

INTRODUCTION TO FEDERATED LEARNING

Webinar

13 October 2026, 14:00–15:00 CEST

Speaker: Sarah Reisenbauer, AIT Center for Energy



AGENDA

- Welcome & presentation of AI Factory Austria AI:AT (5 min)
- Introduction and fundamentals of Federated Learning
- The Federated Learning workflow
- Important aspects
- Example use cases
- Federated Learning frameworks
- Questions & discussion (10 min)

ABOUT THE SPEAKER

Reisenbauer Sarah

Research Engineer Data Analytics
and Machine Learning

Power System Digitalisation
Power and Renewable Gas
Systems

Center for Energy



- Since 2021: research engineer and project coordinator at AIT, focusing on electric power system digitalization through AI
- Main activities: data analytics, distribution grid state estimation, forecasting of electricity generation and load

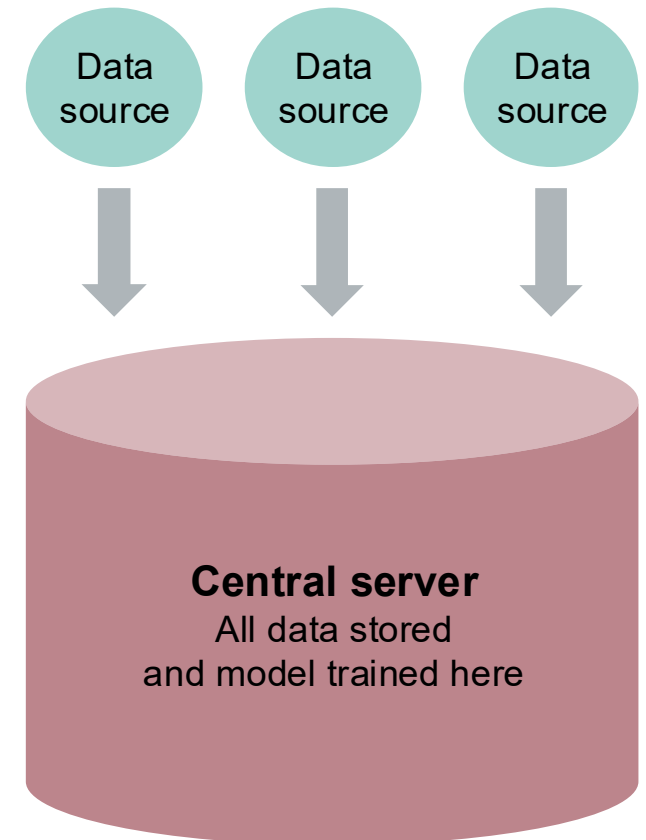
INTRODUCTION AND FUNDAMENTALS

From centralized to distributed machine learning



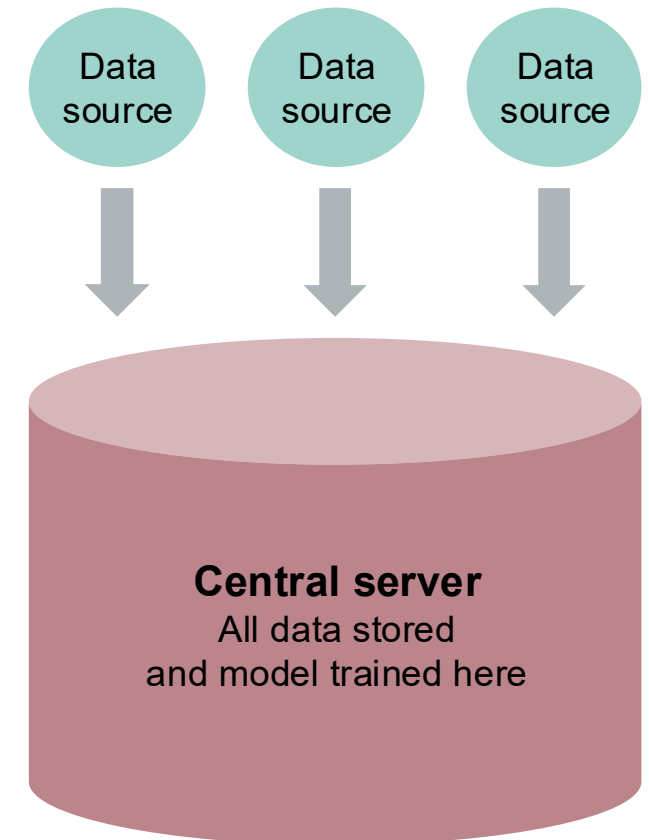
INTRODUCTION AND FUNDAMENTALS

- **Recap centralized ML, standard ML**
 - All data is **collected on one central server / computer** and the model is trained there
 - Usually on the whole available / suitable dataset



