Introduction to Federated Learning

Peiwen Qiu

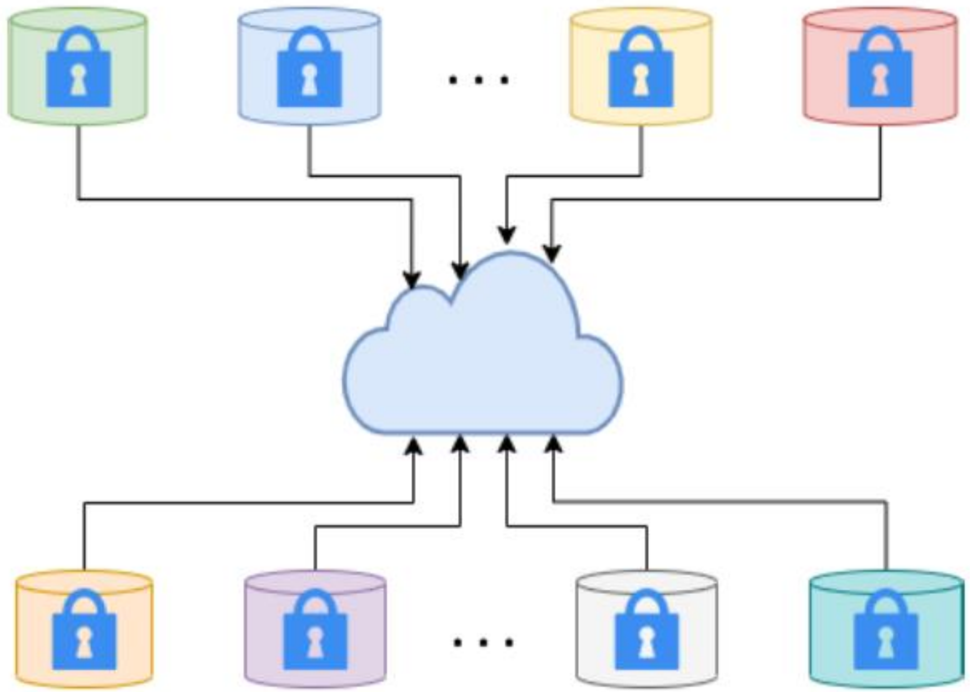
Dept. of Electrical and Computer Engineering

The Ohio State University

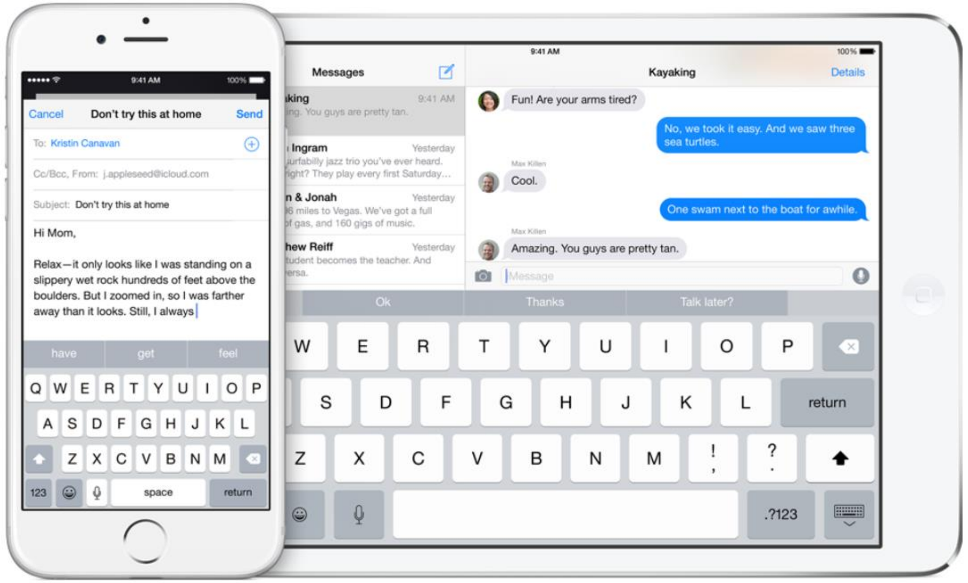
July 17, 2025

Motivation

- Decentralized data
  - Billions of phones & IoT devices constantly generate data
- Data privacy preserving
- Local device hardware resources
  - Improved latency



Applications



- Google Gboard
- Apple QuickType

Next-word prediction

Applications

- Voice assistant Siri

*"Instead, **it relies primarily on a technique called federated learning**, Apple's head of privacy, Julien Freudiger, told an audience at the Neural Processing Information Systems conference on December 8."*

"It allows Apple to train different copies of a speaker recognition model across all its users' devices, using only the audio data available locally."

MIT Technology Review Sign in Subscribe

Artificial intelligence / Machine learning

How Apple personalizes Siri without hoovering up your data

The tech giant is using privacy-preserving machine learning to improve its voice assistant while keeping your data on your phone.

by **Karen Hao** December 11, 2019

Terminology

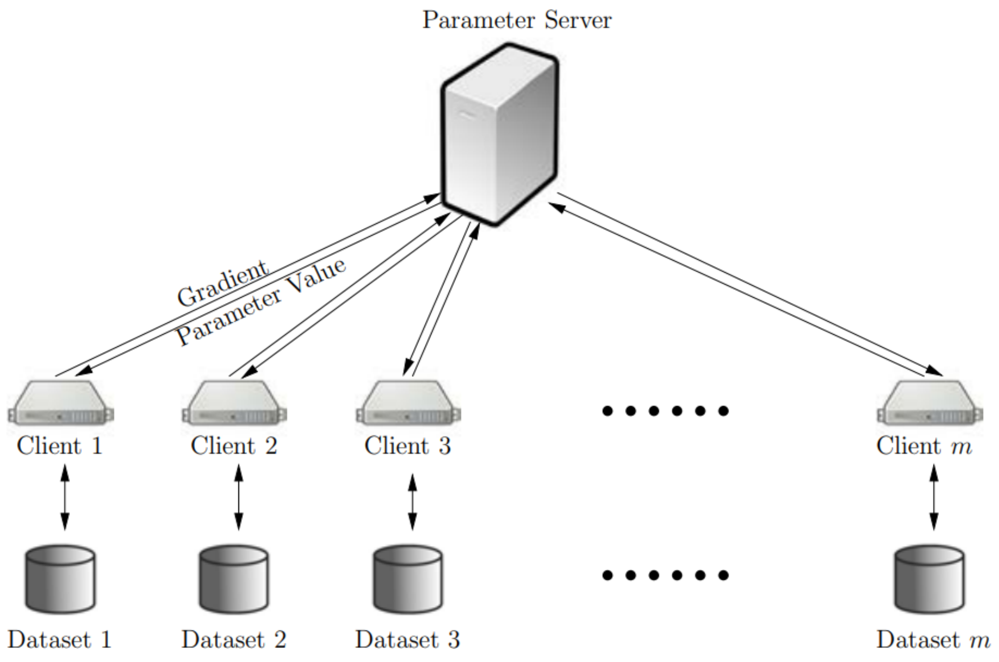
- **Clients** - Compute nodes also holding local data, usually belonging to one entity:
  - IoT devices
  - Mobile devices
  - Data silos
- **Server** - Additional compute nodes that coordinate the FL process but don’t access raw data.
  - Usually not a single physical machine
  - Virtual/cloud-based instances, e.g., AWS

Definition

Federated learning (FL) is a machine learning setting where multiple clients collaborate in solving a ML problem, under the coordination of a central server. Each client's raw data is stored locally and not exchanged or transferred; instead, updates intended for immediate aggregation are used to achieve the learning objective.

