Learning at the edge

Big data management

Assignment 2 feedback

A complete answer contains:

- an explanation of the approach
- code snippet
- results

Assignment 2 feedback

A complete answer contains:

- an explanation of the approach
- code snippet
- results

If you tried several things that didn't work only show what did work.

Answer questions one by one and make sure that your output matches the question:

- if the task is to find a number, don't output a table (you can show the table as part of the explanation, but it's not enough)

Assignment 2 feedback

Use only functions you fully understand:

- during the random fraud check we might ask what a certain function you used does; if you can't respond we could assume the code wasn't written by you;
- for example functions like repartition can have a big overhead and make your code slower; if you can't argue for its usage, don't use it.

Assignment 2 feedback

Part 1, queries

Query 2: businesses reviewed by 750 users = 750 reviews written by *distinct* users

In some cases the column that you needed to filter on (for example average_stars) was already in the table. Not necessary to recompute.

Some of you have used groupBy when it wasn't necessary.

Assignment 2 feedback

Part 2, authenticity language

Data exploration. What is the difference between lower(text) and .contains(word)?

Hypothesis testing. As much as possible formulate your hypothesis in a mathematical way:

- what does authenticity language mean in your analysis
- what does it mean that there is a stronger relationship between authenticity language and negative words
- try to define as much as possible

Assignment 2 feedback

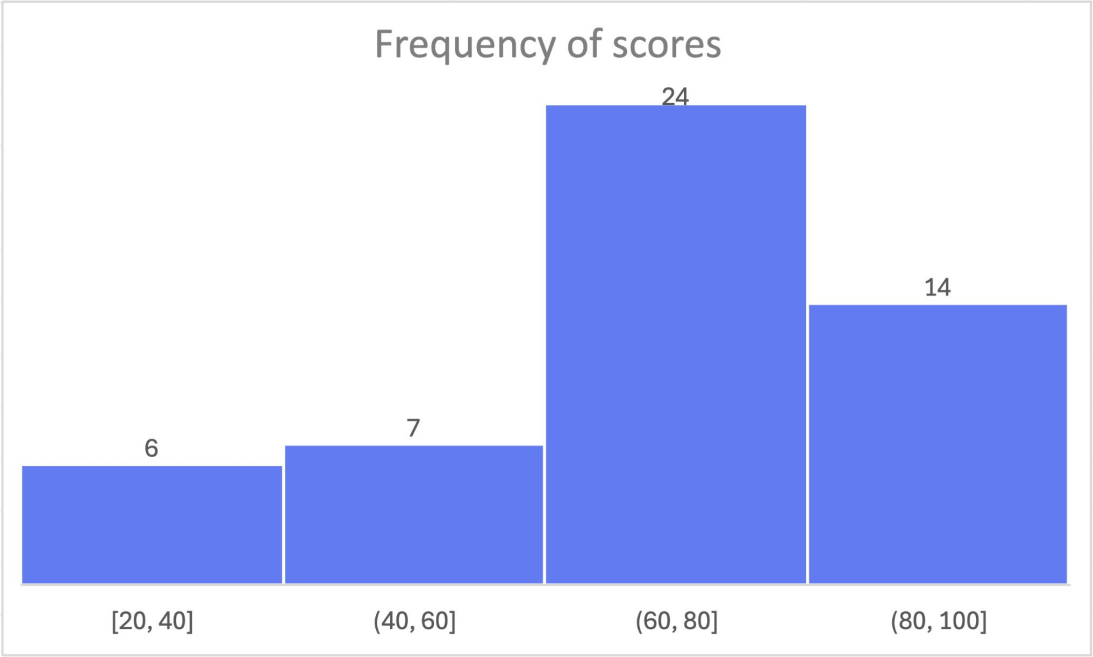
Part 3, prediction model:

While the average ratings are indeed continuous numerical values, the rating for a single review is a discrete number. How can you use that in your model?

Specify:

- why you chose a certain model (classification/regression)?
- what are the features; if you use a tokenizer, hasher, briefly explain what they do
- show the results clearly