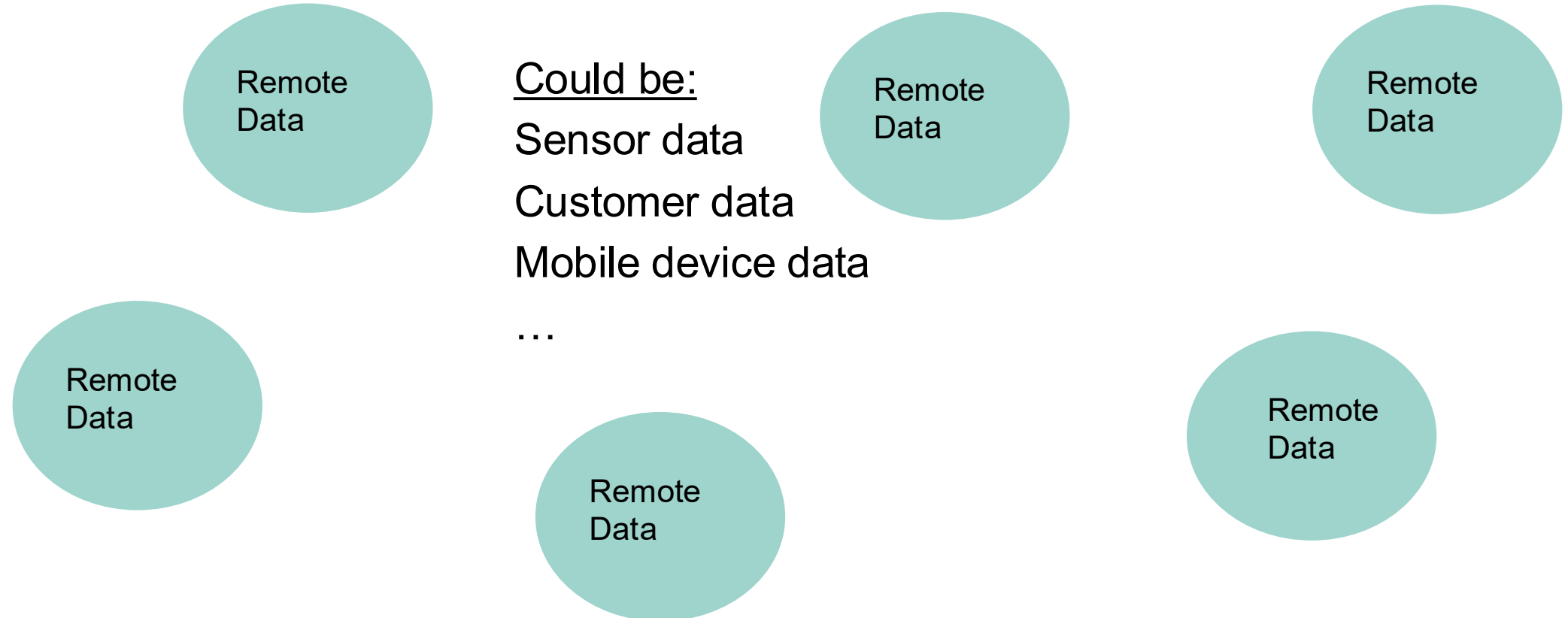
INTRODUCTION AND FUNDAMENTALS

- **Recap centralized ML, standard ML**
 - All data is **collected on one central server / computer** and the model is trained there
 - Usually on the whole available / suitable dataset
- **History of FL**
 - **2016:** McMahan et al. (Google) coin the term "federated learning"
 - At the time misuse of personal data by pages / apps was discussed and shortly after EU GDPR was passed
 - **2017:** Google presents FL publicly and deploys it in the Gboard keyboard on Android phones for word suggestions



