ADAPTIVE FEDERATED OPTIMIZATION

Presented by Mingxi Lei, 03/01/2022

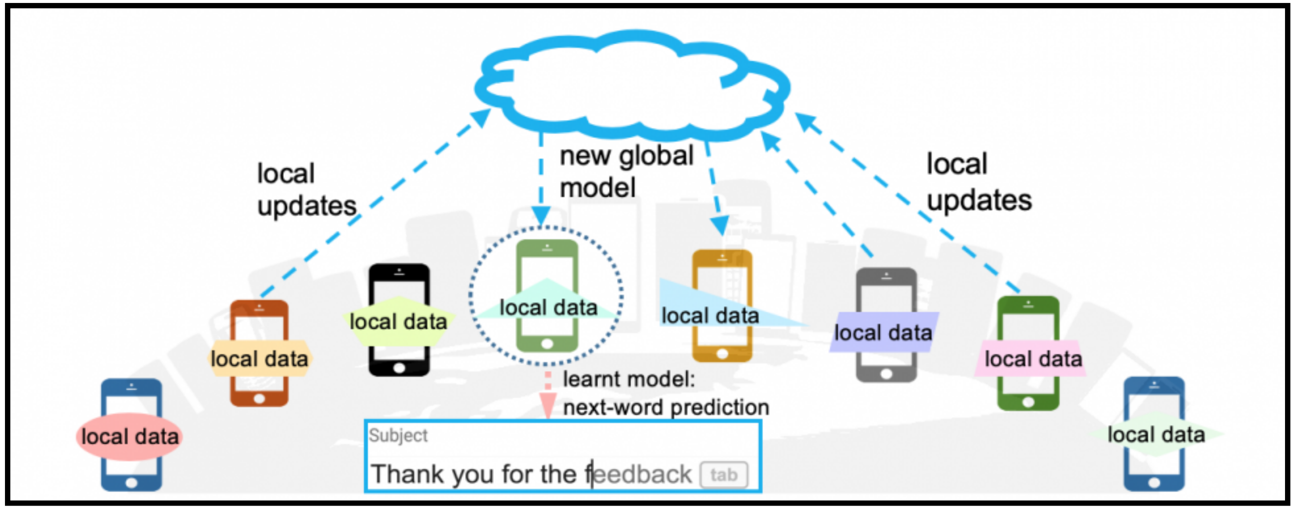
Reddi, Sashank J., et al. "Adaptive Federated Optimization." *International Conference on Learning Representations*, 2021.

List of Contents

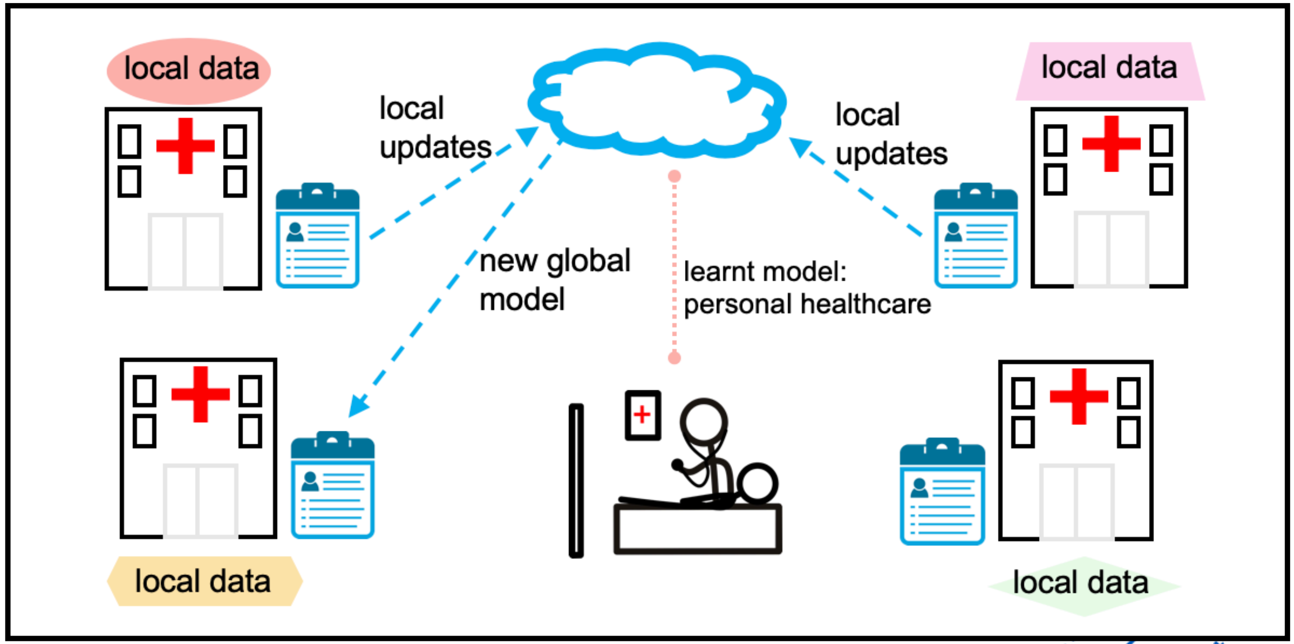
- Introduction
- Motivation
- Problem
- Algorithm
- Theory
- Experiments