Assignment 2 feedback



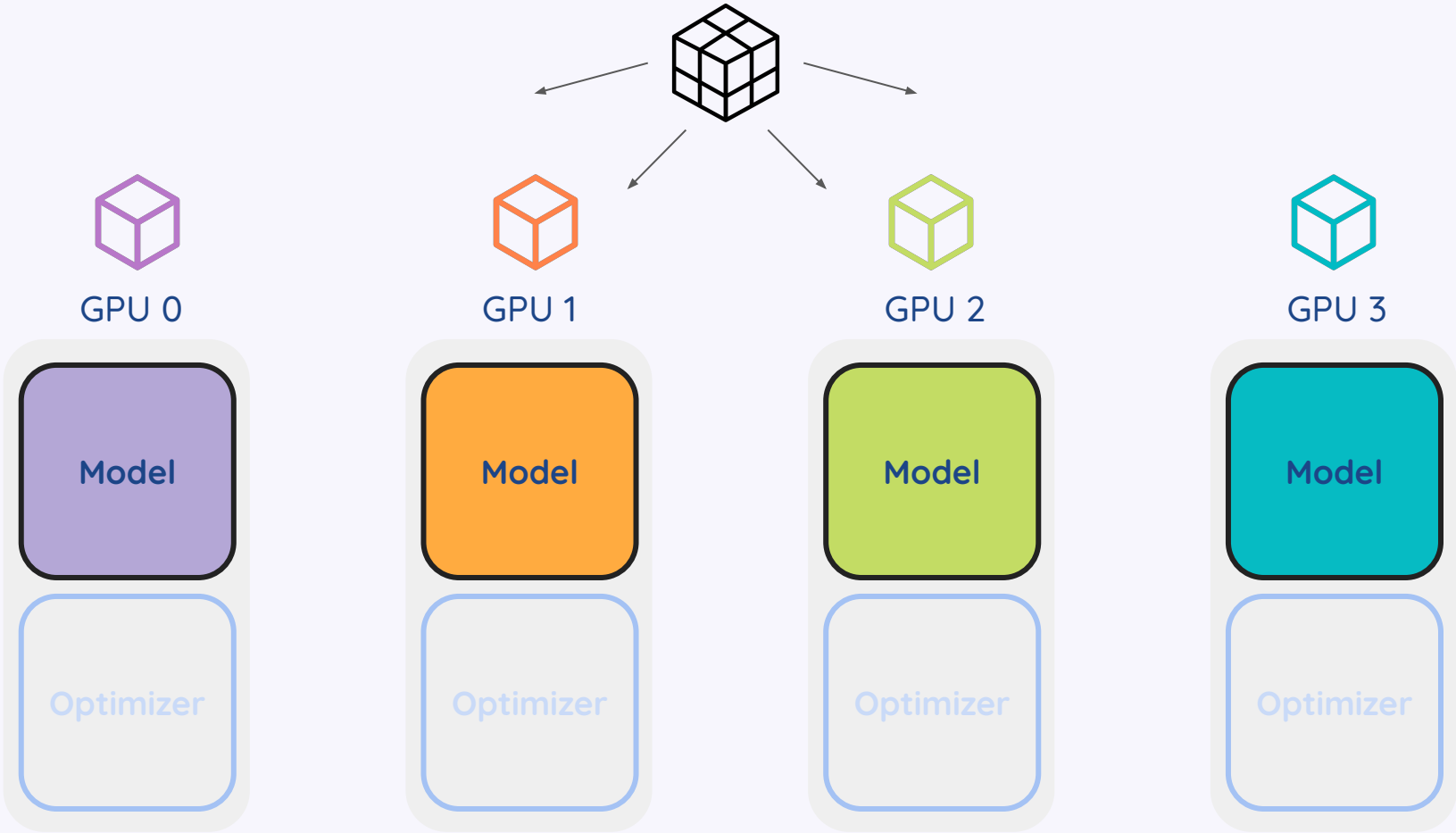
Incomplete

Needs some improvement

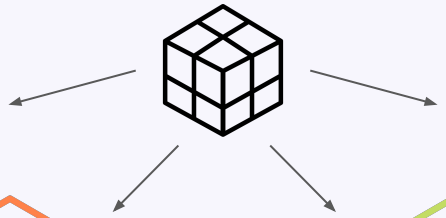
Good

Excellent

RECAP: Training on multiple GPUs - data parallelism



Training on multiple GPUs - data parallelism



each GPU:

- forward pass on its data batch
- compute loss
- backward pass to calculate gradients

GPU 0

Model

Optimizer

GPU 1

Model

Optimizer

GPU 2

Model

Optimizer

GPU 3

Model

Optimizer

Gradient descent

loss

parameters regularizer

$$\min_w \sum_{i \in \text{data}} l_i(w) + R(w)$$

Cost function J

Gradient Descent (GD)