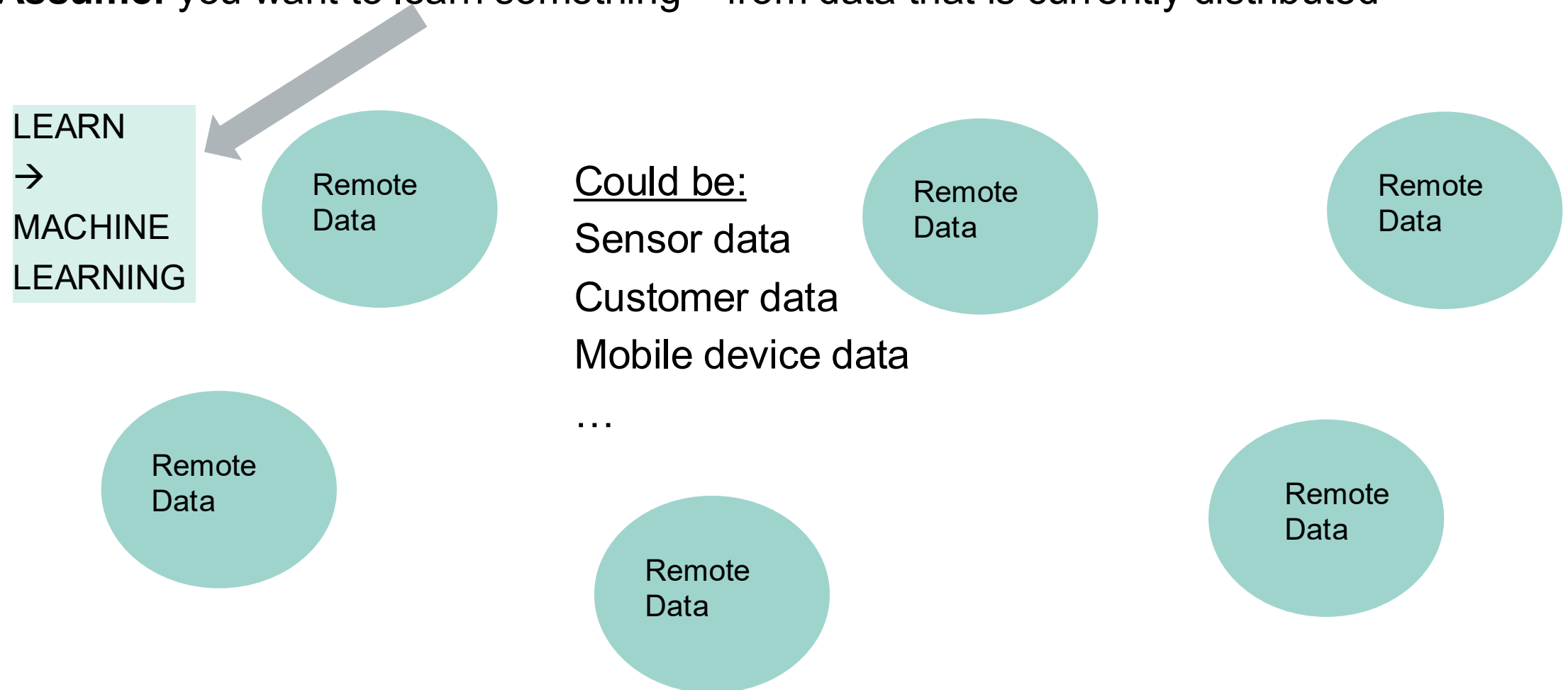
INTRODUCTION AND FUNDAMENTALS

Assume: you want to learn something – from data that is currently distributed



INTRODUCTION AND FUNDAMENTALS

Assume: you want to learn something – from data that is currently distributed

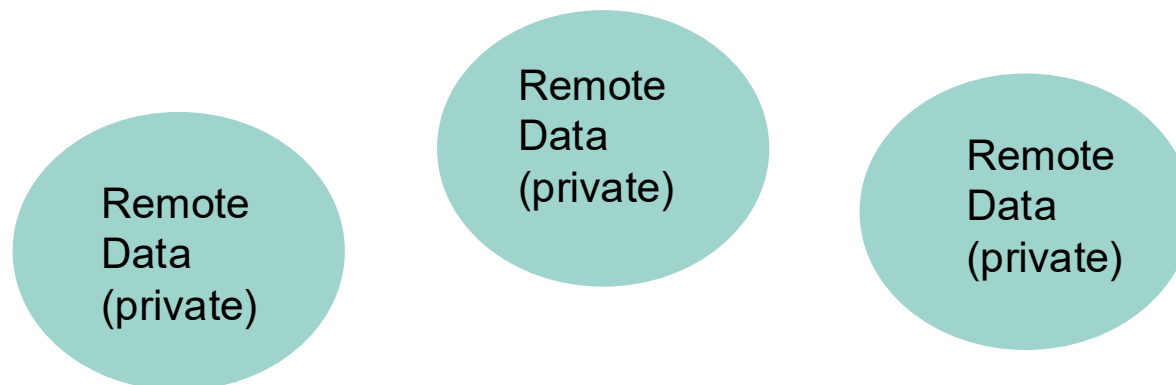


INTRODUCTION AND FUNDAMENTALS

Assume: you want to learn something – from data that is currently distributed

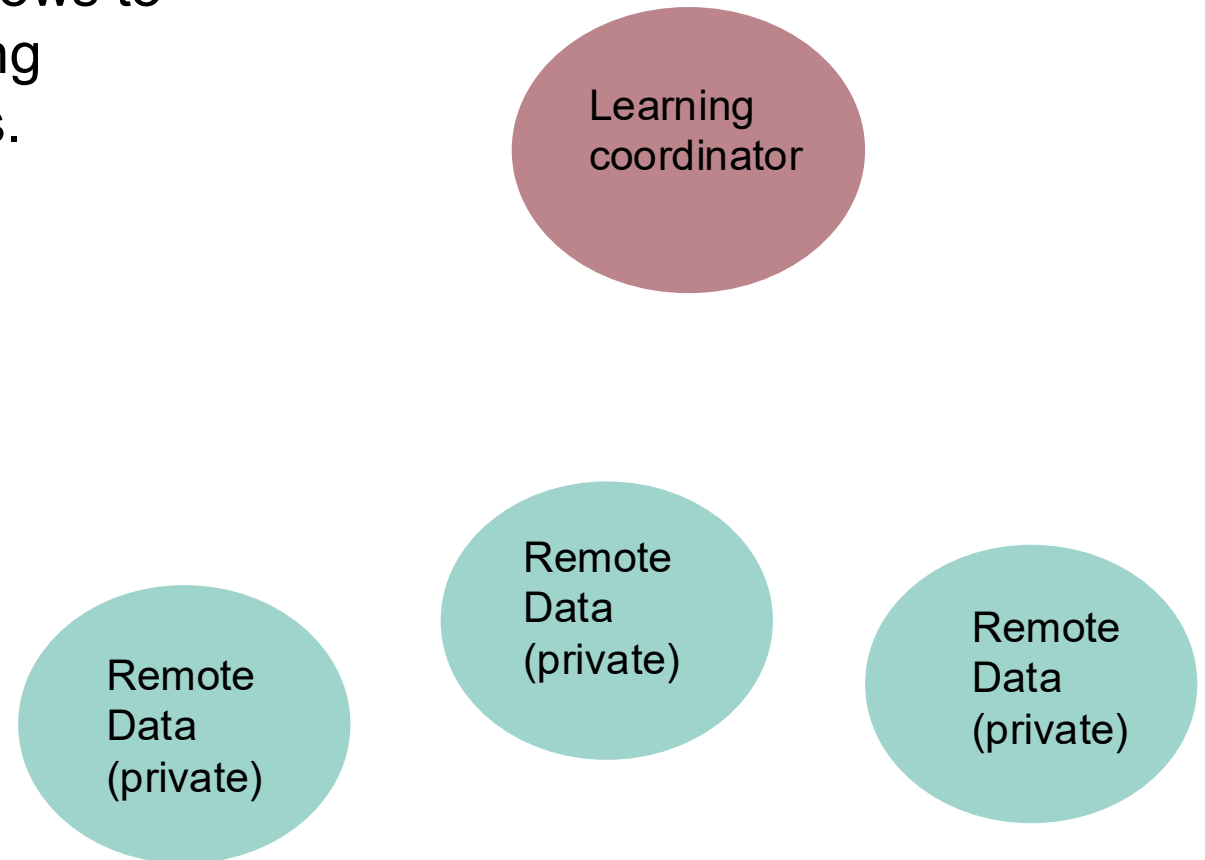
Assume: the data cannot be transmitted to a central location, e.g. due to