Introduction: Federated Learning

Privacy-preserving training in heterogeneous, distributed networks.



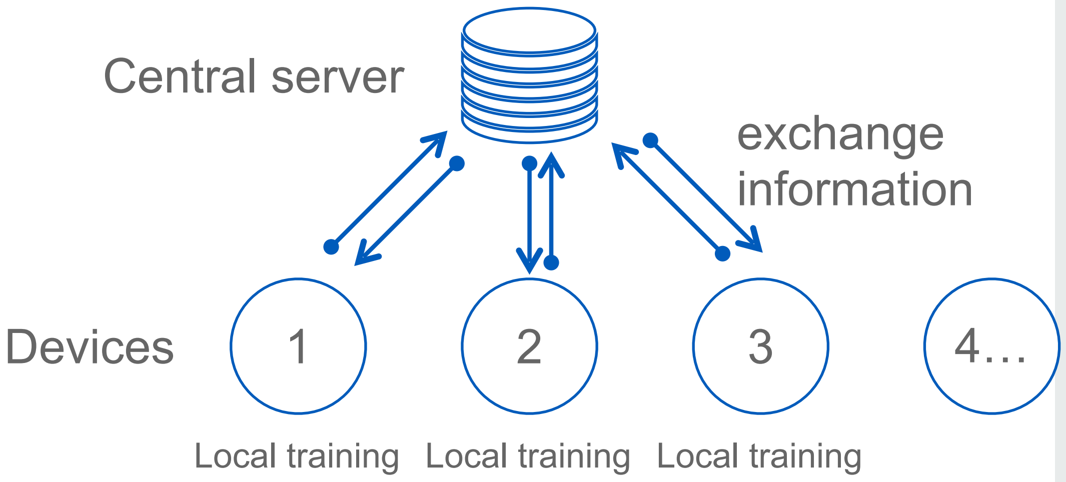
Private mobile data

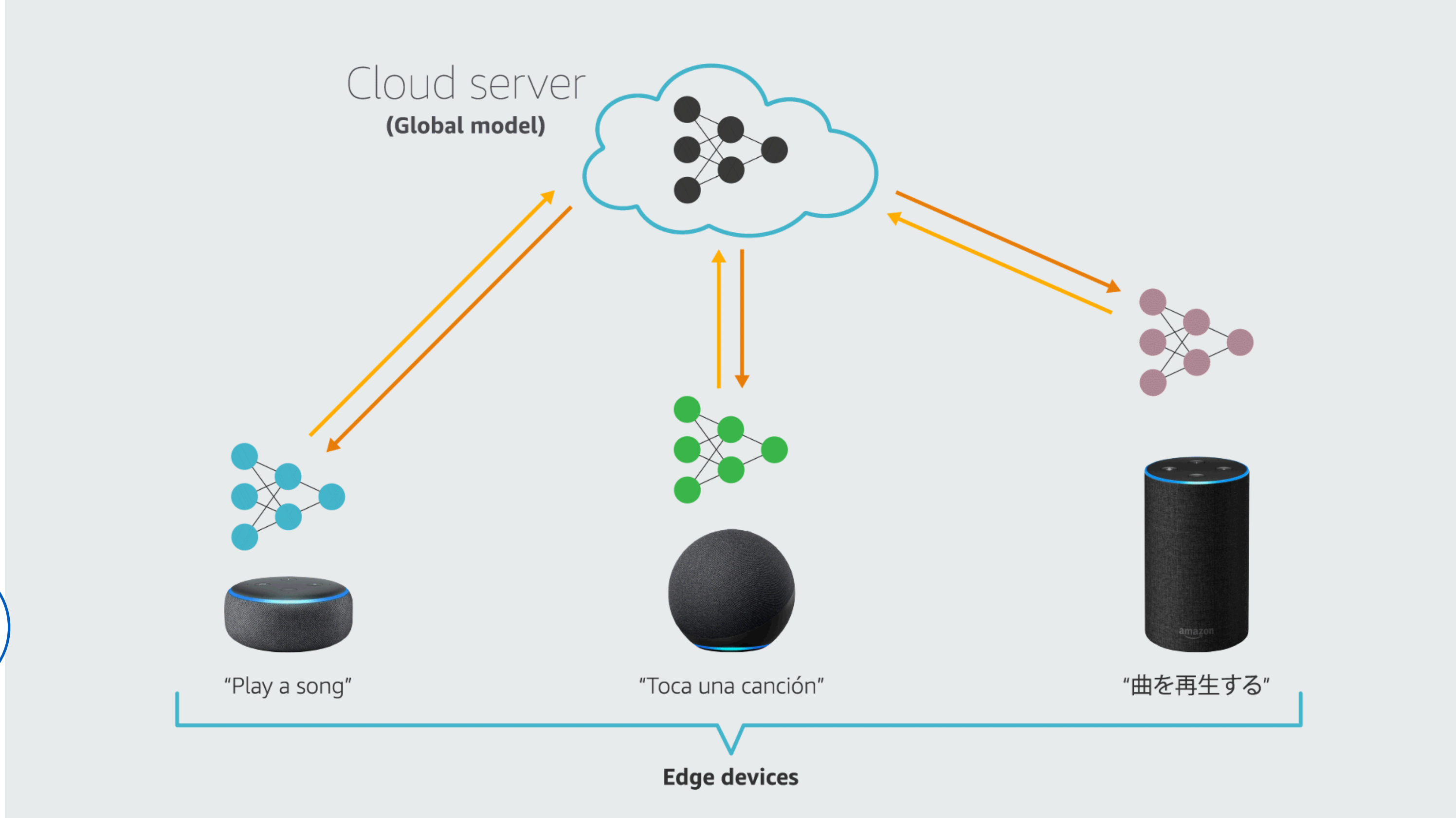


medical records distributed across multiple hospitals

Introduction: Federated Learning

Standard Setup:

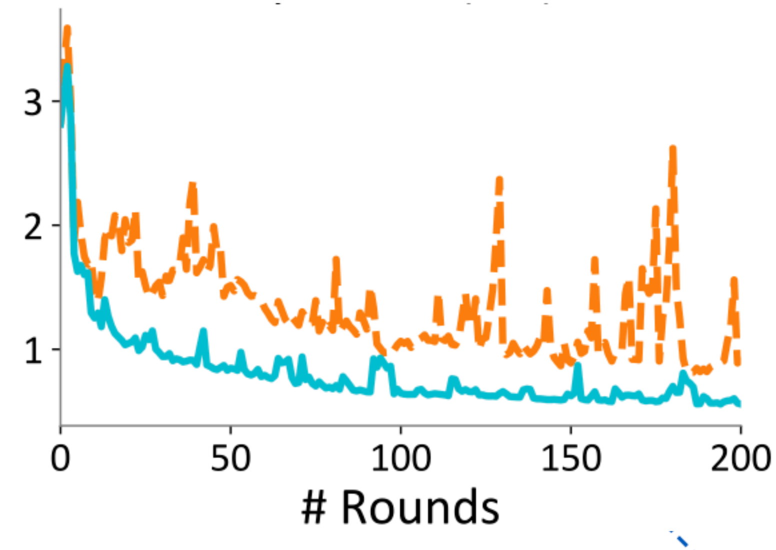




Motivation (Challenge)

- Heterogeneous data, i.e., non-identically distributed (non i.i.d.) data
    - E.g. natural language processing
- Both theoretically and empirically proven

Method	Strongly convex	Non-convex
SGD (large batch)	$\frac{\sigma^2}{\mu N K \epsilon} + \frac{1}{\mu}$	$\frac{\sigma^2}{N K \epsilon^2} + \frac{1}{\epsilon}$
FedAvg		
(Li et al., 2019b)	$\frac{\sigma^2}{\mu^2 N K \epsilon} + \frac{G^2 K}{\mu^2 \epsilon}$	–
(Yu et al., 2019)	–	$\frac{\sigma^2}{N K \epsilon^2} + \frac{G^2 N K}{\epsilon}$
(Khaled et al., 2020)	$\frac{\sigma^2 + G^2}{\mu N K \epsilon} + \frac{\sigma + G}{\mu \sqrt{\epsilon}} + \frac{N B^2}{\mu}$	–



Karimireddy, Sai Praneeth, et al. "Scaffold: Stochastic controlled averaging for federated learning." *International Conference on Machine Learning*. PMLR, 2020.