Characteristics

- Data is generated locally and remains decentralized.
- Each client stores its own data and cannot read the data of other clients.
- Data is not independently or identically distributed (non-IID).
- A central server coordinates the training, but never sees raw data.



IID Data vs Non-IID Data

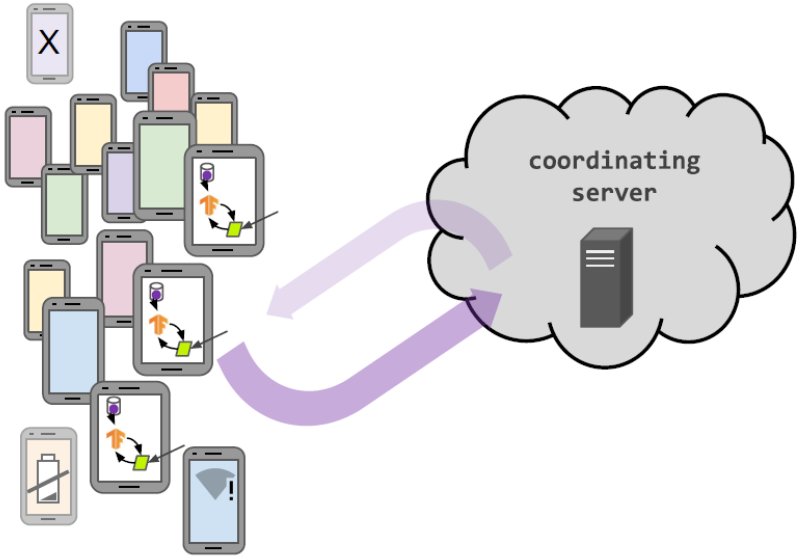
Two Main Settings

- **Cross-device** federated learning
  - Huge number of (unreliable) clients (e.g., mobile devices)
- **Cross-silo** federated learning
  - Small number of (relatively) reliable clients (hospitals, banks, etc.)

Cross-Device FL vs Cross-Silo FL

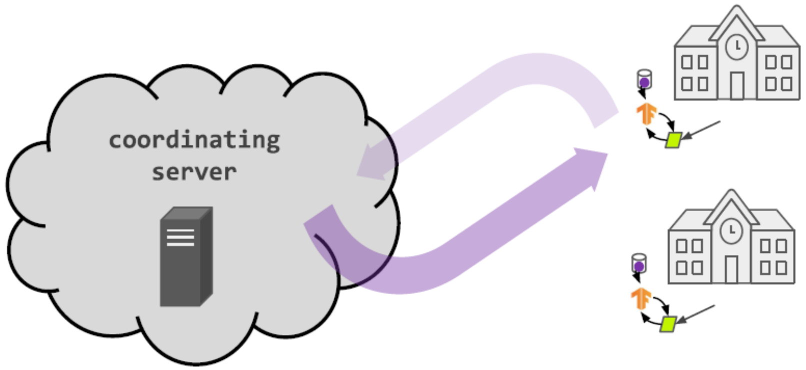
Cross-device federated learning

millions of intermittently
available client devices



Cross-silo federated learning

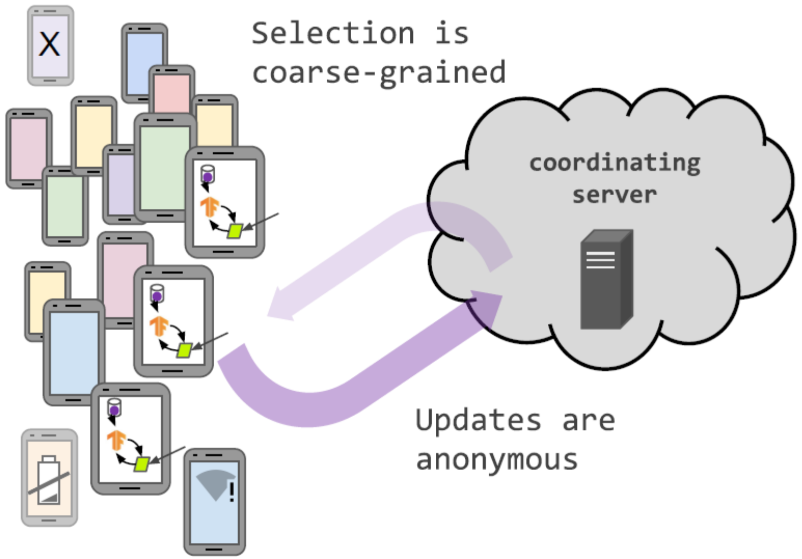
small number of clients
(institutions, data silos),
high availability



Cross-Device FL vs Cross-Silo FL

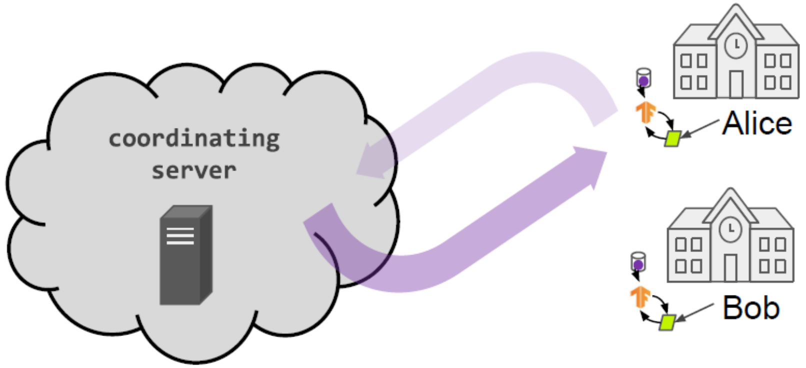
Cross-device federated learning

clients cannot be indexed directly (i.e., no use of client identifiers)



Cross-silo federated learning

each client has an identity or name that allows the system to access it specifically

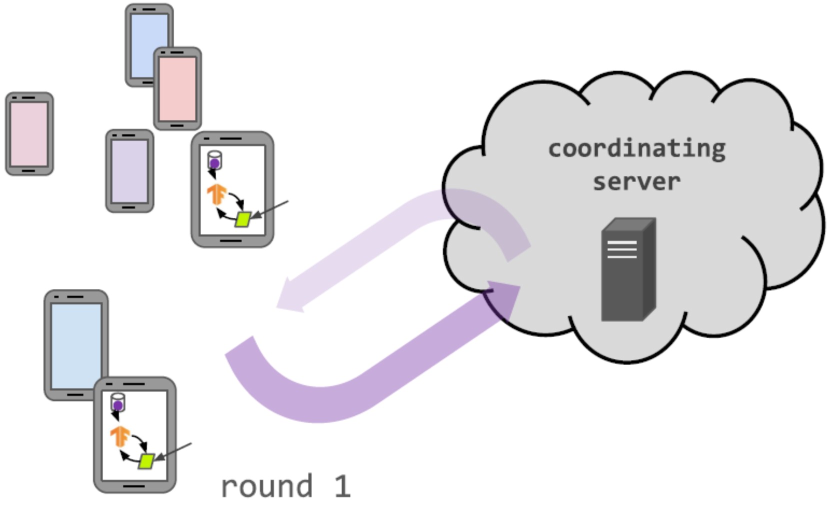


Cross-Device FL vs Cross-Silo FL

Cross-device federated learning

Server can only access a (possibly biased) random sample of clients on each round.