- GDPR reasons, privacy
- security reasons
- bandwidth, transfer costs or size reasons
- ...



INTRODUCTION AND FUNDAMENTALS

Federated learning is a method that allows to **learn from remote data** while preserving privacy, security and other such aspects.



INTRODUCTION AND FUNDAMENTALS

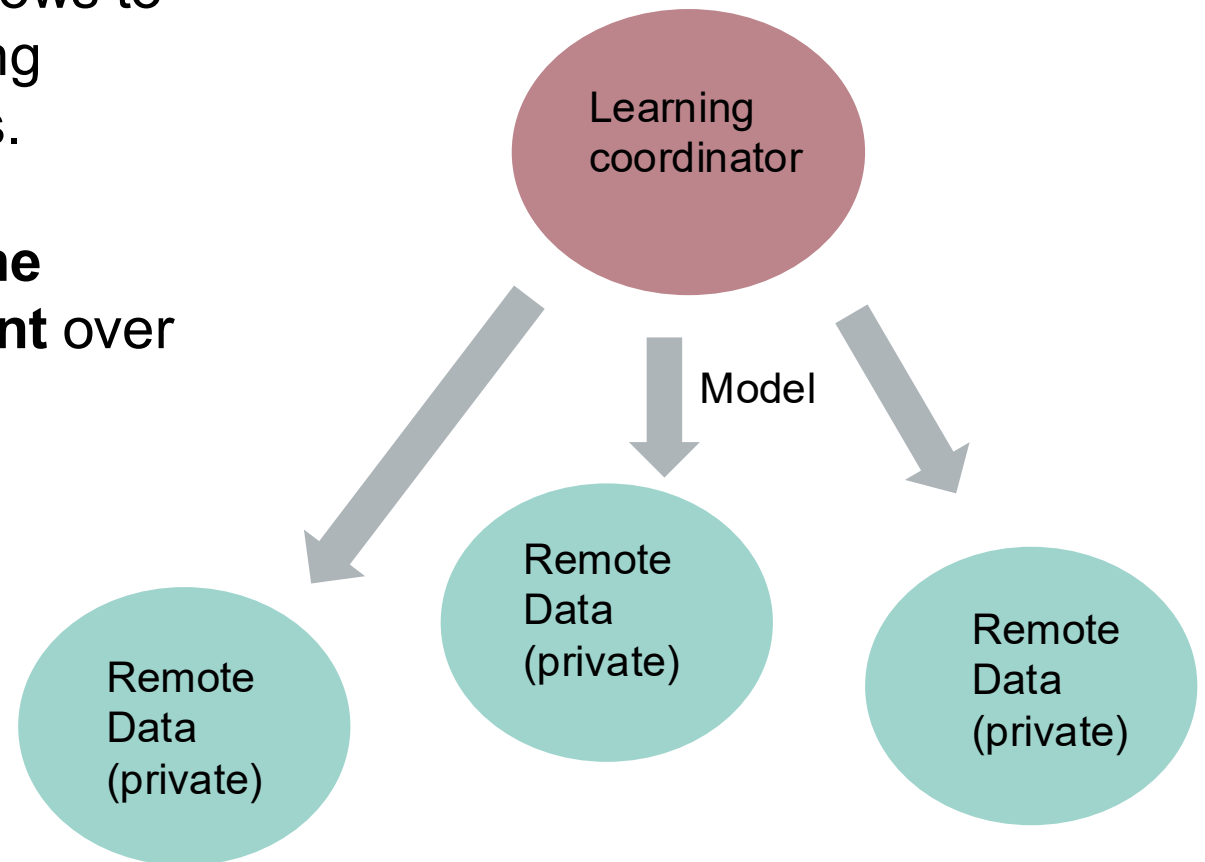
Federated learning is a method that allows to **learn from remote data** while preserving privacy, security and other such aspects.