$$w^{k+1} = w^k - \alpha_k \nabla J(w^k)$$

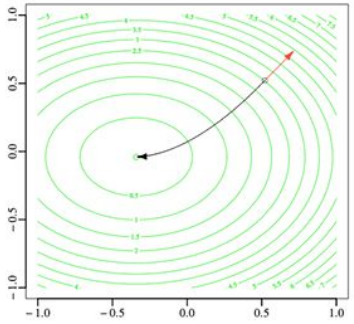
learning rate gradient

Batch: $\sum_{i=0}^{n}$

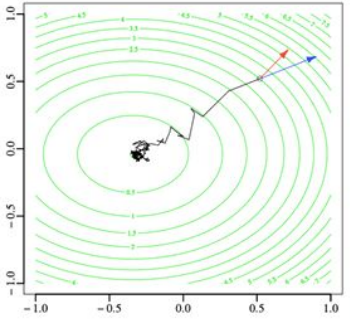
Stochastic: Random i

Mini-batch: $\sum_{i=0}^{m}$ $m \ll n$

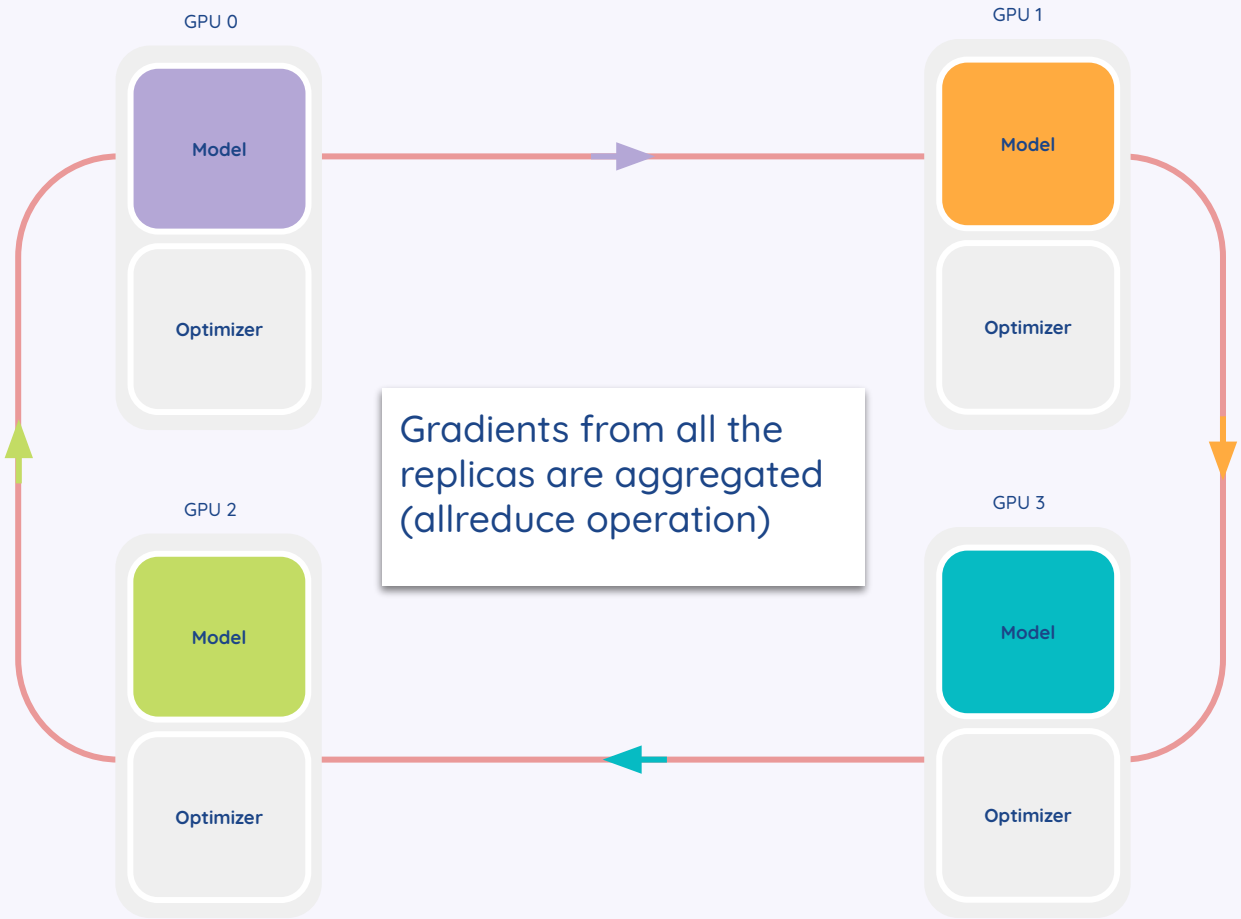
Batch



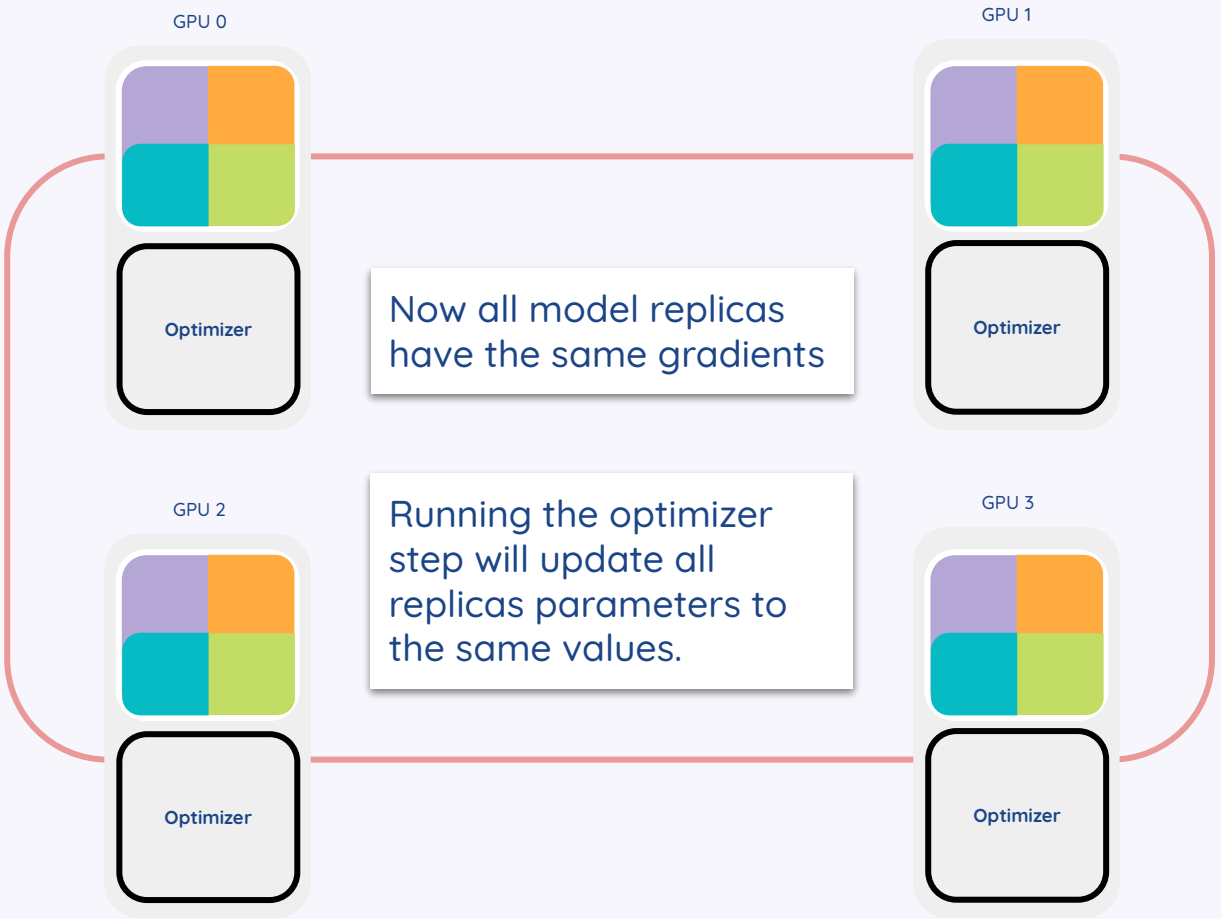
Stochastic



Gradients aggregation



Synchronized replicas



Fully sharded data parallelism

Data parallelism:

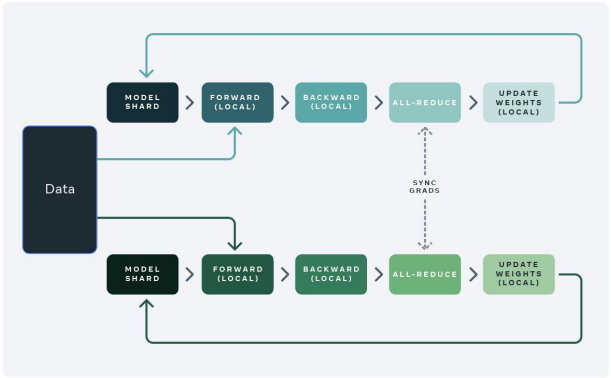
- each processor/worker had a replica of the model (model parameters, gradients, and optimizer states)

Fully sharded data parallelism:

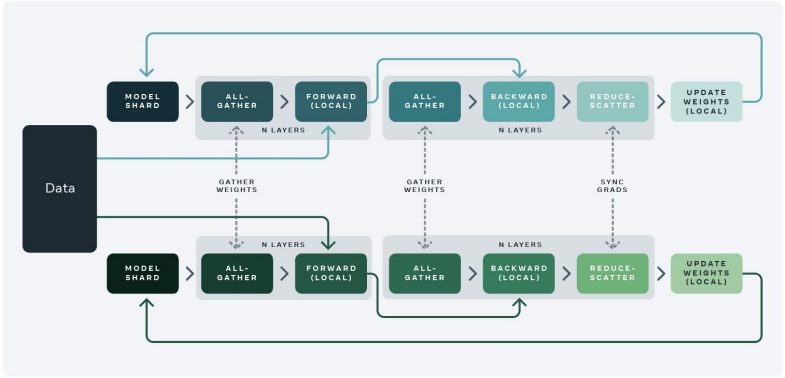
- model parameters, optimizer states and gradients across are distributed across workers
- ⇒ smaller GPU memory footprint, possible to train very large models
- but increased communication costs

Fully sharded data parallelism

Standard data parallel training



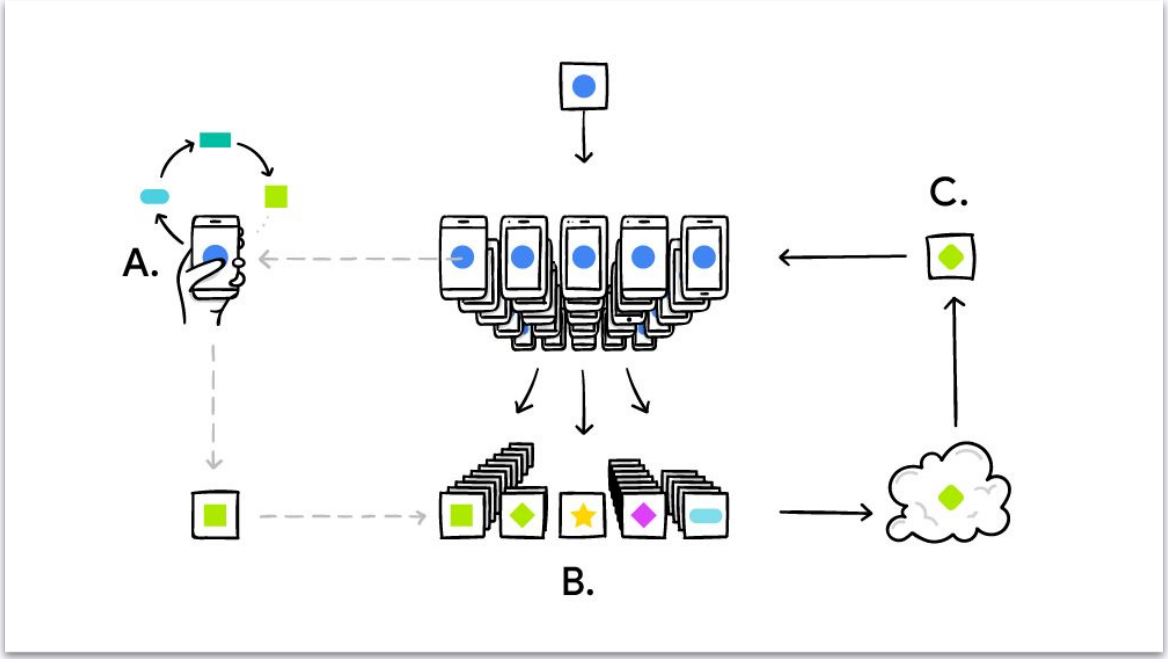
Fully sharded data parallel training



Check this article for more details.

