INFOCOM’20

Optimizing Federated Learning on Non-IID Data with Reinforcement Learning

Hao Wang*, Zakhary Kaplan*, Di Niu^, Baochun Li*

*University of Toronto, ^University of Alberta
Federated Learning
Sure. Umami burger?

Yeah. Know the address?

738 E. 3rd St.
Federated Averaging Algorithm (FedAvg)

Initial model

Local data
Random selection

Update global model

Local model

Local data
Thank you for the feedback.
ML algorithms assume the training data is independent and identically distributed (IID)
Federated Learning reuses the existing ML algorithms but on non-IID data
Non-IID data introduces bias into the training and leads to a slow convergence and training failures
MNIST

http://yann.lecun.com/exdb/mnist/
No, we don’t have any access to the data on your phone.

Build IID training data?
Figure 5: (a) Test accuracy vs. EMD for FedAvg and (b) boxplots of weight divergence when EMD = 1.44 for MNIST, CIFAR-10 and KWS datasets. The mean and standard deviation are computed over 5 distributions for each EMD.

Table 2: The mean and standard deviation of the test accuracy of FedAvg over 5 distributions. The standard deviation is very small compared to the scale of the mean value.

<table>
<thead>
<tr>
<th></th>
<th>MNIST</th>
<th>CIFAR-10</th>
<th>KWS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Earth mover's distance (EMD)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0.9857</td>
<td>0.8099</td>
<td>0.8496</td>
</tr>
<tr>
<td>0.36</td>
<td>0.9860</td>
<td>0.8090</td>
<td>0.8461</td>
</tr>
<tr>
<td>0.72</td>
<td>0.9852</td>
<td>0.8017</td>
<td>0.8413</td>
</tr>
<tr>
<td>1.08</td>
<td>0.9835</td>
<td>0.7817</td>
<td>0.8331</td>
</tr>
<tr>
<td>1.44</td>
<td>0.9799</td>
<td>0.7379</td>
<td>0.7979</td>
</tr>
<tr>
<td>1.62</td>
<td>0.9756</td>
<td>0.6905</td>
<td>0.7565</td>
</tr>
<tr>
<td>1.764</td>
<td>0.922</td>
<td>0.5438</td>
<td>0.5827</td>
</tr>
<tr>
<td>1.8</td>
<td></td>
<td>0.4396</td>
<td>0.4475</td>
</tr>
</tbody>
</table>

| Standard deviation (⇥10^-4) | 6.431     | 2.939     | 4.604     |
| Standard deviation (⇥10^-3) | 4.308     | 3.622     | 5.387     |
| Standard deviation (⇥10^-2) | 4.716     | 3.383     | 1.763     |
| Standard deviation (⇥10^-1) | 8.085     | 9.655     | 3.329     |
| Standard deviation (⇥10^-2) | 8.232     | 1.068     | 4.464     |

4 Proposed Solution

In this section, we propose a data-sharing strategy to improve FedAvg with non-IID data by creating a small subset of data which is globally shared between all the edge devices. Experiments show that test accuracy can be increased by ~30% on CIFAR-10 dataset with only 5% globally shared data.

4.1 Motivation

As shown in Figure 5, the test accuracy falls sharply with respect to EMD beyond a certain threshold. Thus, for highly skewed non-IID data, we can significantly increase the test accuracy by slightly reducing EMD. As we have no control on the clients' data, we can distribute a small subset of global data containing a uniform distribution over classes from the cloud to the clients. This fits in with the initialization stage of a typical federated learning setting. In addition, instead of distributing a model with random weights, a warm-up model can be trained on the globally shared data and distributed to the clients. Because the globally shared data can reduce EMD for the clients, the test accuracy is expected to improve.

Optimizing Federated Learning on Non-IID Data with Reinforcement Learning
Build IID training data? No

Peeking into the data distribution on each device without violating data privacy

Probing the bias of non-IID data
Carefully select devices to balance the bias introduced by non-IID data.
Probing the data distribution
Non-IID data

80% data has the same label, e.g., “6”

Initial model

A two-layer CNN model with 431,080 parameters

Local model

100 devices, each has 600 samples
We apply **Principle Component Analysis (PCA)** to reduce dimensionality.

