:

Client does:
\[ g_i \leftarrow \nabla_{\theta} l(\theta_t, D_i^b) \]

Server does:
\[ \theta_{t+1} = \theta_t - \eta \frac{1}{M} \sum_i g_i \]

Distributed SGD with Error Accumulation:
For $t=1,\ldots,[\text{Communication Rounds}]$:
For $i=1,\ldots,[\text{Participating Clients}]$:

Client does:
\[ R_i \leftarrow R_i + \nabla_{\theta} l(\theta_t, D_i^b) \]
\[ \tilde{g}_i \leftarrow \text{comp}(R_i) \]
\[ R_i \leftarrow R_i - g_i \]

Server does:
\[ \theta_{t+1} = \theta_t - \eta \frac{1}{M} \sum_i \tilde{g}_i \]
Error Accumulation

For a parameter \( \alpha \in [0, 1) \), a \( \alpha \) - contraction operator is a (possibly randomized) operator \( \text{comp} : \mathbb{R}^d \rightarrow \mathbb{R}^d \) that satisfies the contraction property

\[
E\|x - \text{comp}(x)\|^2 \leq \alpha \|x\|^2, \quad \forall x \in \mathbb{R}^d
\]

Theorem (Stich et al.): For any contraction operator, compressed SGD with Error Accumulation for large \( T \) achieves convergence rate on \( \mu \)-strongly convex objective functions with asymptotic rate:

\[
E[f(x_T) - f(x^*)] \leq O\left(\frac{G^2}{\mu T}\right)
\]

Independent of alpha!
# Federated Learning - Recap Compression

<table>
<thead>
<tr>
<th>Methods</th>
<th>Unbiased</th>
<th>Biased</th>
</tr>
</thead>
<tbody>
<tr>
<td>TernGrad, QSGD, Atomo</td>
<td>Gradient Dropping, Deep Gradient Compression, signSGD, PowerSGD,</td>
<td></td>
</tr>
<tr>
<td>Convergence Proofs</td>
<td>Bounded Variance Assumption</td>
<td>k-contraction Framework (Stich et al. 2018)</td>
</tr>
</tbody>
</table>
Combining Methods: Sparse Binary Compression

Sattler, et al. "Sparse binary compression: Towards distributed deep learning with minimal communication." 2019 International Joint Conference on Neural Networks (IJCNN).
# Sparse Binary Compression

<table>
<thead>
<tr>
<th>Compression Method</th>
<th>Method</th>
<th>Baseline</th>
<th>DGC</th>
<th>Fed. Avg.</th>
<th>SBC (1)</th>
<th>SBC (2)</th>
<th>SBC (3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LeNet5-Caffe @MNIST</td>
<td>Accuracy</td>
<td>0.9946</td>
<td>0.994</td>
<td>0.994</td>
<td>0.994</td>
<td>0.994</td>
<td>0.994</td>
</tr>
<tr>
<td></td>
<td>Compression</td>
<td>×1</td>
<td>×718</td>
<td>×500</td>
<td>×2071</td>
<td>×3166</td>
<td>×24935</td>
</tr>
<tr>
<td>ResNet18 @CIFAR10</td>
<td>Accuracy</td>
<td>0.946</td>
<td>0.9383</td>
<td>0.9279</td>
<td>0.9422</td>
<td>0.9435</td>
<td>0.9219</td>
</tr>
<tr>
<td></td>
<td>Compression</td>
<td>×1</td>
<td>×768</td>
<td>×1000</td>
<td>×2369</td>
<td>×3491</td>
<td>×31664</td>
</tr>
<tr>
<td>ResNet34 @CIFAR100</td>
<td>Accuracy</td>
<td>0.773</td>
<td>0.767</td>
<td>0.7316</td>
<td>0.767</td>
<td>0.7655</td>
<td>0.701</td>
</tr>
<tr>
<td></td>
<td>Compression</td>
<td>×1</td>
<td>×718</td>
<td>×1000</td>
<td>×2370</td>
<td>×3166</td>
<td>×31664</td>
</tr>
<tr>
<td>ResNet50 @ImageNet</td>
<td>Accuracy</td>
<td>0.737</td>
<td>0.739</td>
<td>0.724</td>
<td>0.735</td>
<td>0.737</td>
<td>0.728</td>
</tr>
<tr>
<td></td>
<td>Compression</td>
<td>×1</td>
<td>×601</td>
<td>×1000</td>
<td>×2569</td>
<td>×3531</td>
<td>×37208</td>
</tr>
<tr>
<td>WordLSTM @PTB</td>
<td>Perplexity</td>
<td>76.02</td>
<td>75.98</td>
<td>76.37</td>
<td>77.73</td>
<td>78.19</td>
<td>77.57</td>
</tr>
<tr>
<td></td>
<td>Compression</td>
<td>×1</td>
<td>×719</td>
<td>×1000</td>
<td>×2371</td>
<td>×3165</td>
<td>×31658</td>
</tr>
<tr>
<td>WordLSTM* @WIKI</td>
<td>Perplexity</td>
<td>101.5</td>
<td>102.318</td>
<td>131.51</td>
<td>103.95</td>
<td>103.95</td>
<td>104.62</td>
</tr>
<tr>
<td></td>
<td>Compression</td>
<td>×1</td>
<td>×719</td>
<td>×1000</td>
<td>×2371</td>
<td>×3165</td>
<td>×31657</td>
</tr>
</tbody>
</table>
Sparse Binary Compression for non-iid Data