This lecture

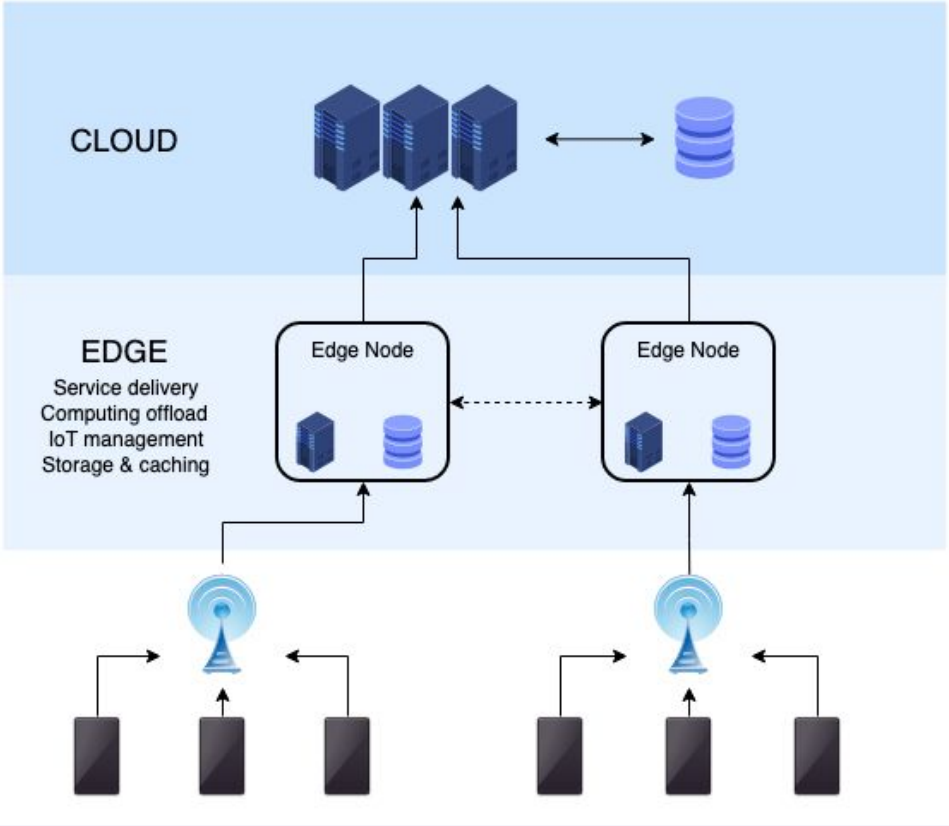
Edge computing and federated learning



	Distributed Computing	Parallel Computing
Definition	Network of nodes working together to perform a task.	Multiple processors or cores within a single computer working on different parts of a task simultaneously.
Architecture	Each node has its own private memory and may operate autonomously.	Processors may share memory (shared memory parallelism) or use separate memory with communication via message passing.
Objective	Handle large-scale, long-term projects across geographically dispersed machines.	Speed up computing tasks by dividing them into smaller parts to solve at the same time.
Scalability	Highly scalable by adding more machines.	Limited by the number of processors in a system.
Fault Tolerance	More fault-tolerant, system can often continue functioning if one node fails.	Less fault-tolerant as all processors share a single physical machine.
Examples	Apache Spark, Hadoop	Supercomputers, GPUs, Multi-core CPUs

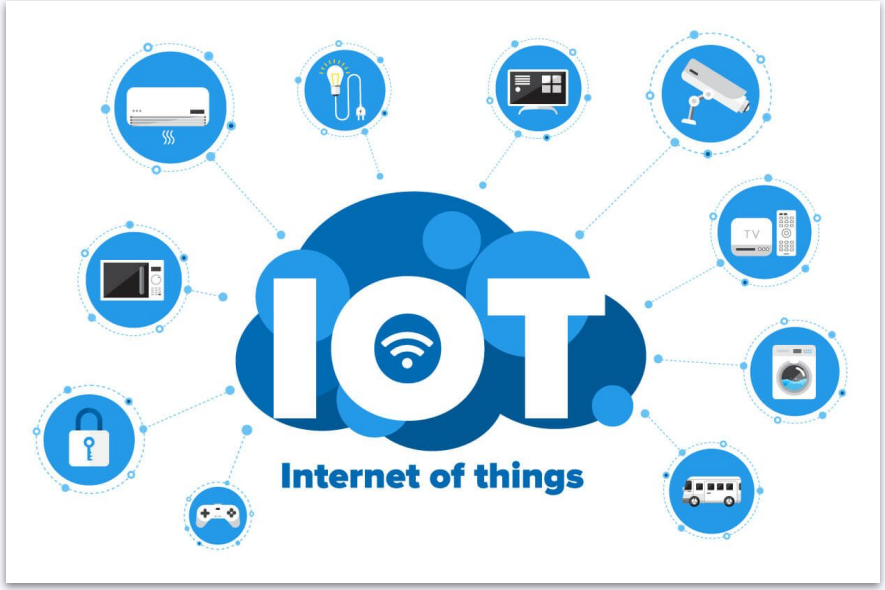
Edge computing

Edge computing is a distributed computing paradigm that brings computation and data storage closer to the sources of data.