Instead of sending the data, the **machine learning models** for the training **are sent** over a network to the remote locations.

The models are **trained at the edge**.

No original data is transmitted!

Core idea of federated learning:
train locally, share only model updates,
not raw data



THE FEDERATED LEARNING WORKFLOW

From local training to global model aggregation

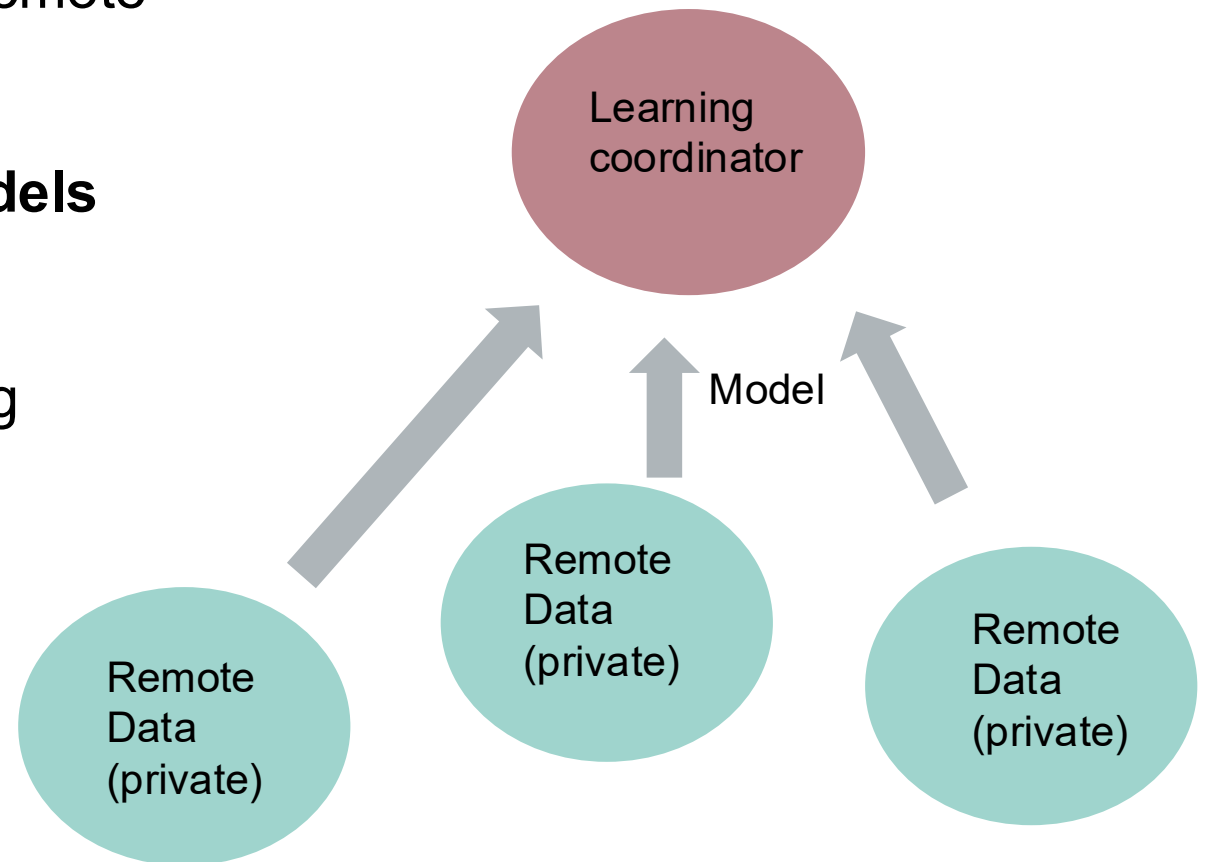


FEDERATED LEARNING WORKFLOW

Models are being trained locally in the remote location (at the edge).

Afterwards, only the **updated local models** are **collected** back by the global server.

