MSiA 490 Lab 7

Shuyang Wang Fall 2023

Agenda

- Assignment 3 Part 1 due on Nov. 16th by 10pm:

  Submit report on Canvas; submit code and instructions on Github.

- FedAvg
- Parallel clients using Ray Actor

Federated Learning

Federated Learning: Each client has a local training dataset which is never uploaded to the server

Key properties that differentiate it from a typical distributed optimization problem:

1. Non-IID: training data on a given client is typically based on the particular user.
2. Unbalanced classes
3. Massively distributed
4. Limited communication

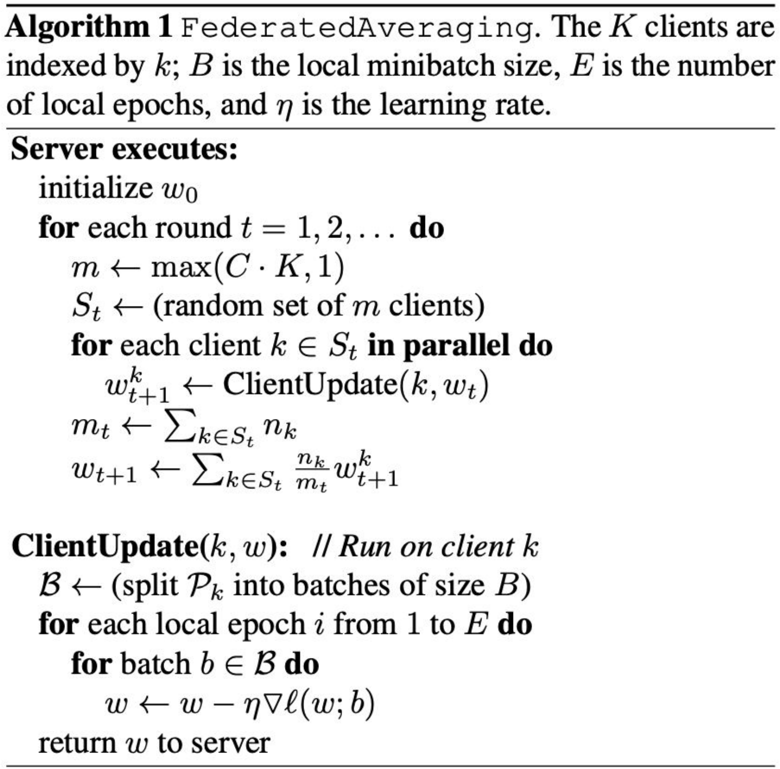
FedAvg

Some suggested components:

- `Client` class:

  Define a Python class Client, which stores local data, performs training and evaluation on the client side.

- A global model as the server
- A helper function that aggregates train/validation metrics from the clients at each round.
- For-loop of communication rounds.



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
      $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
    $m_t \leftarrow \sum_{k \in S_t} n_k$
    $w_{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{m_t} w_{t+1}^k$

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
```

Aggregation

- Model weights of selected clients per round;
- Evaluation metrics:

  1. Let n_k be the number of samples at client k. At each round, compute the total of samples at ***selected*** clients: $N = \sum_{k \in \{selectedclients\}} n_k$

  2. Compute aggregated metrics for tracking

$$aggLoss = \sum_{k \in \{selectedclients\}} \frac{n_k}{N} clientLoss(k)$$

$$aggAccu = \sum_{k \in \{selectedclients\}} \frac{n_k}{N} clientAccu(k)$$