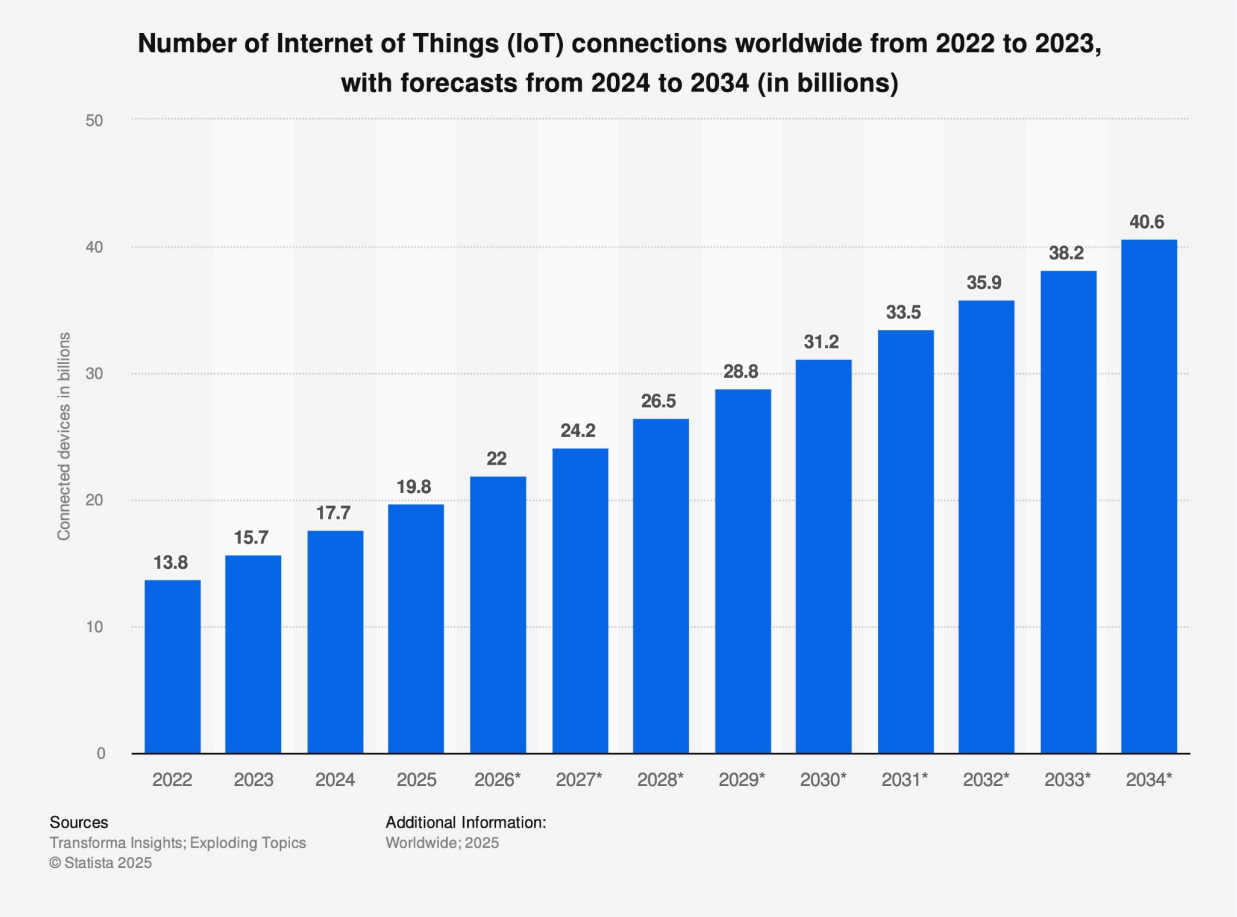


Internet of Things (IoT)

Physical objects (or groups of such objects) with sensors, processing ability, software and other technologies that connect and exchange data with other devices and systems over the Internet or other communications networks.



IoT devices


Number of Internet of Things (IoT) connections worldwide from 2022 to 2023, with forecasts from 2024 to 2034 (in billions)
Connected devices in billions
50
40
30
20
10
0
13.8
15.7
17.7
19.8
22
24.2
26.5
28.8
31.2
33.5
35.9
38.2
40.6
2022
2023
2024
2025
2026*
2027*
2028*
2029*
2030*
2031*
2032*
2033*
2034*
Sources
Transforma Insights; Exploding Topics
© Statista 2025
Additional Information:
Worldwide; 2025

Edge computing

Raw data may not need to be sent through a network

- models can work where Internet connections are unreliable = offline availability
- network latency not a problem = improved response time
- sensitive data stays on device = data security

But:

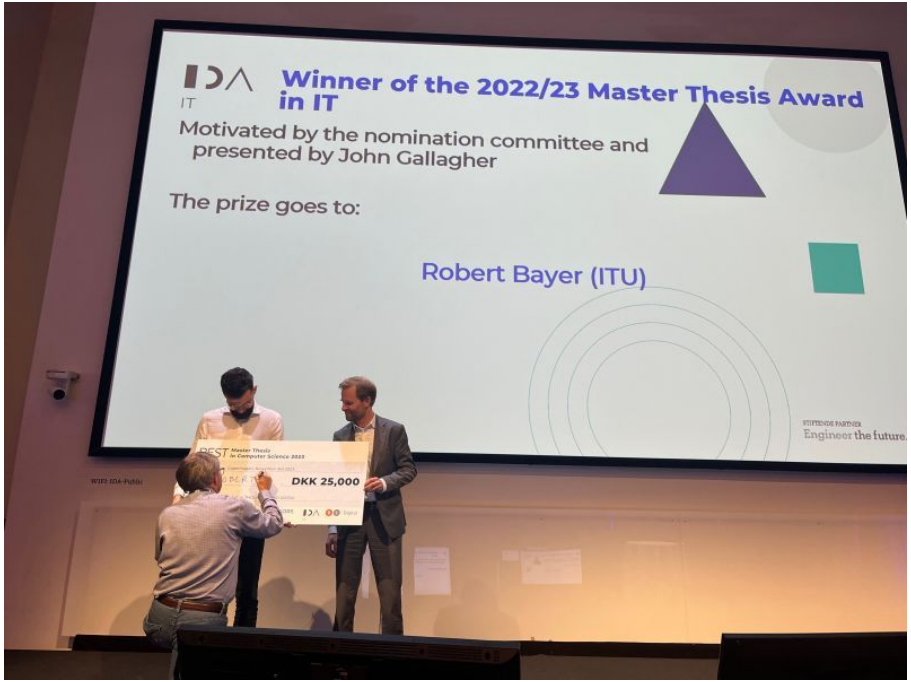
Devices need to be powerful enough to handle the computation

Reaching the Edge of the Edge: Image Analysis in Space

Robert Bayer
roba@itu.dk
IT University of Copenhagen
Copenhagen, Denmark

Julian Priest
jucp@itu.dk
IT University of Copenhagen
Copenhagen, Denmark

Pınar Tözün
pito@itu.dk
IT University of Copenhagen
Copenhagen, Denmark



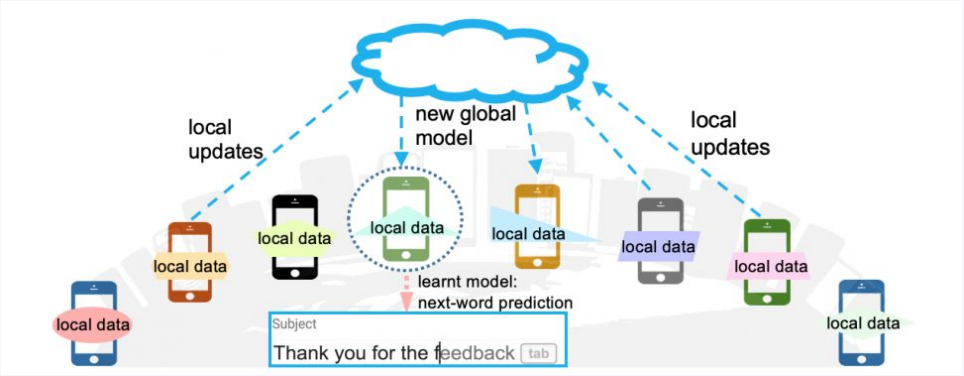
Students launch a satellite to test artificial intelligence in space

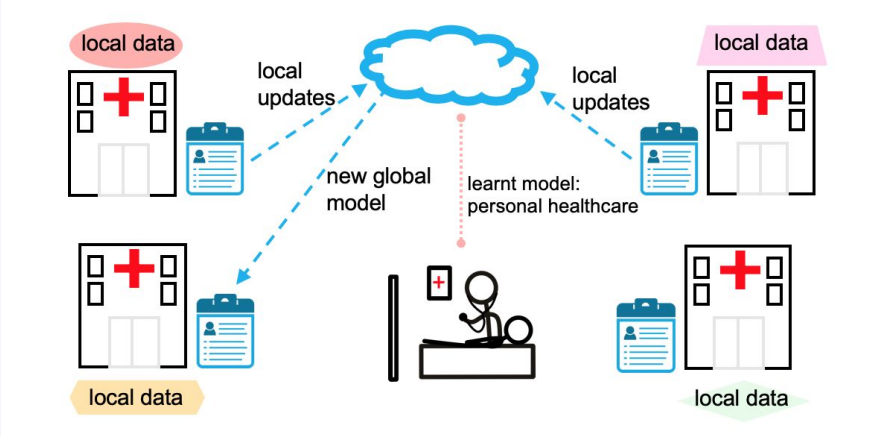
On April 14, students from ITU will contribute to writing space history. The satellite, DISCO-1, is launched into space and it carries a microcomputer to test artificial intelligence outside the atmosphere. The satellite is developed by the space program, DISCO, which is a collaboration between students from four Danish universities.