431,080-dimension model weight $\rightarrow$ 2-dimension space
An implicit connection between model weights and data distribution
Probing the data distribution

Selecting devices for federated learning
K-Center Clustering
Random Selection from Groups
Probing the data distribution

Selecting devices for federated learning

How to select devices to speed up training?
It is difficult to select the appropriate subset of devices

- Model weights $\rightarrow$ device selection choice
- A dynamic and undeterministic problem

Reinforcement Learning (RL)
(..., state, action, reward, state', action', ..., end)
(..., state, action, reward, state', action', ..., end)
(..., state, action, reward, state', action', ..., end)
(..., state, action, reward, state', action', ..., end)
(..., state, action, reward, state', action', ..., end)
(..., state, action, reward, state', action', ..., end)
(..., state, action, reward, state', action', ..., end)

::

(..., state, action, reward, state', action', ..., end)
(..., state, action, reward, state', action', ..., end)
Learn to maximize $\text{sum}(\text{reward})$
States

Global weights

Local model weights

100-dimension vector
Select K devices from a pool of N devices — a huge action space

Selecting 10 devices from a pool of 100 devices leads to

$1.7310309e+13$ possible actions
Modify the RL training algorithm
Selecting the Top K Devices

*Only one device* is selected during the RL training.

Now the action space is \( \{1, 2, \ldots, N\} \), instead of selecting K devices from N devices.
Evaluating Each Device

Select the top K

Scores

0.3
0.5
0.1
0.2
Rewards

\[ r_t = \Xi (\omega_t - \Omega) - 1 \]

\[ 0 \leq \omega_t \leq \Omega \leq 1 \]

\[ r_t \in (-1, 0] \]

<table>
<thead>
<tr>
<th>( \Xi )</th>
<th>Positive constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \omega_t )</td>
<td>Training Accuracy</td>
</tr>
<tr>
<td>( \Omega )</td>
<td>Target accuracy</td>
</tr>
<tr>
<td>( t )</td>
<td>Communication round #</td>
</tr>
</tbody>
</table>

👍 Accuracy increase: \(\omega_t \uparrow \rightarrow r_t \uparrow\)

👎 More communication rounds: \(t \uparrow \rightarrow \text{sum}(r_t) \downarrow\)
Training the DRL Agent

Look for a function that points out the actions leading to the maximum cumulative return under a particular state

\[ \text{Max } R = \sum_{t=1}^{T} \gamma^{t-1} r_t = \sum_{t=1}^{T} \gamma^{t-1}(\Xi(\omega_t-\Omega) - 1) \]

discount factor \( \gamma \in (0,1) \)
Action $a_t$  

Environment  

Agent  

DDQN  

Features  

Reward $r_t$  

State $s_{t-1}$  

softmax  

FL server  

$a_t$  

Action
Training the DRL agent
Evaluating Our Solution

Benchmark: MNIST, FashionMNIST, CIFAR-10

Non-IID level: 1, half-and-half, 80%, 50%

Half-and-half

80%
Non-IID level

1

Communication Rounds

MNIST

FashionMNIST

CIFAR-10

FedAvg

K-Center

Favor
Non-IID level

half & half

Communication Rounds

- **FedAvg**
- **K-Center**
- **Favor**

- **MNIST**
- **FashionMNIST**
- **CIFAR-10**
Non-IID level
80%
Non-IID level
50%

Communication Rounds

- **MNIST**
  - FedAvg: 53
  - K-Center: 69
  - Favor: 90

- **FashionMNIST**
  - FedAvg: 18
  - K-Center: 25
  - Favor: 42

- **CIFAR-10**
  - FedAvg: 70
  - K-Center: 69
  - Favor: 70
Indirect data distribution probing
DRL-based device selection
Communication rounds can be reduced by up to

- 49% on the MNIST
- 23% on FashionMNIST
- 42% on CIFAR-10