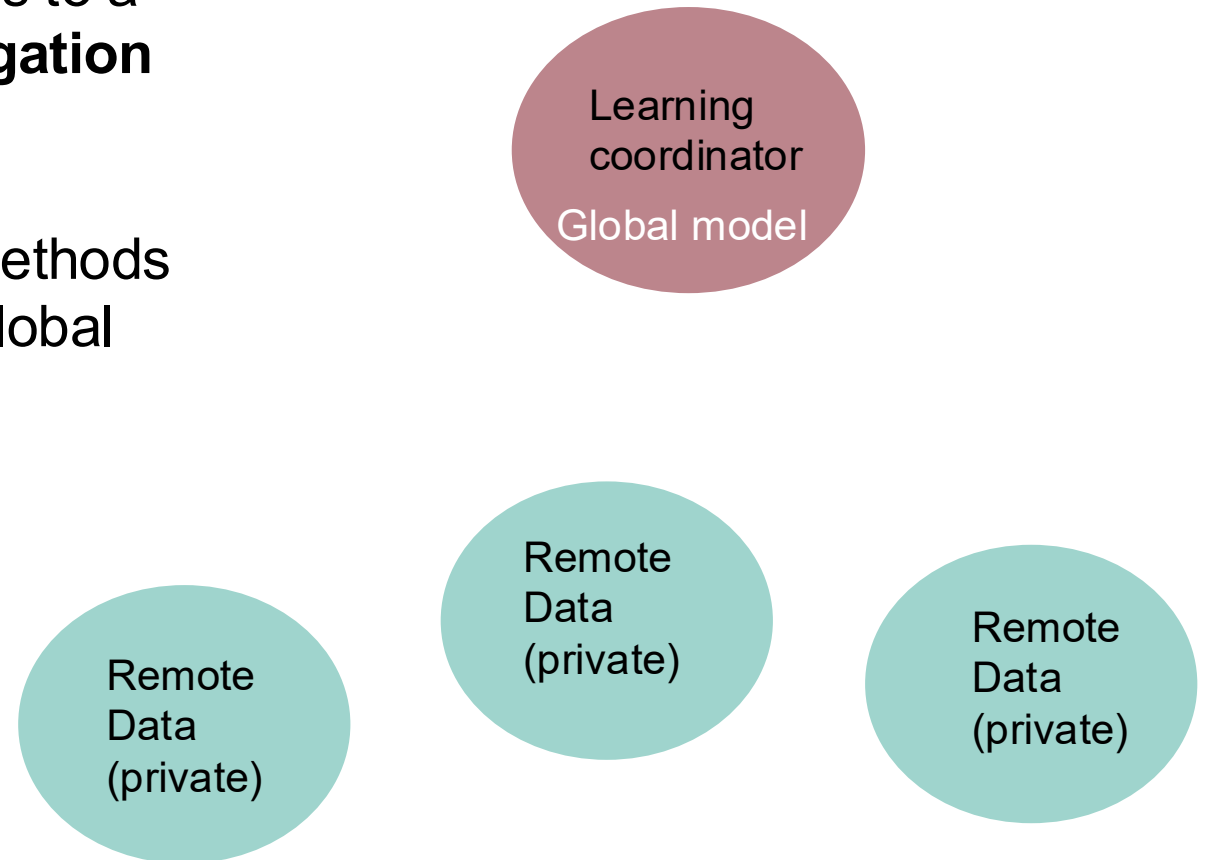
A **coordinator** orchestrates this learning process.



FEDERATED LEARNING WORKFLOW

The coordinator merges the local models to a **global model** with some form of **aggregation** step like parameter averaging.

There are many different aggregation methods to aggregate the local models into the global model.