Federated learning

Distributed approach where multiple parties collaboratively train a model to solve a common machine learning task, without exposing data.

- Mitigate privacy risks
- Reduce communication and training costs





Federated learning

- Federated learning settings, challenges and workflow
- Federated averaging algorithm
- Privacy
- Attacks
- Fairness and bias

Federated learning settings

	Distributed learning	Cross-silo federated learning	Cross-device federated learning
Participants	“Flat” dataset, computation split between several nodes.	Different organizations (medical, financial etc.)	Large number of devices.
Data distribution	Data is centrally stored.	Data is generated and stored locally. Not communicated to other participants.	
Data availability	All participants always available.		A fraction of participants available at one time.
Scale	1-1000 participants.	2-100 participants.	up to 10^{10} participants.
Participant reliability	Few failures.		Highly unreliable.

Challenges

- Non-independent and identically distributed (non-IID) data
  - observations are not independent of each other and/or the distribution of the observations can change

Challenges

- Non-independent and identically distributed (non-IID) data
  - observations are not independent of each other and/or the distribution of the observations can change
- Communication constraints
  - higher-latency, lower-throughput connections

Challenges

- Non-independent and identically distributed (non-IID) data
  - observations are not independent of each other and/or the distribution of the observations can change
- Communication constraints
  - higher-latency, lower-throughput connections
- Harder to identify unwanted biases
  - without access to the data

Challenges

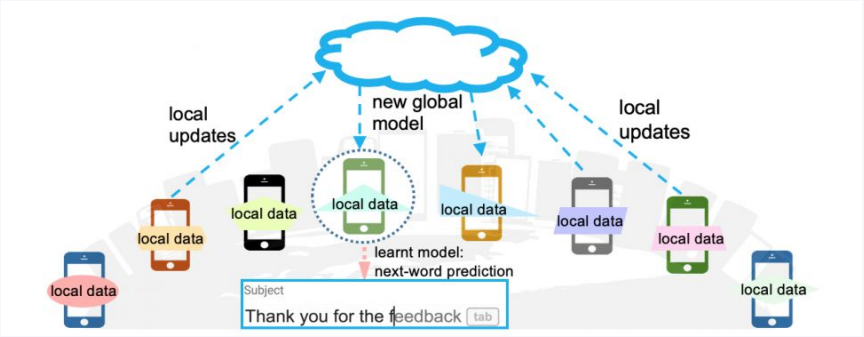
- Non-independent and identically distributed (non-IID) data
  - observations are not independent of each other and/or the distribution of the observations can change
- Communication constraints
  - higher-latency, lower-throughput connections
- Harder to identify unwanted biases
  - without access to the data
- Vulnerable to model-poisoning attacks
  - due to sending model information

Challenges

- Non-independent and identically distributed (non-IID) data
  - observations are not independent of each other and/or the distribution of the observations can change
- Communication constraints
  - higher-latency, lower-throughput connections
- Harder to identify unwanted biases
  - without access to the data
- Vulnerable to model-poisoning attacks
  - due to sending model information
- Difficult to prevent attacks by detecting anomalies
  - without access to the data

Federated learning workflow

1. Participant selection
   - eligibility requirements
2. Broadcast
   - global model sent to participants
3. Local computation
   - compute update based on local data
4. Aggregation
   - a server collects an aggregate of updates
5. Model update
   - a server updates the global model

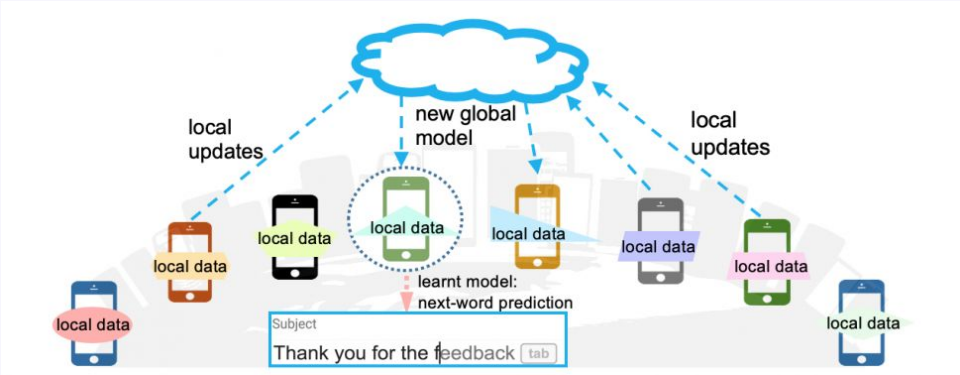


Communication-Efficient Learning of Deep Networks from Decentralized Data

H. Brendan McMahan **Eider Moore** **Daniel Ramage** **Seth Hampson** **Blaise Agüera y Arcas**
Google, Inc., 651 N 34th St., Seattle, WA 98103 USA

Federated Stochastic Gradient Descent

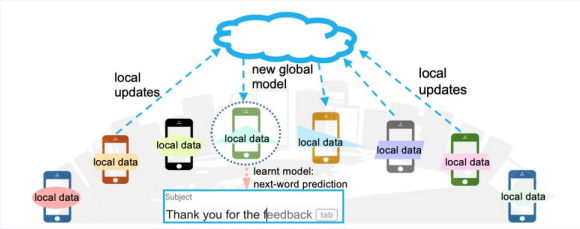
- **FedSGD** - SGD in a federated learning setting
- Select a fraction of participants to take a step of gradient descent.
- The server takes a weighted average of the gradients.



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

Federated Stochastic Gradient Descent

- **FedSGD** - SGD in a federated learning setting
- Select a fraction of participants to take a step of gradient descent.
- The server takes a weighted average of the gradients.
- **FedAvg**: Each participant iterates the update multiple times.
- 3 key parameters:
  - C = fraction of participants
  - E = number of training passes on a local dataset
  - B = local minibatch size (B = inf -> full local dataset)



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

Federated Averaging

3 key parameters:

- C = fraction of participants
- E = number of training passes on a local dataset
- B = local minibatch size