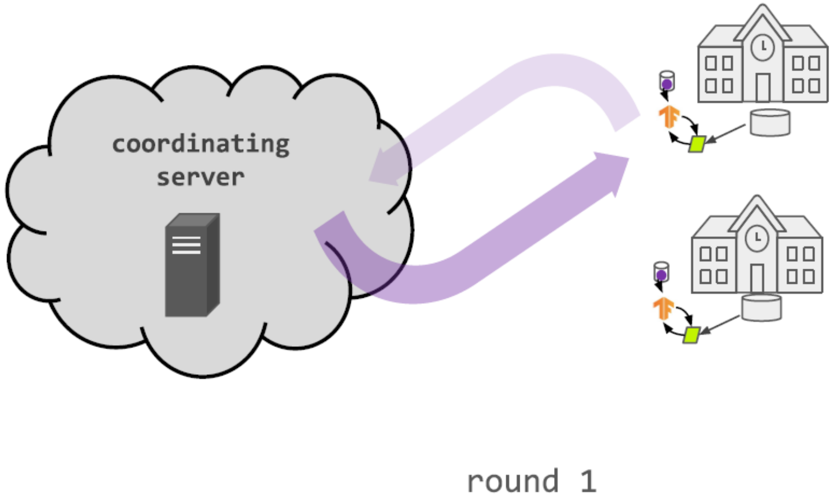
Large population => most clients only participate once.



Cross-silo federated learning

Most clients participate in every round.

Clients can run algorithms that maintain local state across rounds.

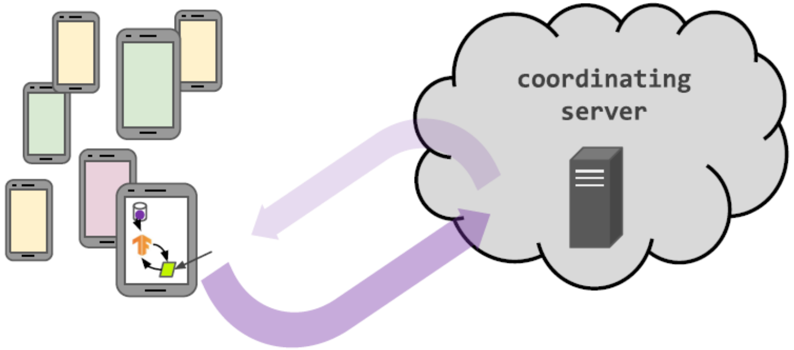


Cross-Device FL vs Cross-Silo FL

Cross-device federated learning

Server can only access a (possibly biased) random sample of clients on each round.

Large population => most clients only participate once.

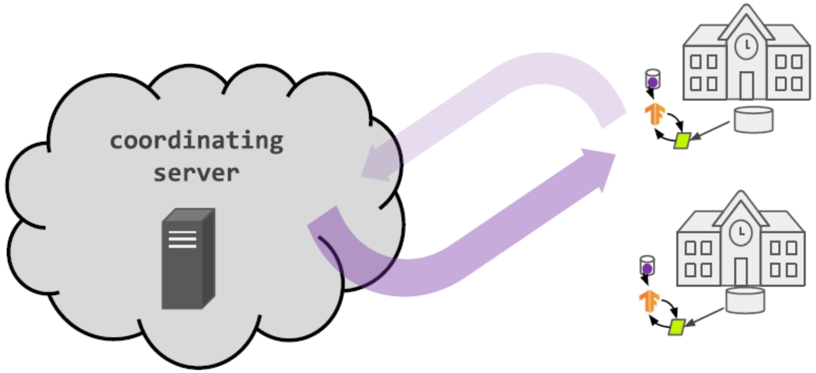


round 2
(completely new set of devices participate)

Cross-silo federated learning

Most clients participate in every round.

Clients can run algorithms that maintain local state across rounds.



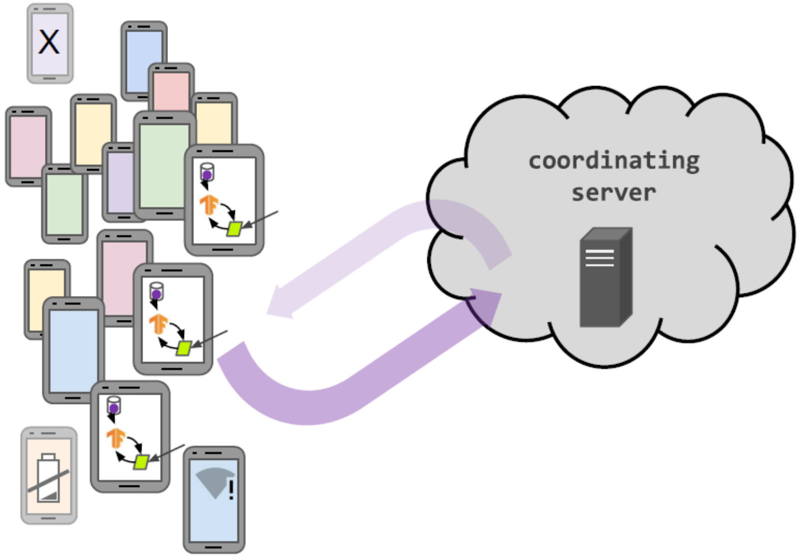
round 2
(same clients)

Cross-Device FL vs Cross-Silo FL

Cross-device federated learning

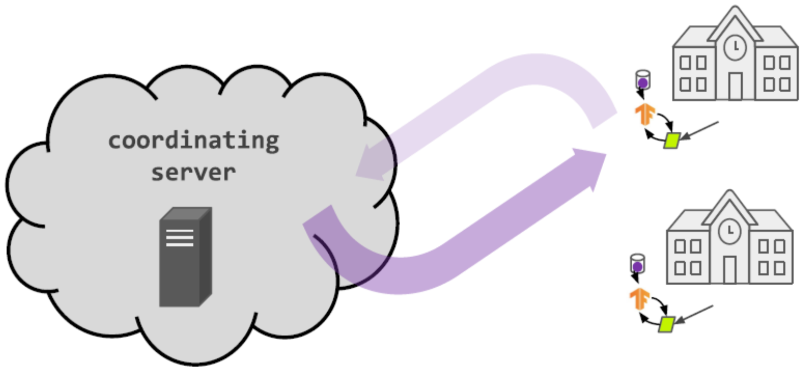
communication is often the primary bottleneck