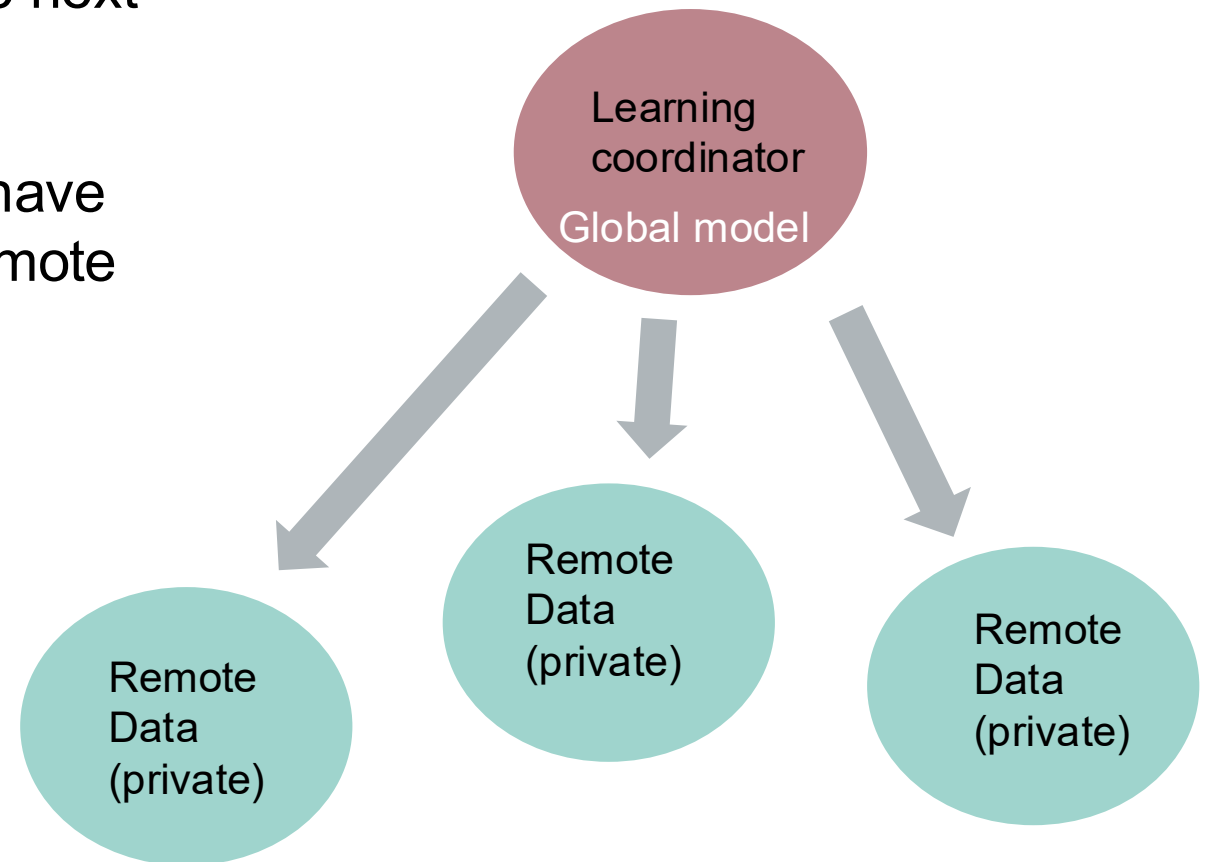


FEDERATED LEARNING WORKFLOW

The global model is then sent out for the next iteration of training.

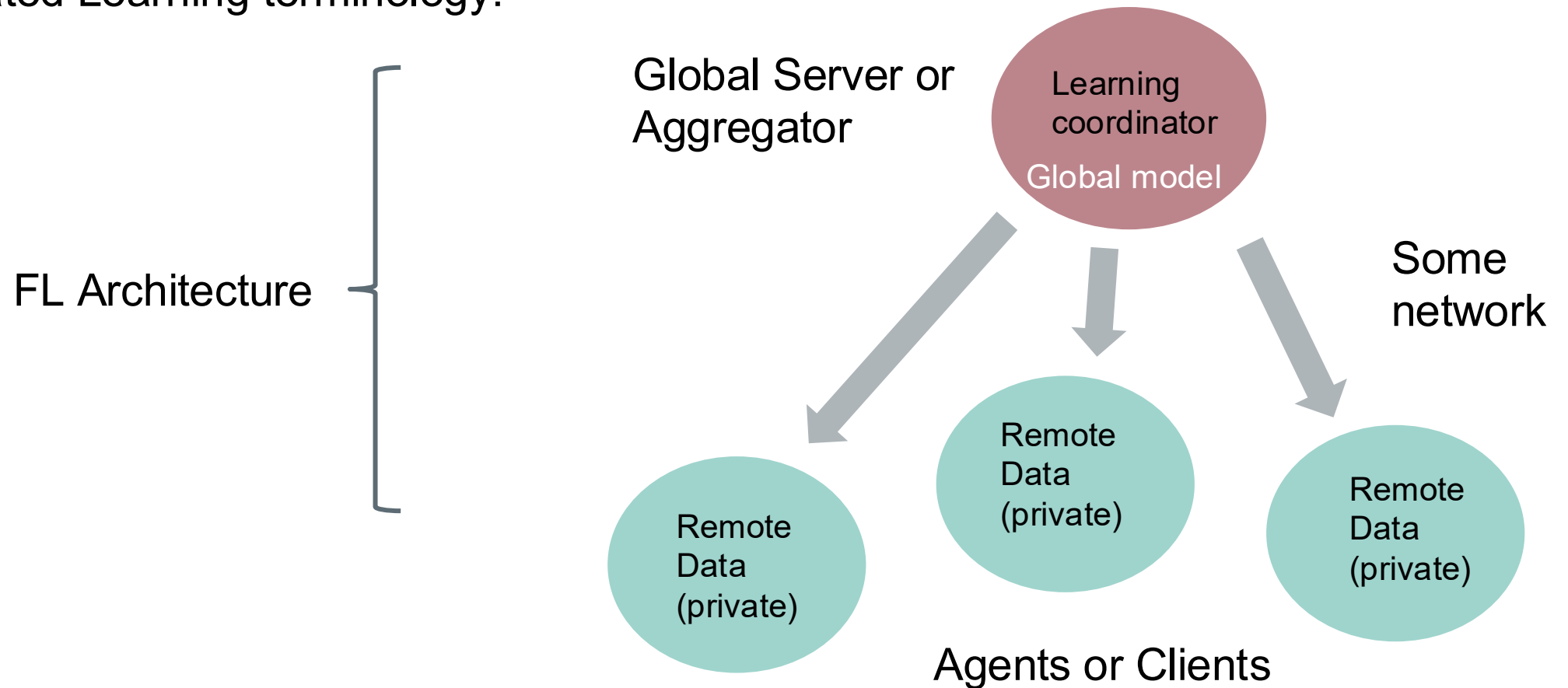
After N iterations, the global model will have learned common properties of all the remote data.