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

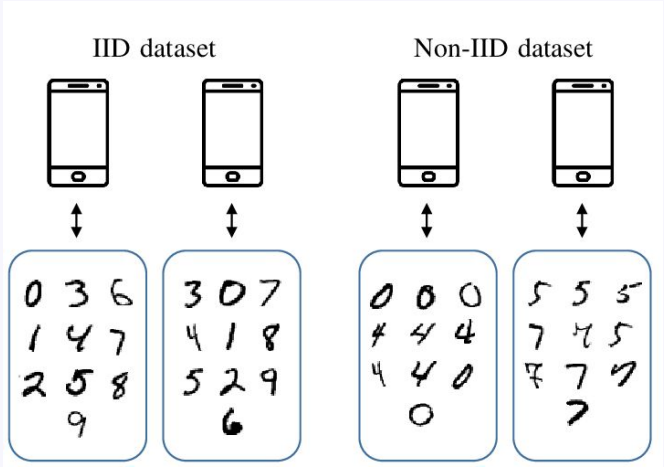
FedAvg - experimental set-up

- Image classification and language modeling tasks
- Digit recognition on *MNIST* dataset
  - IID setting: data is shuffled and partitioned into 100 clients (600 samples each)
  - Non-IID: sort the data by digit label, divide into 200 partitions of 300 and assign 2 partitions to 100 clients each

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

FedAvg - experimental set-up

- Image classification and language modeling tasks
- Digit recognition on *MNIST* dataset
  - IID setting: data is shuffled and partitioned into 100 clients (600 samples each)
  - Non-IID: sort the data by digit label, divide into 200 partitions of 300 and assign 2 partitions to 100 clients each



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

FedAvg - experimental set-up

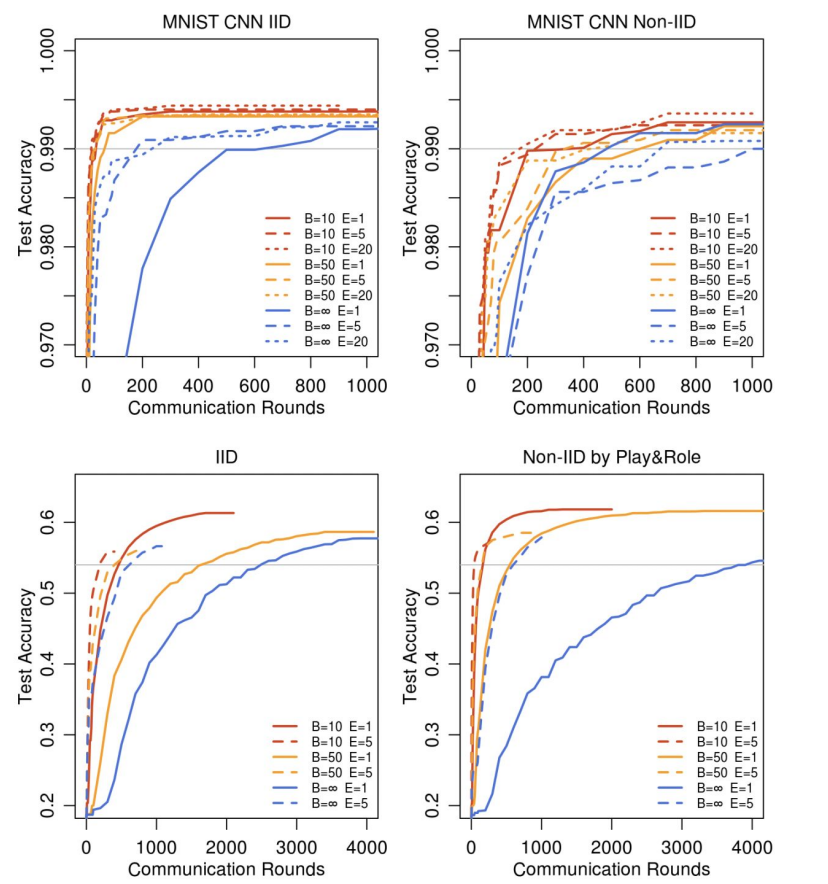
- Image classification and language modeling tasks
- Digit recognition on *MNIST* dataset
  - IID setting: data is shuffled and partitioned into 100 clients (600 samples each)
  - Non-IID: sort the data by digit label, divide into 200 partitions of 300 and assign 2 partitions to 100 clients each
- Language modeling on *The Complete Works of William Shakespeare*
  - 1146 participants
  - for each participant, split the data into set of training lines and set of test lines
  - next character detection

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

FedAvg - experimental results

- Adding more local SGD updates per round can produce a dramatic decrease in communication costs

- Shakespeare task: learning on the non-IID and unbalanced data is actually much easier; some roles have relatively large local datasets



Variations of the FL framework

- Fully decentralized
- Data partitioning by features
- Split learning

Fully decentralized

- Server - point of failure, bottleneck ⇒ replace communication to server by peer-to-peer
- Topology = connected graph
  - sparse, with small max degree
- One round = each participant makes a local update and exchanges info with neighbors
  - for example in SGD - one gradient step + averaging one's local parameters with neighbors
- No more global state
  - all local models should converge to a desired global solution

Split learning

- Model itself is executed on different devices, in a per-layer fashion
- A participant computes a pass through a network up to a specific layer
- The reminder of the computation is done by another participant or the server
- Potential leakage via communicated activations

Properties of data in the FL setting

- Non-identical participant distributions:
  - **Feature distribution skew** (covariate shift) - local data has different statistical distributions across participants
  - **Label distribution skew** (prior probability shift) - labels have different statistical distributions across participants
  - **Concept drift** - same labels correspond to different features
  - **Concept shift** - same features correspond to different labels
  - **Quantity skew** (unbalancedness) - different quantities

Properties of data in the FL setting

- Non-identical participant distributions:
  - **Feature distribution skew** (covariate shift) - local data has different statistical distributions across participants
  - **Label distribution skew** (prior probability shift) - labels have different statistical distributions across participants
  - **Concept drift** - same labels correspond to different features
  - **Concept shift** - same features correspond to different labels
  - **Quantity skew** (unbalancedness) - different quantities
- Violations of independence - temporal/geographical patterns bias in the source of data
- Dataset shift - training participants might be different than deployment participants

Global FL vs local models

- Small and IID datasets - global models have higher accuracy
- Pathologically non-IID - local models are better
- Global model particularly useful to
    - Assign a model to users with no data
    - Validation before deployment

Privacy

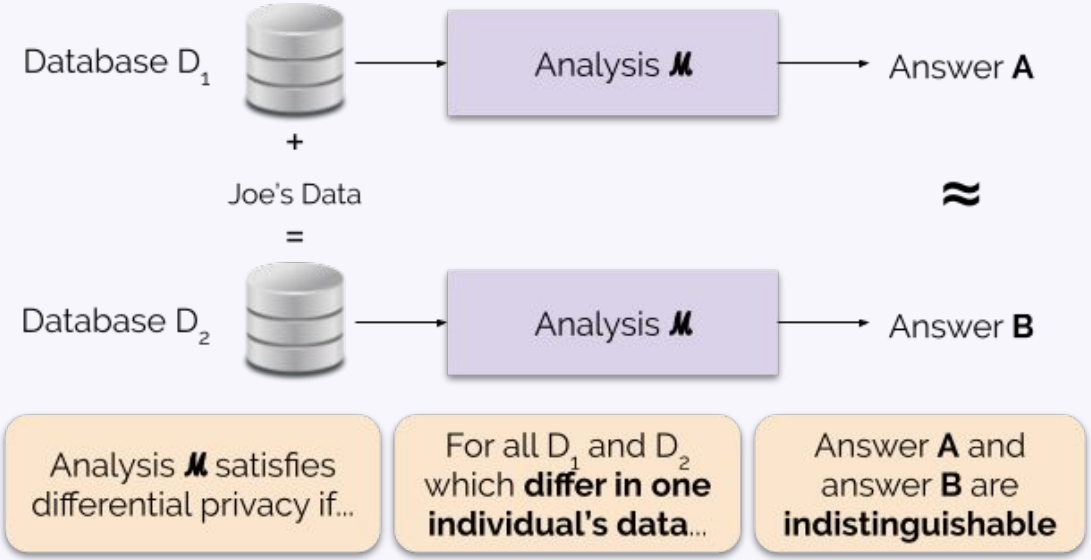
- FL provides a level of privacy
- But no formal guarantee that no information about raw data is being leaked
    - For example: knowing a previous model for a user and a current update could allow to infer training examples from that user
- Privacy threats are present at different levels:
    - participating parties, coordinating server, engineers & analysts who have access to output from the system