connection might be another bottleneck

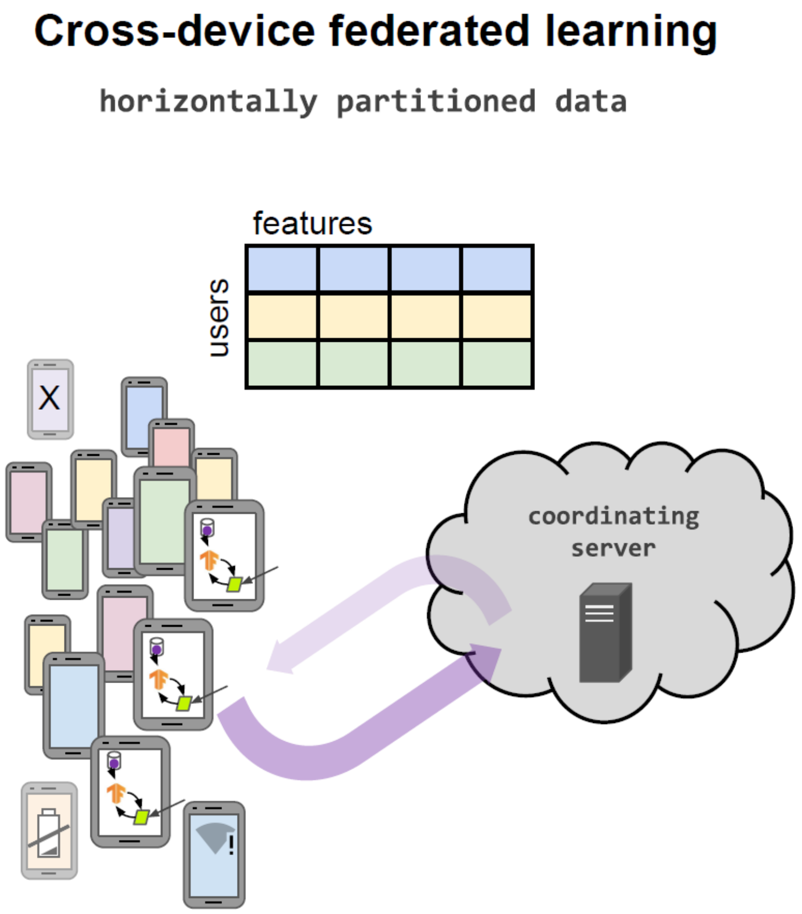


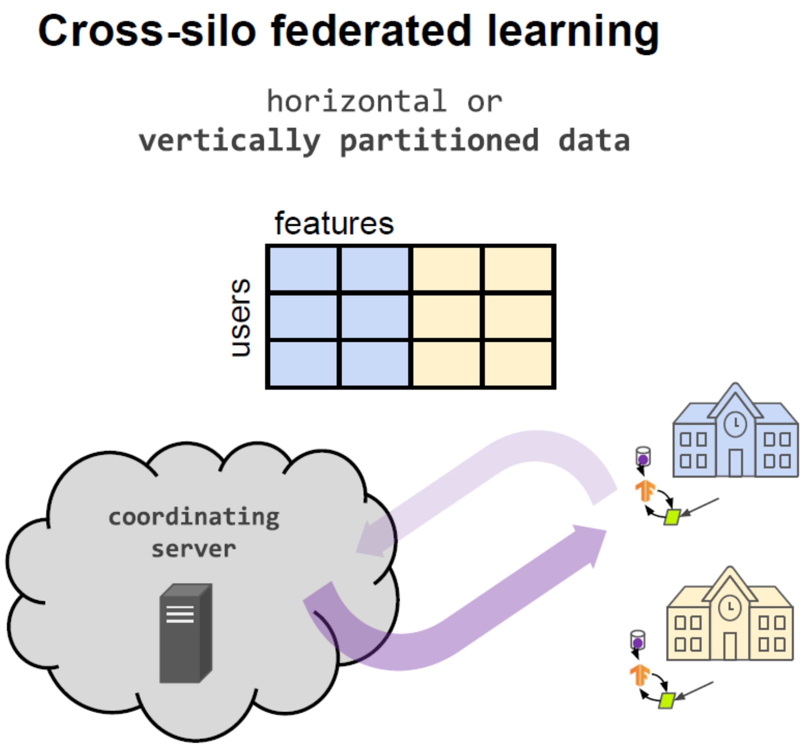
Cross-silo federated learning

communication or computation might be the primary bottleneck



Cross-Device FL vs Cross-Silo FL





Summary of Differences

	Cross-device FL	Cross-silo FL
Example	mobile or IoT devices	medical or financial institutes
Data availability	available only a fraction of clients	available all clients
Distribution scale	massively parallel	2-100 clients
Addressability	not accessible	accessible to client ids
Client statefulness	stateless	stateful
Client reliability	highly unreliable	relatively few failures
Primary bottleneck	connection and communication	computation or communication
Data partition axis	fixed (HFL)	fixed (HFL&VFL)

Federated Averaging (FedAvg) Algorithm

- The first approach to federated learning (FL).
- Simply extend **SGD** to FL setting by **averaging**.
- Reduce communication by:
  - performing local updating
  - communicating with a subset of devices

Objective

- Goal: minimize weighted average of losses across K clients and their local data