VGG11* @ CIFAR
IID Data

VGG11* @ CIFAR
NON-IID Data (2)

VGG11* @ CIFAR
NON-IID Data (1)
Federated Learning - Combining Methods

Sattler, Wiedemann, Müller, Samek. "Robust and communication-efficient federated learning from non-iid data." IEEE TNNLS (2019).
Efficient Encoding - DeepCABAC

- CABAC best encoder for quantized parameter tensors
- Plug & Play
- Can be used as a final lossless compression stage for all compression methods that we have presented

Wiedemann et al. "DeepCABAC: A Universal Compression Algorithm for Deep Neural Networks."
Federated Learning - Challenges

Challenges in Federated Learning

- Privacy
- Communication
- Convergence
- Robustness
- Personalization
- Heterogeneity

✔ Convergence
✔ Communication
( ) Heterogeneity
✔ Privacy
Federated Learning - Privacy

Hitaj, Briland, Giuseppe Ateniese, and Fernando Perez-Cruz. "Deep models under the GAN: information leakage from collaborative deep learning."
Federated Learning - Privacy

Privacy Protection Mechanisms:

- Secure Multi-Party Computation
- Homomorphic Encryption
- Trusted Execution Environments
- **Differential Privacy**

Dwork, Cynthia, and Aaron Roth. "The algorithmic foundations of differential privacy."
Differential Privacy

A mechanism $A: \mathcal{P}(\mathcal{X}) \to \mathcal{Y}$ is called differentially private with parameter $\varepsilon$ iff,

$$\sup_{S \subseteq \mathcal{Y}} \log \left( \frac{P[A(D') \in S]}{P[A(D) \in S]} \right) \leq \varepsilon$$

for any two data sets $D$ and $D'$ which differ in only one element.
Differential Privacy - Mechanisms

Global Sensitivity:

\[ S(f) = \max_{\text{dist}(D,D')=1} |f(D) - f(D')| \]

The Laplace mechanism \( A(D) = f(D) + Z \) with \( Z \sim \frac{S(f)}{\varepsilon} \text{Lap}(0, 1) \) is \( \varepsilon \)-differentially private.
Differential Privacy - Post Processing

Data released via a $(\varepsilon, \delta)$-private mechanism is $(\varepsilon, \delta)$-private under arbitrary post processing!
Differential Privacy - Basic Composition

After applying $R$ algorithms with $(\varepsilon_i, \delta_i) : i = 1, \ldots, R$ to the data, the total privacy loss is:

$$\left( \sum_{i=1}^{R} \varepsilon_i, \sum_{i=1}^{R} \delta_i \right)$$

→ privacy loss is additive!

This is a worst-case analysis – better bounds can be found using more elaborated accounting mechanisms (e.g. moments accountant)
We need methods which are both communication-efficient and privacy-preserving!

Differential Privacy:
- adds artificial noise to the parameter updates to obfuscate them

Compression Methods:
- add quantization noise to the parameter updates to reduce the bitwidth

→ combine the two approaches!

Federated Learning - Challenges

Challenges in Federated Learning

- Privacy
- Robustness
- Personalization
- Convergence
- Communication
- Heterogeneity

- ✔ Communication
- ✔ Convergence
- ✔ Heterogeneity
Federated Learning - Meta- and Multi Task-Learning

Federated Learning Environments are characterized by a high degree of statistical heterogeneity of the client data.

→ In many situations, learning one single central model is suboptimal or even undesirable.
Federated Learning - Meta- and Multi Task-Learning

Client data:

\[ p_i(x, y), \ i = 1, \ldots, n \]

\[ p_i(y|x) \text{ shared} \quad \rightarrow \text{one model can be learned} \]

\[ p_i(y|x) \text{ varies} \quad \rightarrow \text{no single model can fit the data of all clients} \]

\[ p_i(x) \text{ shared} \quad p_i(x) \text{ and/or } p_i(y) \text{ varies} \]

\[ \rightarrow \text{IID data} \quad \rightarrow \text{non-IID data} \]
Federated Learning - Meta- and Multi Task-Learning

I like ice cream.

I like Beyoncé.

I like cats.

A linear classifier can correctly separate the data of every single client, but not simultaneously for all clients.
Clustered Federated Learning

Clustered Federated Learning groups the client population into clusters with jointly trainable data distributions and trains a separate model for every cluster.

How to identify the clusters?
Clustered Federated Learning

How to identify the clusters?

→ via the model updates!
At every stationary solution of the Federated Learning objective, the angle between the parameter updates of the different clients is highly indicative of their distribution similarity!
Clustered Federated Learning - Algorithm

1.) Run Federated Learning until convergence to a stationary solution

2.) Compute the **pairwise cosine similarity** between the latest parameter updates from all clients

3.) If there exists a client whose local empirical risk is not sufficiently minimized by the federated learning solution ...