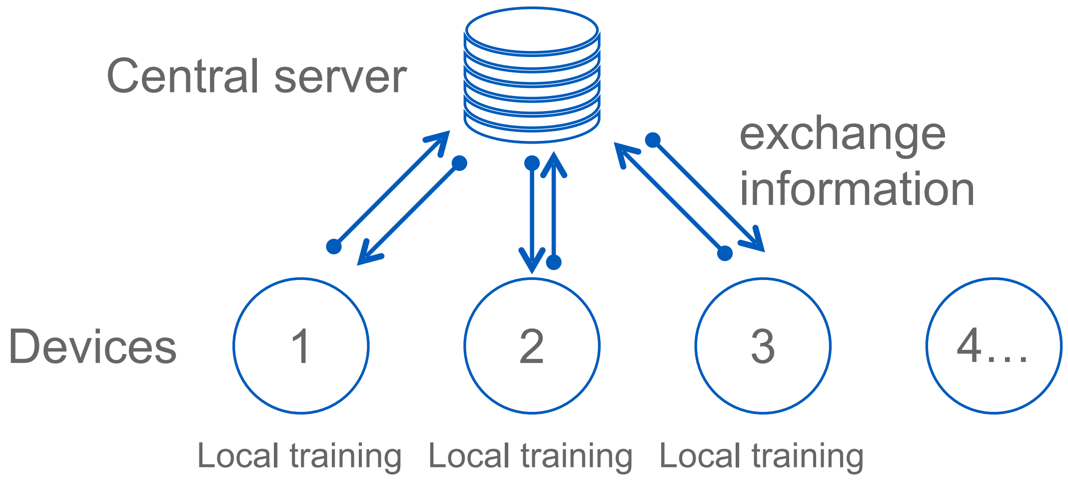
Problem Formulation

$$\min_{w} f(w) = \sum_{k=1}^{N} p_k F_k(w) = \mathbb{E}_k \left[F_k(w)\right]$$

$f(w)$: Global objective

$F_k(w)$: Local objective for device k

p_k: Weights for device k



Adaptive optimization

$$x_{t+1} = \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} x_i^t = x_t - \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} (x_t - x_i^t).$$

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**
    - $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
  - $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$

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

Algorithm 1 FedOpt
- Input: x_0, ClientOpt, ServerOpt
- **for** $t = 0, \cdots, T-1$ **do**
  - Sample a subset $\mathcal{S}$ of clients
  - $x_{i,0}^t = x_t$
  - **for each client** $i \in \mathcal{S}$ **in parallel do**
    - **for** $k = 0, \cdots, K-1$ **do**
      - Compute an unbiased estimate $g_{i,k}^t$ of $\nabla F_i(x_{i,k}^t)$
      - $x_{i,k+1}^t = \text{ClientOpt}(x_{i,k}^t, g_{i,k}^t, \eta_l, t)$
    - $\Delta_i^t = x_{i,K}^t - x_t$
  - $\Delta_t = \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} \Delta_i^t$
  - $x_{t+1} = \text{ServerOpt}(x_t, -\Delta_t, \eta, t)$

Adaptive optimization

Algorithm 2 FedAdagrad, FedYogi, and FedAdam

Initialization: $x_0, v_{-1} \geq \tau^2$, decay parameters $\beta_1, \beta_2 \in [0, 1)$
for $t = 0, \cdots, T-1$ **do**
  Sample subset $\mathcal{S}$ of clients
  $x_{i,0}^t = x_t$
  for each client $i \in \mathcal{S}$ **in parallel do**
    for $k = 0, \cdots, K-1$ **do**
      Compute an unbiased estimate $g_{i,k}^t$ of $\nabla F_i(x_{i,k}^t)$
      $x_{i,k+1}^t = x_{i,k}^t - \eta_l g_{i,k}^t$
    $\Delta_i^t = x_{i,K}^t - x_t$
  $\Delta_t = \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \Delta_i^t$
  $m_t = \beta_1 m_{t-1} + (1-\beta_1)\Delta_t$
  $v_t = v_{t-1} + \Delta_t^2$ (**FedAdagrad**)
  $v_t = v_{t-1} - (1-\beta_2)\Delta_t^2 \operatorname{sign}(v_{t-1} - \Delta_t^2)$ (**FedYogi**)
  $v_t = \beta_2 v_{t-1} + (1-\beta_2)\Delta_t^2$ (**FedAdam**)
  $x_{t+1} = x_t + \eta \frac{m_t}{\sqrt{v_t} + \tau}$