$$\min_{\mathbf{w}} F(\mathbf{w}) = \sum_{k=1}^{K} p_k \boxed{f_k(\mathbf{w})}$$

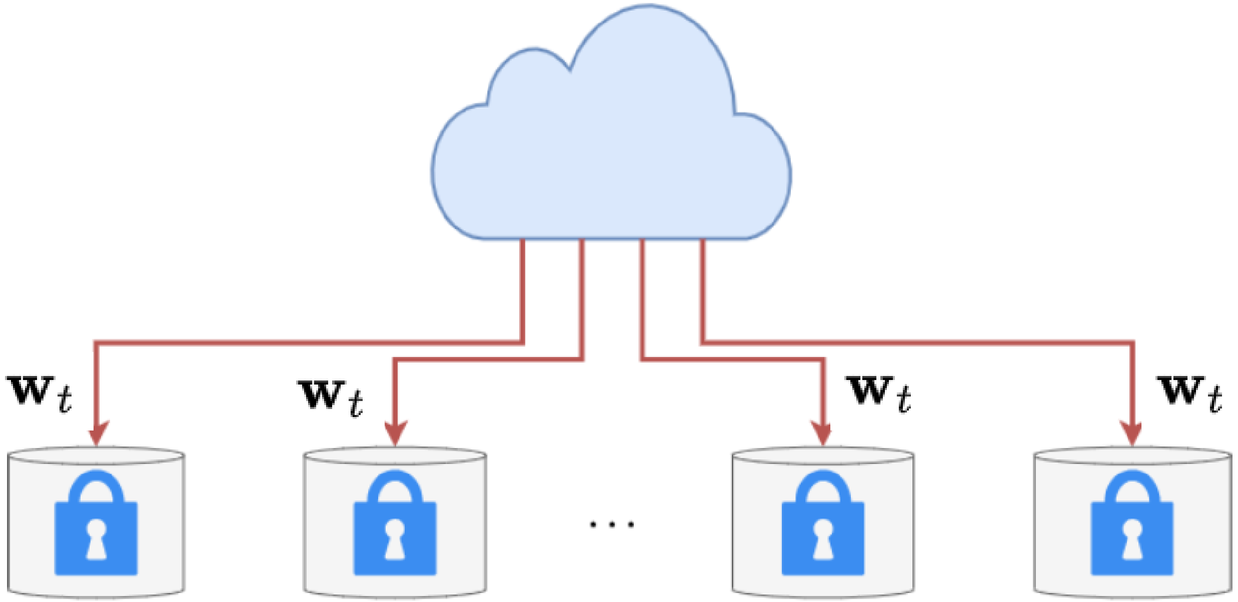
loss of k-th client

$$p_k = \frac{\boxed{n_k}}{\sum_{i=1}^{K} n_i}$$

number of local data points of k-th client

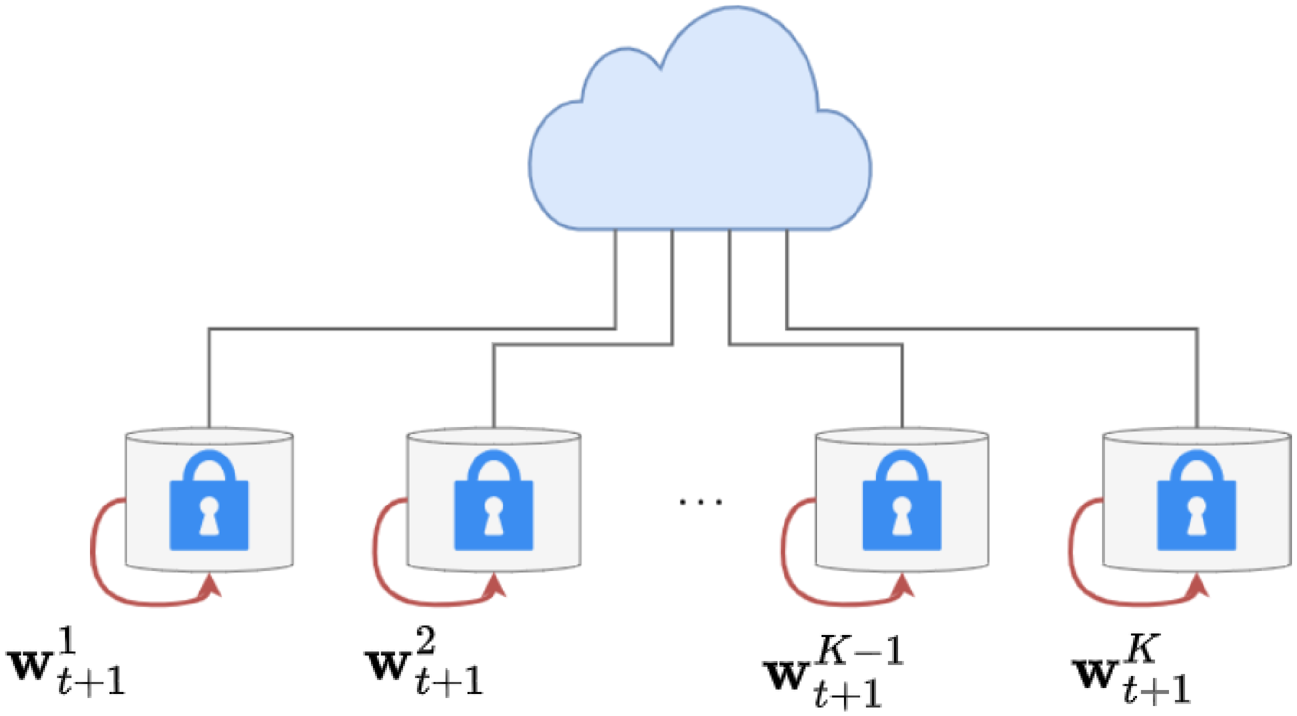
Workflow of FedAvg – Client-Side

- Step 1: Get the global model



Workflow of FedAvg – Client-Side

- Step 2: Local training – E **epochs of SGD**



For each Client k in parallel do

$$\mathbf{w}_t^0 = \mathbf{w}_t$$

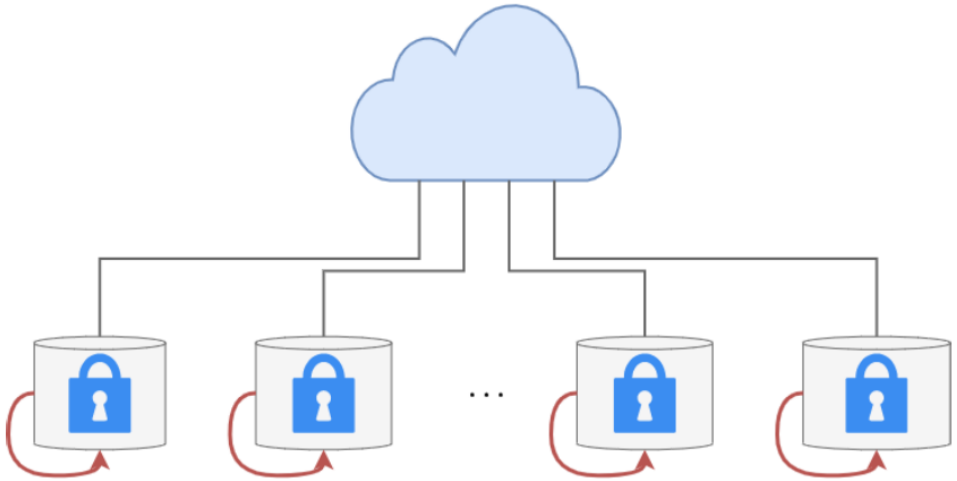
For $m = 0$ to $E - 1$, do

$$\mathbf{w}_t^{m+1} = \mathbf{w}_t^m - \eta \nabla f_k\left(\mathbf{w}_t^m\right)$$

$$\mathbf{w}_{t+1}^k = \mathbf{w}_t^E$$

Workflow of FedAvg – Client-Side

- Step 2: Local training – $\boldsymbol{E}$ **epochs of SGD**

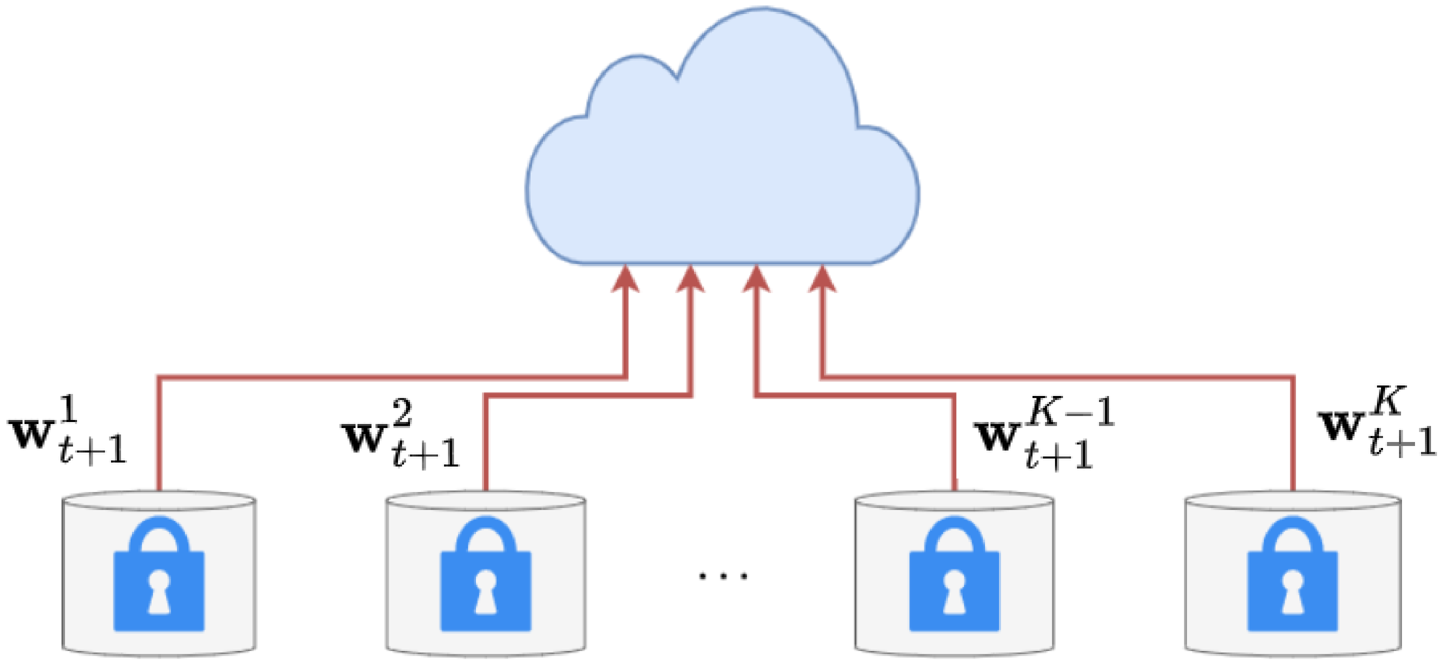


Benefits:

- Can perform more local computation (i.e., more than just one mini-batch)
- Incorporate updates more quickly (immediately apply gradient information)
- Can lead to algorithm converging in many fewer communication rounds

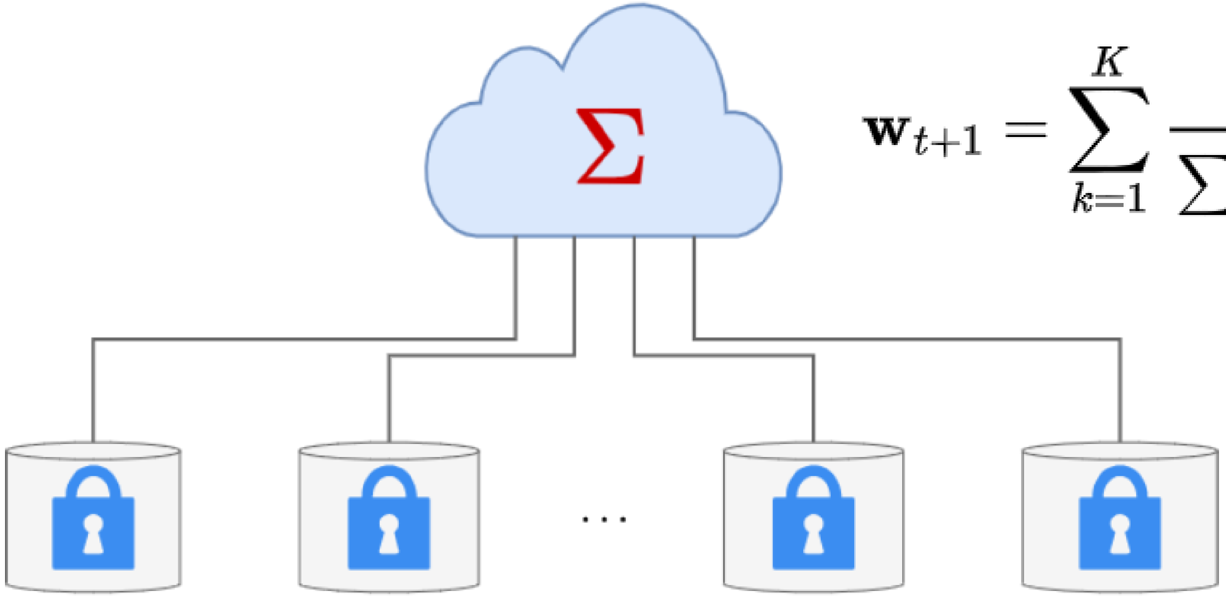
Workflow of FedAvg – Client-Side

- Step 3: Send update to server



Workflow of FedAvg – Server-Side

- Step 4: Aggregate and update global model



$$\mathbf{w}_{t+1} = \sum_{k=1}^{K} \frac{n_k}{\sum_{i=1}^{K} n_i} \mathbf{w}_{t+1}^{k}$$

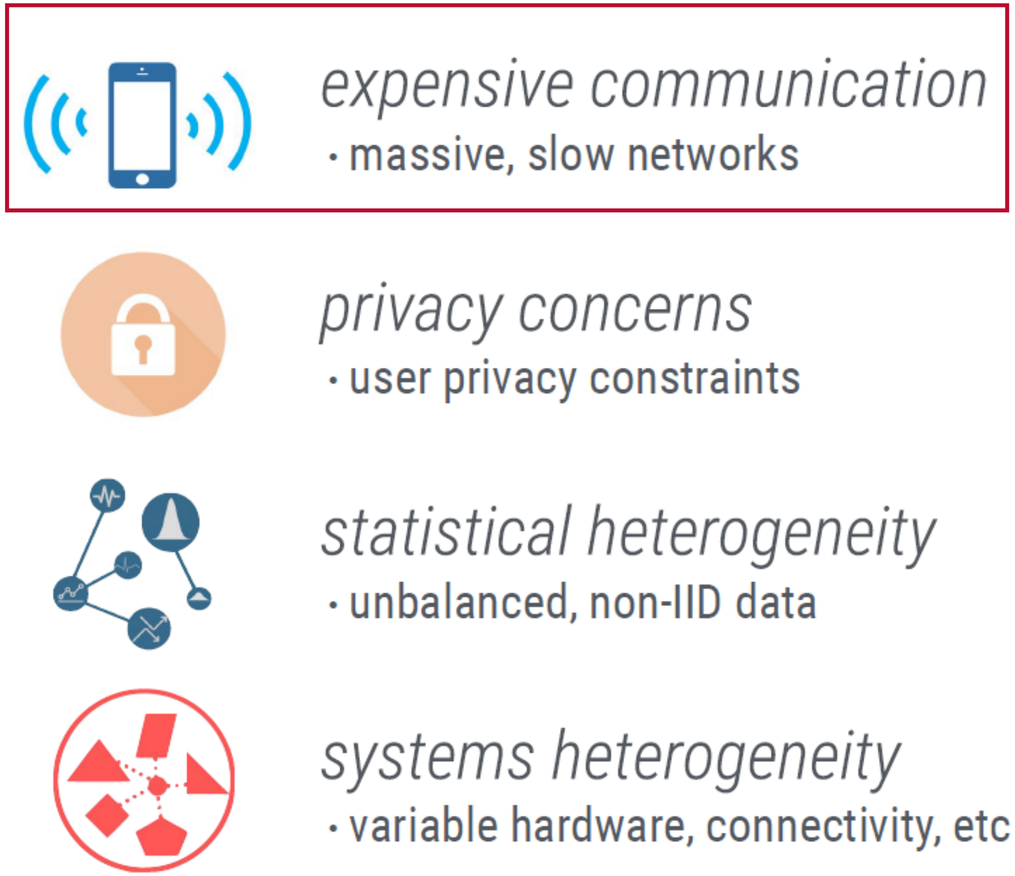
Challenges in FL

- Privacy concerns
  - User privacy constraints
- Communication costs
  - Communication: transmission between server or clients
  - Massive, slow networks
- Data heterogeneity
  - Violation of IID assumption (Non-IID)
- System heterogeneity
  - Variable hardware, network bandwidth, asynchronous Internet connections, etc

Challenges in FL

Can **reduce communication** in FL by:

- Limiting *number of clients* involved in communication
- Reducing number of *communication rounds*
- Reducing *size of messages* sent over network
  - Compression techniques: quantization, sparsification, dropout



