TOWARDS TAMING THE RESOURCE AND DATA HETEROGENEITY IN FEDERATED LEARNING

Zheng Chai\textsuperscript{1}, Hannan Fayyaz\textsuperscript{2}, Zeshan Fayyaz\textsuperscript{3}, Ali Anwar\textsuperscript{4}, Yi Zhou\textsuperscript{4}, Nathalie Baracaldo\textsuperscript{4}, Heiko Ludwig\textsuperscript{4}, Yue Cheng\textsuperscript{1}

\textsuperscript{1}George Mason University, \textsuperscript{2}York University, \textsuperscript{3}Ryerson University, \textsuperscript{4}IBM Research–Almaden
Federated Learning - Background

• Federated learning - learning in distributed data environments
  • In a federated learning system, multiple data sources collaborate to learn a predictive model
  • Limited trust is likely to exist between each client, so clients do not want other participants to learn their private data in the process

• Limited communication – Data/Resource Heterogeneity
  • Data heterogeneity some clients will have more data
  • Resource heterogeneity some clients may have lesser compute/connectivity

• Clients can drop out during the training
Federated Learning – Problem Statement

$M$: Composition of $R_1, R_2, ..., R_N$

$R$: Gradients
$C$: Client
$D$: Data on client

Issue: Aggregation happens after each client has replied
Federated Learning – Example Problem: Stragglers

Aggregator $A$

- $C_1$: 1 minute
- $C_2$: 0.5 minute
- $C_3$: 2 minutes
- $C_4$: 4 minutes
- $C_5$: 1.5 minutes

$D_5$

- Fast clients
- Moderate clients
- Slow clients
Federated Learning – Example Problem: Dropouts

Aggregator $A$ 4 minutes

$R_1$  
$C_1$  
1 minute

$R_2$  
$C_2$  
0.5 minute

$R_3$  
$C_3$  
2 minutes

$R_4$  
$C_4$  
4 minutes

$R_5$  
$C_5$  
1.5 minutes

$D_5$
Methodology

**Model:** CNN (consists of two CNN layers and one Max Pooling layer)

**Dataset:** MNIST data

**Non-IID:** each client sample data from randomly selected 5 digit categories

**No. of Total Clients:** 100
Setup and Hyperparameters

Platform: AWS EC2 (m4.10xlarge with 40 vCPUs and 160 GB memory)

Library: TensorFlow

Hyperparameters:

Optimizer: Adadelta

Epochs: 8 (each client per round)

Drop out rate: 0.25 & 0.5 for each CNN layer
Impact of resource heterogeneity on training time

<table>
<thead>
<tr>
<th>Test</th>
<th># of Clients</th>
<th># of CPUs</th>
<th>CPUs per Client</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>16</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>1</td>
<td>1/3</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>1</td>
<td>1/5</td>
</tr>
</tbody>
</table>

Table 1: Distribution of data parties and CPUs.

Per-round training time different CPU resources and different dataset sizes.
Impact of data heterogeneity on training time

14 clients with different dataset size (varying from 100–5000 data points) but with the same amount of CPU resources

Training time gets linearly increased as the dataset size gets bigger

Data heterogeneity can significantly impact the FL system’s training time

Per-epoch training time with different dataset sizes
How to classify devices based on their response time and then use this information for our advantage without affecting the FL process?

How to incorporate data of each device in the FL process without worrying about stragglers?

How to identify drop-out devices and mitigate the effect of drop-out without affecting the ML process?
Thanks!