Theoretical analysis

Corollary 1. *Suppose η_l is such that the conditions in Theorem 1 are satisfied and $\eta_l = \Theta(1/(KL\sqrt{T}))$. Also suppose $\eta = \Theta(\sqrt{Km})$ and $\tau = G/L$. Then, for sufficiently large T, the iterates of Algorithm 2 for* FedAdagrad *satisfy*

$$\min_{0 \le t \le T-1} \mathbb{E}\|\nabla f(x_t)\|^2 = \mathcal{O}\left(\frac{f(x_0) - f(x^*)}{\sqrt{mKT}} + \frac{2\sigma_l^2 L}{G^2\sqrt{mKT}} + \frac{\sigma^2}{GKT} + \frac{\sigma^2 L\sqrt{m}}{G^2\sqrt{K}T^{3/2}}\right).$$

Theoretical analysis

Corollary 2. *Suppose η_l is chosen such that the conditions in Theorem 2 are satisfied and that $\eta_l = \Theta(1/(KL\sqrt{T}))$. Also, suppose $\eta = \Theta(\sqrt{Km})$ and $\tau = G/L$. Then, for sufficiently large T, the iterates of Algorithm 2 for* FedAdam *satisfy*

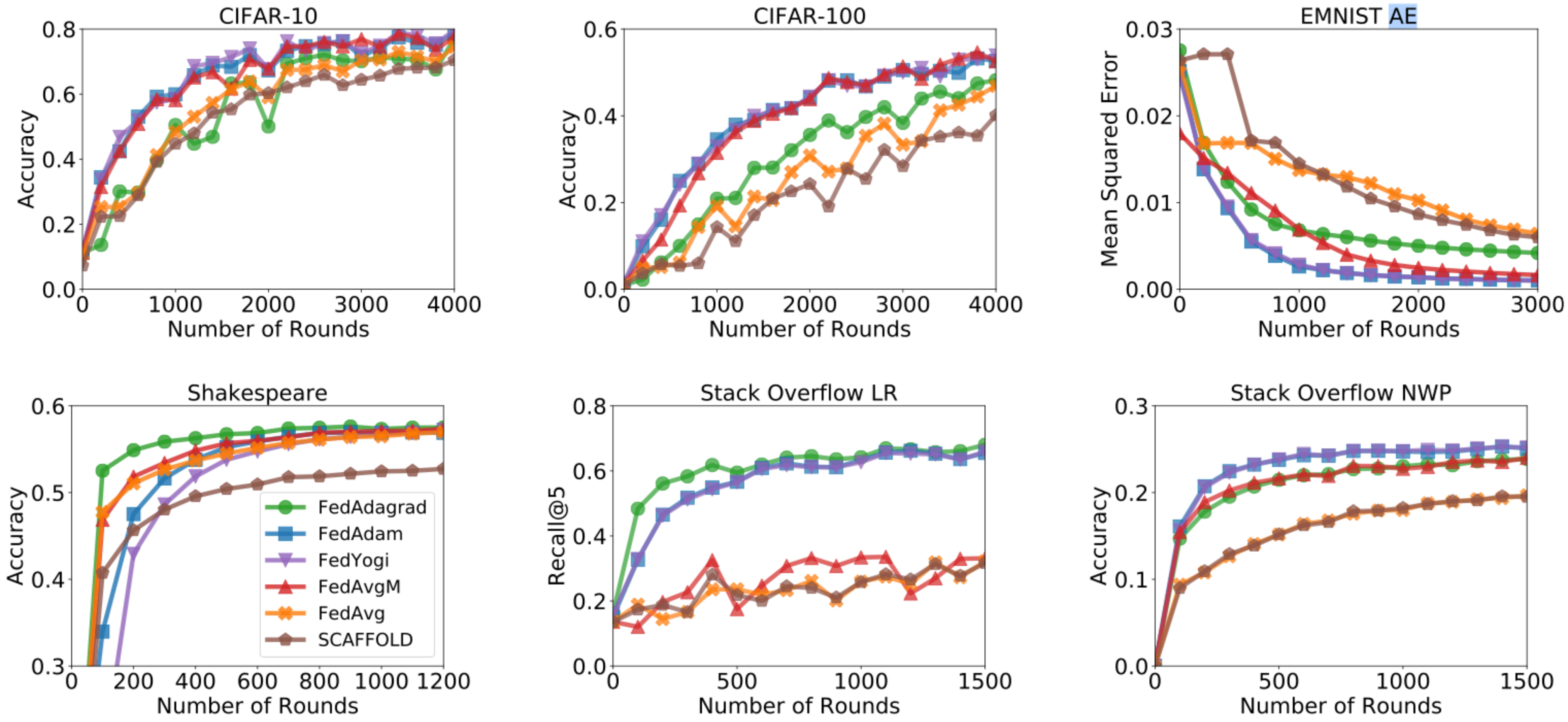
$$\min_{0 \le t \le T-1} \mathbb{E}\|\nabla f(x_t)\|^2 = \mathcal{O}\left(\frac{f(x_0) - f(x^*)}{\sqrt{mKT}} + \frac{2\sigma_l^2 L}{G^2\sqrt{mKT}} + \frac{\sigma^2}{GKT} + \frac{\sigma^2 L\sqrt{m}}{G^2\sqrt{K}T^{3/2}} \right).$$