Challenges in FL

Keeping **raw data local** to each client is a first step

Can be further improved by adding encryption methods (e.g., differential privacy)

expensive communication
- massive, slow networks

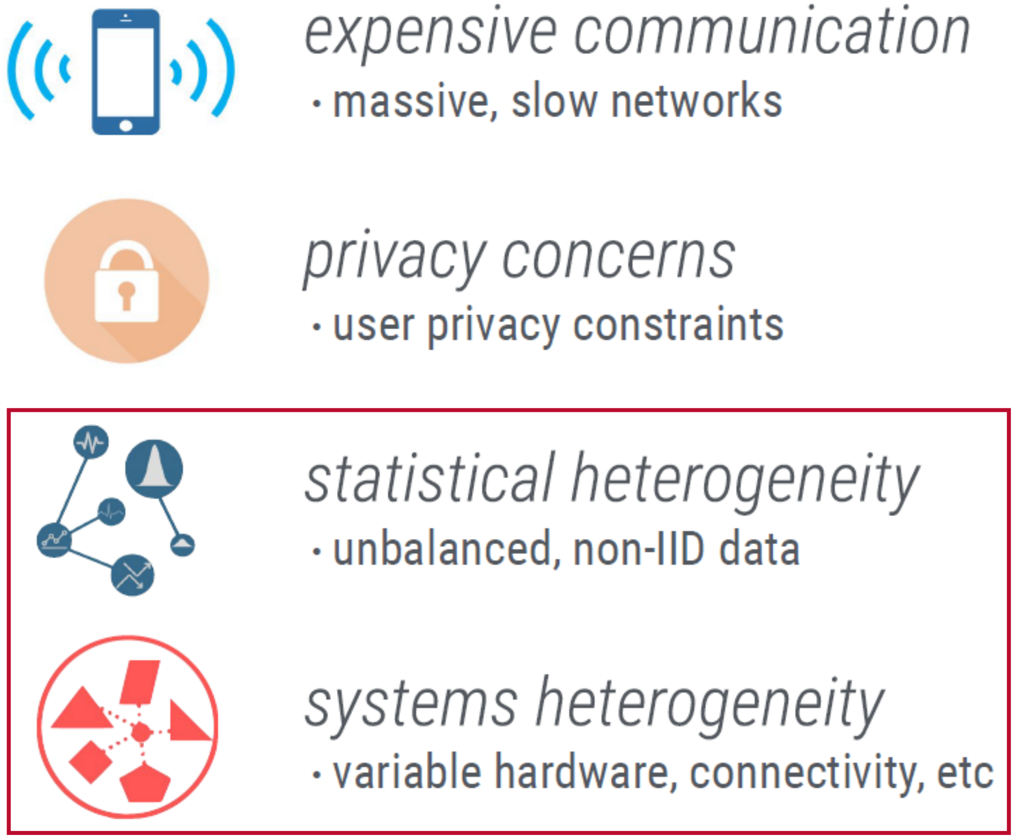
privacy concerns
- user privacy constraints

statistical heterogeneity
- unbalanced, non-IID data

systems heterogeneity
- variable hardware, connectivity, etc

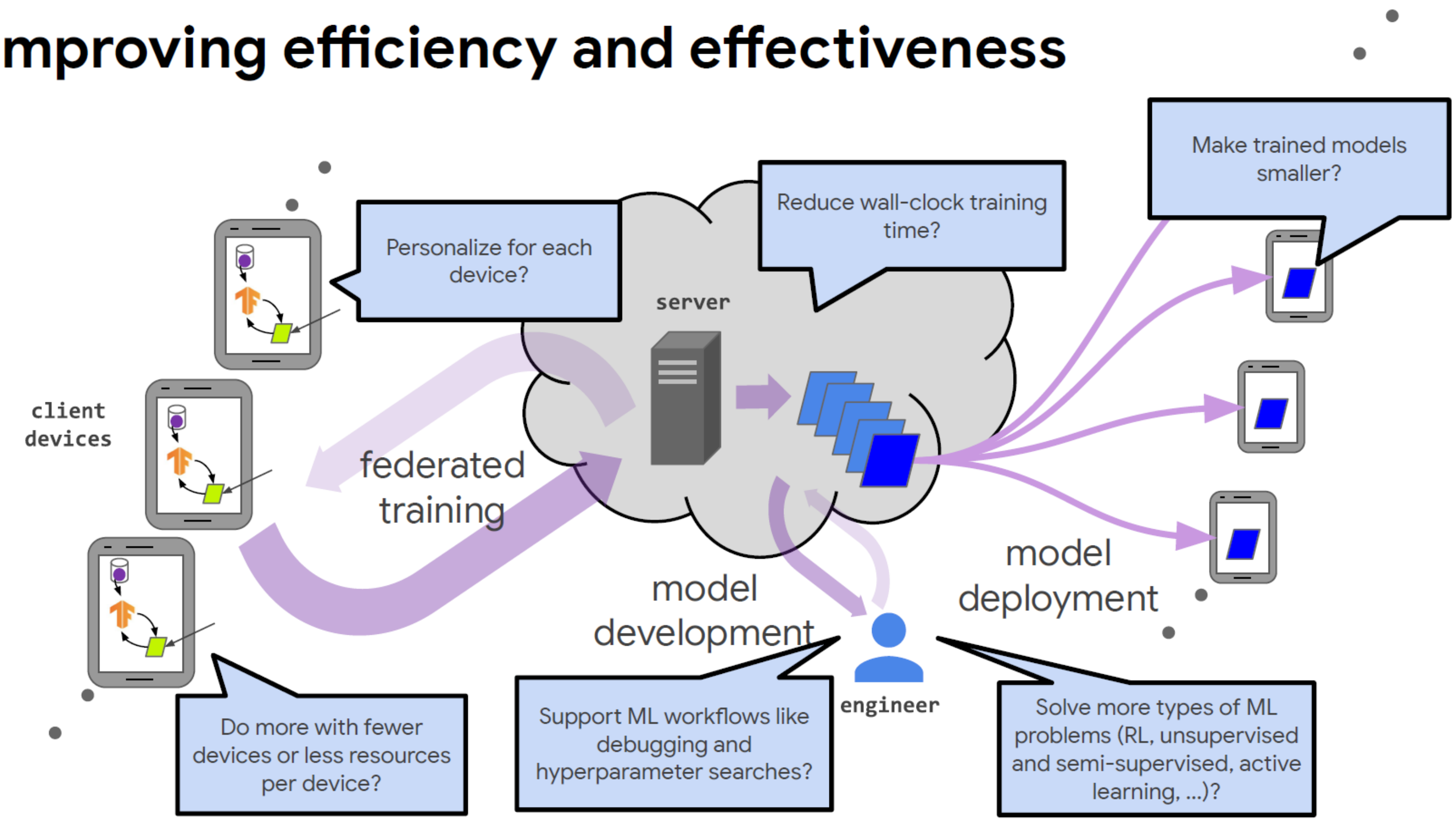
Challenges in FL

Heterogeneous data (e.g., non-IID) and systems (e.g., dropping clients) can bias optimization procedures, and hence degrade the performance of FL