4.) … then bi-partition the client population into two groups of **minimal pairwise similarity**

5.) Repeat everything for the two groups, starting from 1.)

\[
\theta^* \leftarrow \text{FederatedLearning}(\theta, c)
\]

\[
\alpha_{i,j} \leftarrow \frac{\langle \nabla r_i(\theta^*), \nabla r_j(\theta^*) \rangle}{\| \nabla r_i(\theta^*) \| \| \nabla r_j(\theta^*) \|}
\]

\[
\text{if } \max_{i \in c} \| \nabla r_i(\theta^*) \| > \varepsilon \text{ then}
\]

\[
c_1, c_2 \leftarrow \arg \min_{c_1 \cup c_2 = c} \left( \max_{i \in c_1, j \in c_2} \alpha_{i,j} \right)
\]
Clustered Federated Learning - Clustering Guarantees

Let $D_i \sim \varphi_{I(i)}$

$$r_i(\theta) := \frac{1}{|D_i|} \sum_{(x,y) \in D_i} l(f_{\theta}(x), y)$$

$$R_i(\theta) := \int l(f_{\theta}(x), y) d\varphi_{I(i)}(x, y)$$

$$F(\theta) := \sum_{i=1}^{m} \frac{|D_i|}{|D|} r_i(\theta)$$

and $\theta^*$ s.t. $\nabla_{\theta} F(\theta^*) = 0$

Then the proposed mechanism will correctly separate the clients if

$$\max_{i=1,\ldots,m} \gamma_i < \sin\left(\frac{\pi}{4(k-1)}\right)$$

with

$$\gamma_i := \frac{\|\nabla_{\theta} R_{I(i)}(\theta^*) - \nabla_{\theta} r_i(\theta^*)\|}{\|\nabla_{\theta} R_{I(i)}(\theta^*)\|}$$

and $k$ being the number of clusters.
Federated Learning - Clustered Federated Learning

Correct Clustering with $P \approx 1$
Correct Clustering with $P \approx 0$
Correct Clustering
Clustered Federated Learning

Measure cluster quality via:
\[ g(\alpha) := \min_{i,j} \alpha_{i,j} - \min_{I(i)=I(j)} \max_{c_1 \cup c_2 = c, i \in c_1, f \in c_2} \alpha_{i,j} \]

Then: \( g(\alpha) > 0 \Rightarrow \) "correct clustering"
Clustered Federated Learning

Few communication rounds are sufficient in order to obtain a correct clustering.

Work also with weight updates.

Few data points are sufficient to obtain correct clustering.
Clustered Federated Learning

1\textsuperscript{st} Split

2\textsuperscript{nd} Split

3\textsuperscript{rd} Split
Clustered Federated Learning

1) FL has converged to a stationary solution

2) After the 1st split Accuracy drastically increases for the group of clients that was separated out

3) After the third round of splitting $g(a)$ has reduced to below zero for all remaining clusters

Sattler, Müller, Samek. "Clustered Federated Learning: Model-Agnostic Distributed Multi-Task Optimization under Privacy Constraints."
Federated Learning - Clustered Federated Learning

Table 1: Accuracy achieved by conventional Federated Learning and CFL in the four investigated scenarios.

<table>
<thead>
<tr>
<th></th>
<th>Byzantine</th>
<th>Noisy</th>
<th>Label-Flip</th>
<th>Clean</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>FL</td>
<td>9.8%</td>
<td>96.9%</td>
<td>91.3%</td>
</tr>
<tr>
<td></td>
<td>CFL (ours)</td>
<td><strong>93.19%</strong></td>
<td><strong>97.4%</strong></td>
<td><strong>97.4%</strong></td>
</tr>
<tr>
<td>Fashion-MNIST</td>
<td>FL</td>
<td>9.6%</td>
<td>77.12%</td>
<td>60.6%</td>
</tr>
<tr>
<td></td>
<td>CFL (ours)</td>
<td><strong>78.0%</strong></td>
<td><strong>79.7%</strong></td>
<td><strong>79.7%</strong></td>
</tr>
<tr>
<td>CIFAR</td>
<td>FL</td>
<td>10.0%</td>
<td>70.4%</td>
<td>40.1</td>
</tr>
<tr>
<td></td>
<td>CFL (ours)</td>
<td><strong>61.7%</strong></td>
<td><strong>74.6%</strong></td>
<td><strong>74.7%</strong></td>
</tr>
</tbody>
</table>

Sattler, Müller, Samek. “On the Byzantine Robustness of Clustered Federated Learning” (ICASSP 2020)
Federated Learning - Challenges

Challenges in Federated Learning

- Privacy
- Robustness
- Personalization
- Heterogeneity
- Communication
- Convergence
References

Neural Network Compression

http://dx.doi.org/10.1109/JSTSP.2020.2969554

https://arxiv.org/abs/1912.08881


http://dx.doi.org/10.1109/IJCNN.2019.8852119
References

Efficient Deep Learning


References

Federated Learning


References

Federated Learning


Slides and Papers available at

www.federated-ml.org