Theoretical analysis - takeaway

- Best known convergence rate: $\mathcal{O}({}^{1}/\sqrt{mKT})$
  - m: total number of clients
  - K: number of local updates
  - T: number of global updates
- learning rate decay can improve empirical performance
- the effect of client heterogeneity can be reduced by carefully choosing client and server learning rates (on convergence can be reduced by choosing sufficiently η_l and a reasonably large η)

Experiment

Dataset	Model	Task Summary
CIFAR-100	ResNet-18 (with GroupNorm layers)	Image classification
EMNIST	Bottleneck network	Autoencoder
EMNIST	CNN (with dropout)	Character recognition
Shakespeare	RNN with 2 LSTM layers	Next-character prediction
Stack Overflow	RNN with 1 LSTM layer	Next-word prediction
Stack Overflow	Logistic regression classifier	Tag prediction


CIFAR-10
Accuracy
Number of Rounds
CIFAR-100
Accuracy
Number of Rounds
EMNIST AE
Mean Squared Error
Number of Rounds
Shakespeare
Accuracy
Number of Rounds
FedAdagrad
FedAdam
FedYogi
FedAvgM
FedAvg
SCAFFOLD
Stack Overflow LR
Recall@5
Number of Rounds
Stack Overflow NWP
Accuracy
Number of Rounds

FED...	ADAGRAD	ADAM	YOGI	AVGM	AVG
CIFAR-10	72.1	77.4	**78.0**	77.4	72.8
CIFAR-100	47.9	**52.5**	**52.4**	**52.4**	44.7
EMNIST CR	85.1	**85.6**	**85.5**	**85.2**	84.9
SHAKESPEARE	**57.5**	57.0	**57.2**	**57.3**	56.9
SO NWP	23.8	**25.2**	**25.2**	23.8	19.5
SO LR	**67.1**	65.8	65.9	36.9	30.0
EMNIST AE	4.20	1.01	**0.98**	1.65	6.47

Table 1: Average validation performance over the last 100 rounds: % accuracy for rows 1–5; Recall@5 ($\times$100) for Stack Overflow LR; and MSE ($\times$1000) for EMNIST AE. Performance within 0.5% of the best result for each task are shown in bold.

Ease of tuning

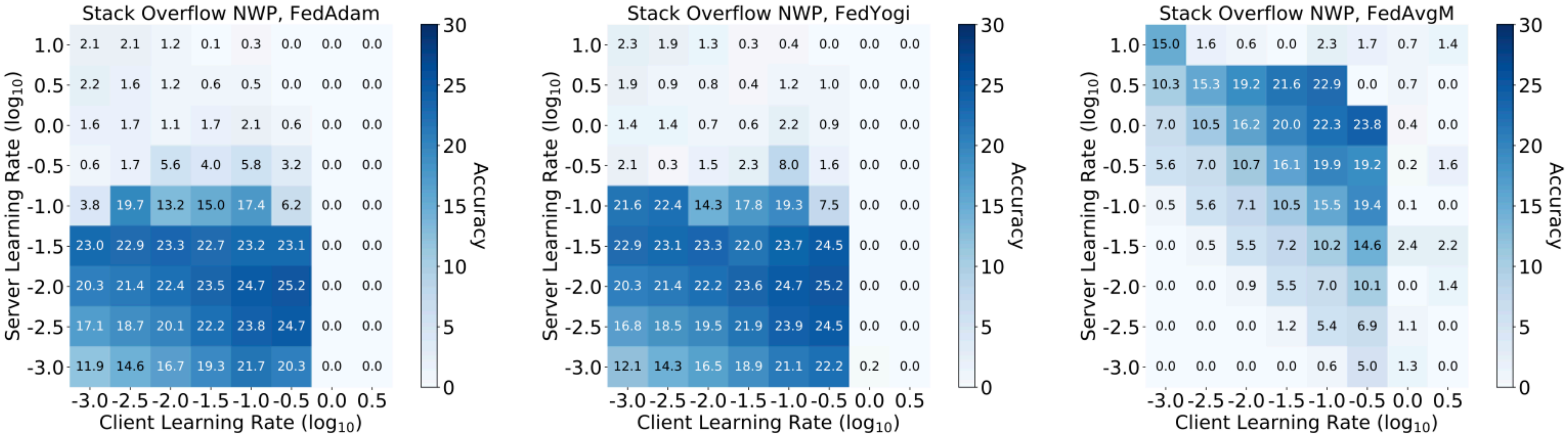


Figure 2: Validation accuracy (averaged over the last 100 rounds) of FEDADAM, FEDYOGI, and FEDAVGM for various client/server learning rates combination on the SO NWP task. For FEDADAM and FEDYOGI, we set $\tau = 10^{-3}$.