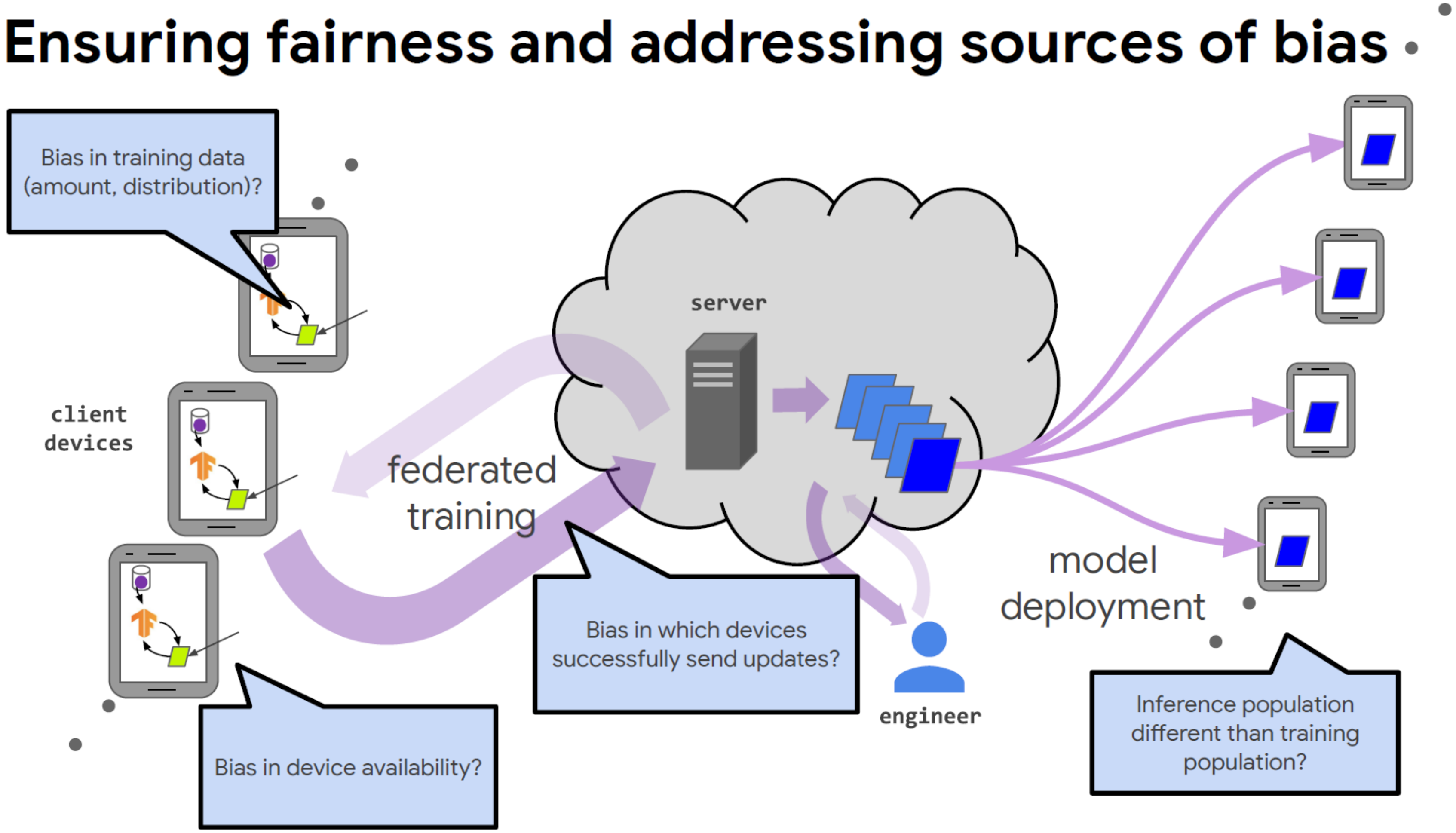


Open Problems

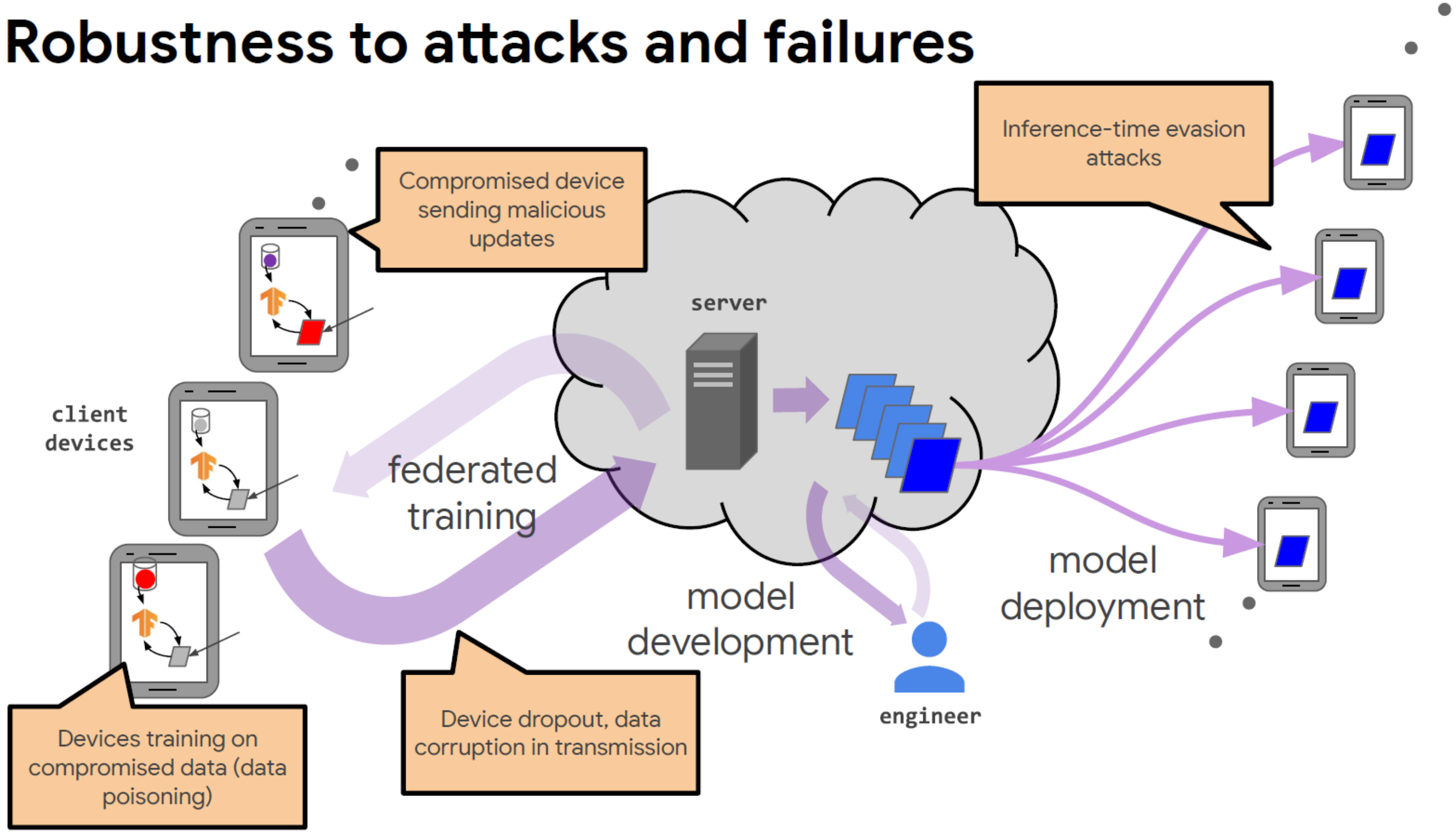
Improving efficiency and effectiveness



Open Problems



Open Problems



The End

Questions?