Privacy preserving technologies

- Secure multi-party computation
  - secure computation where a fraction of participants collaborate to compute an agreed-function of their private inputs
- Homomorphic encryption
  - some operations can be performed without decrypting the input
- Differential privacy
  - quantifies how much can be learned about an individual based on the output of an algorithm

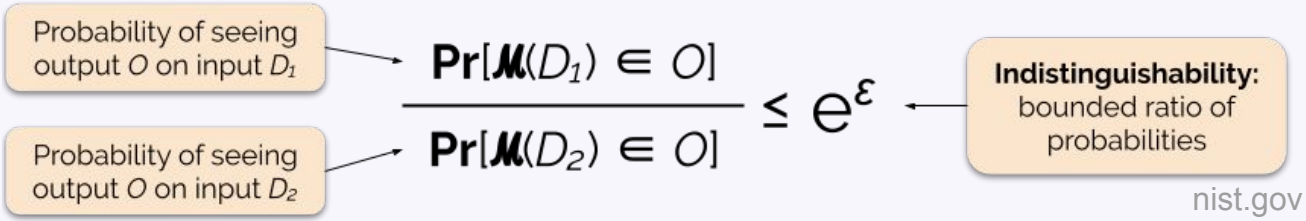
Differential privacy

- State of the art for limiting information sharing about participants.
- Introduces enough uncertainty to mask any given individual.



nist.gov

Differential privacy



- Analysis applied on adjacent datasets should give similar output
  - similarity measured as the ratio between the probabilities of seeing output O when applying analysis *M* to the datasets
- Epsilon is the measure of privacy loss
- This applies for any pair of adjacent datasets

Local differential privacy

Each participant applies a transformation to their data - perturbing with a randomized parameter

- LDP has been deployed effectively to gather statistics on popular items across large user bases by Google, Apple and Microsoft
- has been used in federated settings for spam classifier training
- LDP deployments all involve large numbers of clients and reports (can be up to a billion)
- achieving LDP while maintaining utility is difficult

Fairness and privacy

- The notions of fairness and privacy seem to be in tension.
- Differentially-private learning can have disparate impacts on sensitive groups.
- A possible solution:
  - personalization - more local training

Federated learning in a nutshell

- Federated learning aims to improve learning tasks by:
  - using data from multiple participants
- Privacy guarantees need to be in place; usually done by:
  - sharing model parameters instead of the data
  - aggregation of parameters
  - secure multi party computation, homomorphic computing, trusted execution environments
  - differential privacy
- Different from distributed learning because:
  - data across participants is non IID
  - communication constraints
  - failure of nodes more probable
  - more difficult to identify anomalies or unwanted biases
  - susceptible to model attacks (which could have higher impact than data poisoning)
- So far, most work focused on in using/adapting distributed learning algorithms:
  - aggregation of model parameters
  - some privacy guarantees
  - showing convergence in the case of node failures

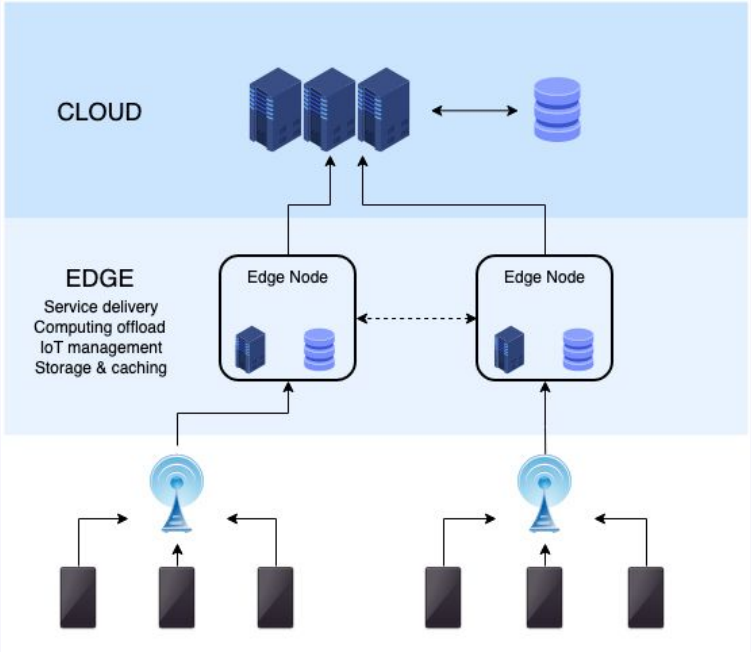
Research questions in federated learning

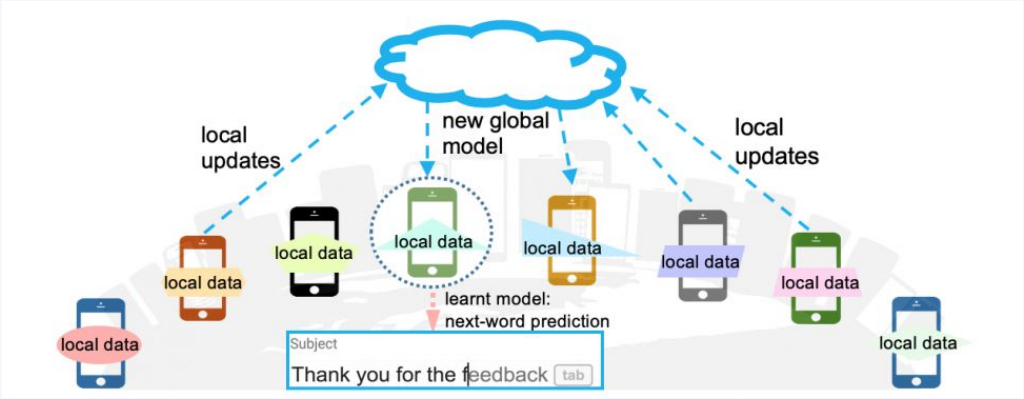
- Convergence guarantees analysis
  - aggregation methods
  - asynchronous setting
  - non IID case
- Trade off between local updates and global updates in terms of:
  - efficiency (speed of convergence)
  - global accuracy vs local accuracy
- Analysis of leakage from the communicated parameters
- Trade off between privacy, accuracy, bias

This lecture

What is edge computing? What are the advantages and challenges?

What is federated learning? Advances and open problems





Next lectures

Rodrigo Nunes Laigner:

Data Management in Cloud Applications: The Good, The Bad, and The Ugly

Henrik Norhøj Nielsen:

Introduction to Snowflake: Managing data at scale