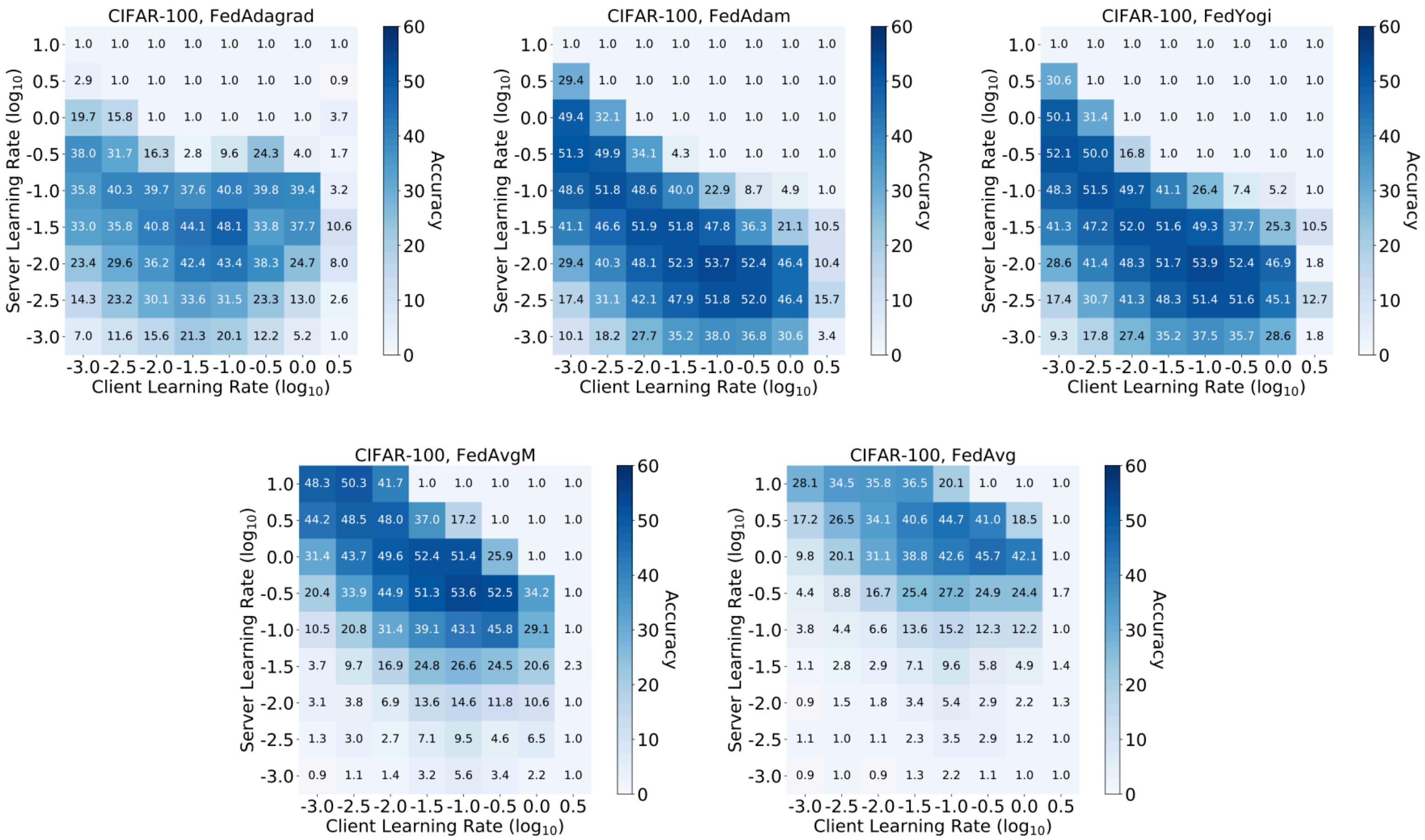
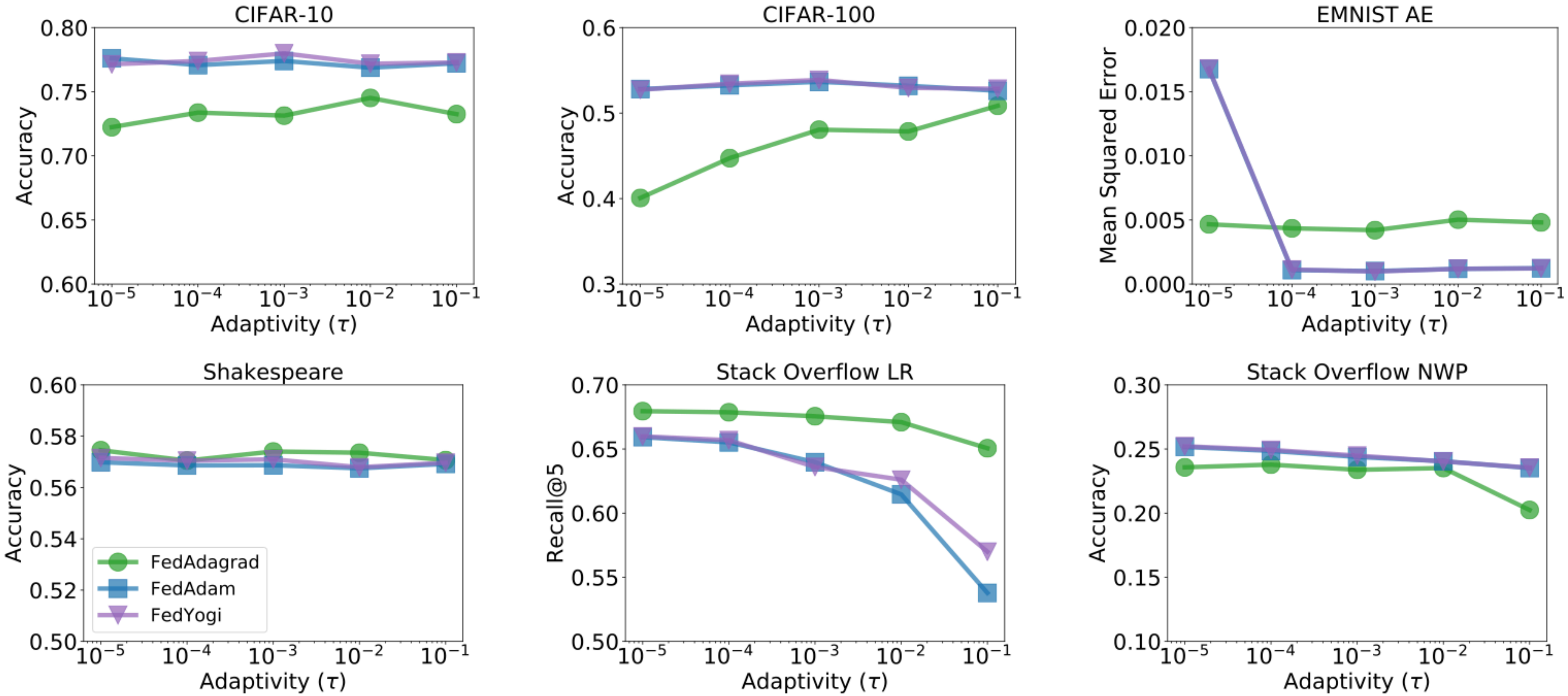


