Boulder

# Computer Vision; Image Classification; Federated Learning


**Maziar Raissi**

**Assistant Professor**

Department of Applied Mathematics

University of Colorado Boulder

maziar.raissi@colorado.edu

# Communication-Efficient Learning of Deep Networks from Decentralized Data

Boulder

Federated Learning

sensors on phones & tablets

– cameras – microphones – GPS

clients (local training datasets) & server

Federated Optimization v.s. Distributed Optimization

– Non-IID

– Unbalanced

– Massively distributed

– Limited Communication

$K \rightarrow$ number of clients (each with a fixed local dataset)

$C \rightarrow$ fraction of clients selected at random

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

$f_i(w) = \ell(x_i, y_i; w) \rightarrow$ loss of the prediction

$\mathcal{P}_k \rightarrow$ set of indices of data points on client $k$

$n_k = |\mathcal{P}_k|$

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

$\mathbb{E}_{\mathcal{P}_k}[F_k(w)] = f(w) \rightarrow$ IID assumption

In federated optimization, communication costs dominate!

1) Increase parallelism

2) increase computation on each client

The FederatedAveraging Algorithm

Large-batch synchronous SGD

$C = 1 \rightarrow$ full-batch (non-stochastic) gradient descent

$g_k = \nabla F_k(w_t) \rightarrow$ computed by each client $k$

$$w_{t+1} \leftarrow w_t - \eta \underbrace{\sum_{k=1}^{K} \frac{n_k}{n} g_k}_{\nabla f(w_t)}$$

Equivalently

$$w_{t+1}^k \leftarrow w_t - \eta g_k, \forall k$$

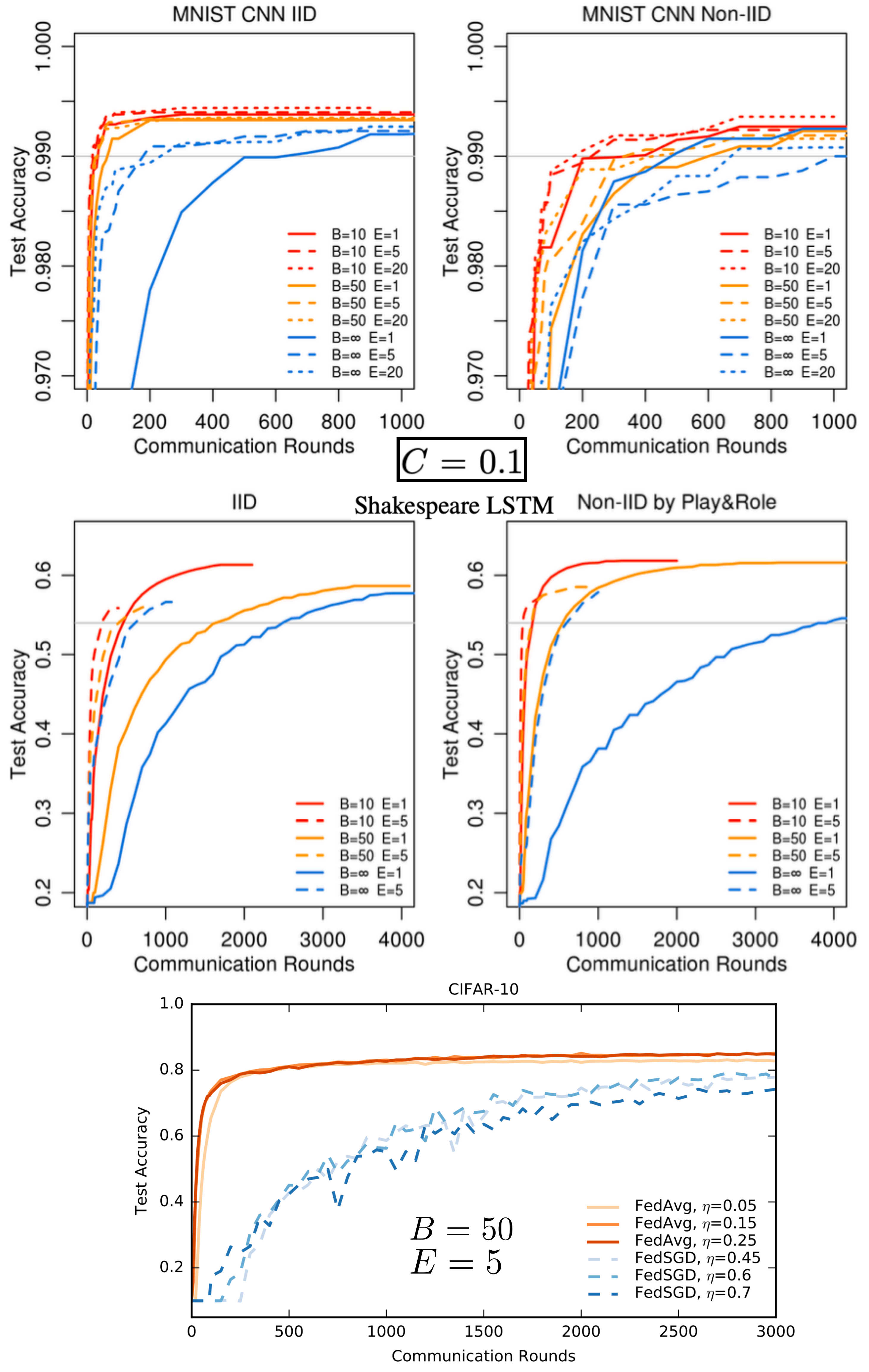
$$w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$$

Iterate the local update multiple times!

**Algorithm 1** `FederatedAveraging`. The $K$ clients are indexed by $k$; $B$ is the local minibatch size, $E$ is the number of local epochs, and $\eta$ 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**
      $w_{t+1}^k \leftarrow$ 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




McMahan, Brendan, et al. "Communication-efficient learning of deep networks from decentralized data." *Artificial Intelligence and Statistics*. PMLR, 2017.

# Questions?