Figure 6: Validation accuracy (averaged over the last 100 rounds) of FEDADAGRAD, FEDADAM, FEDYOGI, FEDAVGM, and FEDAVG for various client/server learning rates combination on the CIFAR-100 task. For FEDADAGRAD, FEDADAM, and FEDYOGI, we set $\tau = 10^{-3}$.

Ease of tuning



Learning rate decay

Table 11: (Top) Test accuracy (%) of a model trained centrally with various optimizers. (Bottom) Average test accuracy (%) over the last 100 rounds of various federated optimizers on the EMNIST CR task, using constant learning rates or the EXPDECAY schedule for η_l. Accuracies (for the federated tasks) within 0.5% of the best result are shown in bold.

	ADAGRAD	ADAM	YOGI	SGDM	SGD
CENTRALIZED	88.0	87.9	88.0	87.7	87.7
FED...	ADAGRAD	ADAM	YOGI	AVGM	AVG
CONSTANT η_l	85.1	85.6	85.5	85.2	84.9
EXPDECAY	85.3	**86.2**	**86.2**	**85.8**	85.2

Summary

- Adaptive optimization in federated learning
- Faster convergence
- Easy to tune

Reference

Karimireddy, Sai Praneeth, et al. "Scaffold: Stochastic controlled averaging for federated learning." *International Conference on Machine Learning*. PMLR, 2020.