One such round of training is called a “communication round”.



FEDERATED LEARNING WORKFLOW

In Federated Learning terminology:



IMPORTANT ASPECTS

Challenges and trade-offs to be aware of



IMPORTANT ASPECTS

What kind of models work for FL?

Model has parameters that can be combined (aggregated) in some way.

→ well suited are:

- Neural Networks (CNN, RNNs, Transformers, ...)
- Regression models (linear, logistic, SVM,...)

→ with dedicated algorithms also tree-based models can work and some clustering algos

→ problems may arise with very large models (lots of parameters) and instance based models

IMPORTANT ASPECTS

What does happen in aggregation and what types are there?

Model parameters of the locally trained and updated models need to be combined (aggregated) in a manner that strongly depends on the type of application.



Each parameter is averaged element by element: $W = (W_1 + W_2 + W_3) / 3$
FedAvg weights each client by its share of the training samples: $W = \sum (n_k / n) \cdot W_k$

IMPORTANT ASPECTS

What does happen in aggregation and what types are there?

Model parameters of the locally trained and updated models need to be combined (aggregated) in a manner that strongly depends on the type of application.

→ Averaging, mean weighting: each client that sends an update in the communication round contributes to the global model equally.

IMPORTANT ASPECTS

What does happen in aggregation and what types are there?

Model parameters of the locally trained and updated models need to be combined (aggregated) in a manner that strongly depends on the type of application.

- Averaging, mean weighting: each client that sends an update in the communication round contributes to the global model equally.
- Weighted averaging (e.g. FedAvg algorithm and similar): takes into account that clients may have different amounts of local data the model is updated on.

IMPORTANT ASPECTS

What does happen in aggregation and what types are there?

Model parameters of the locally trained and updated models need to be combined (aggregated) in a manner that strongly depends on the type of application.

- Averaging, mean weighting: each client that sends an update in the communication round contributes to the global model equally.
- Weighted averaging (e.g. FedAvg algorithm and similar): takes into account that clients may have different amounts of local data the model is updated on.
- There are many more aggregation strategies, for example for maximum security or privacy, for very diverse datasets (non identically and independently distributed data – non-IID), ...

IMPORTANT ASPECTS

Data partitioning and heterogeneity

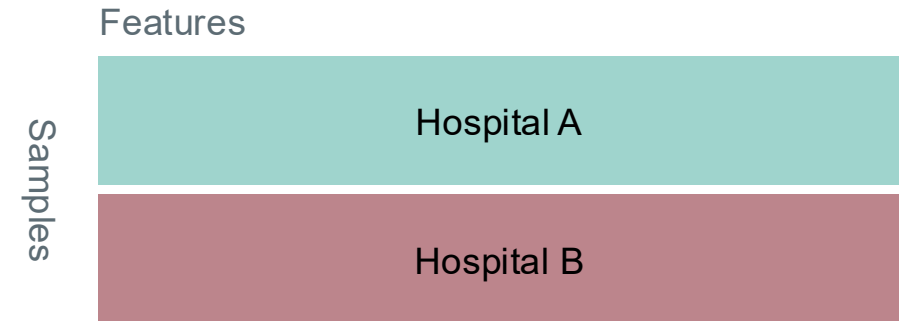
- **Data heterogeneity:** participants' data is often non-IID (not independently and identically distributed)
 - Clients differ in labels, feature distribution and data volume
 - Local models drift apart, so the global model converges slower

IMPORTANT ASPECTS

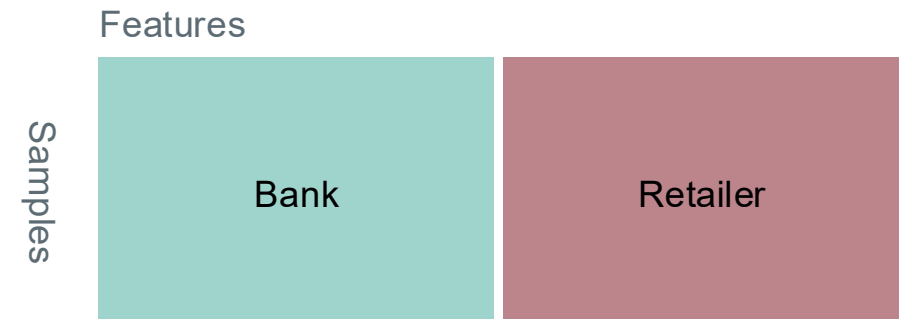
Data partitioning and heterogeneity

- **Data heterogeneity:** participants' data is often non-IID (not independently and identically distributed)
 - Clients differ in labels, feature distribution and data volume
 - Local models drift apart, so the global model converges slower
- **Horizontal FL:** same features, different samples
 - e.g. hospitals with the same record format but different patients
- **Vertical FL:** same samples, different features
 - e.g. a bank and a retailer with overlapping customers, linked by ID

Horizontal FL: split by samples



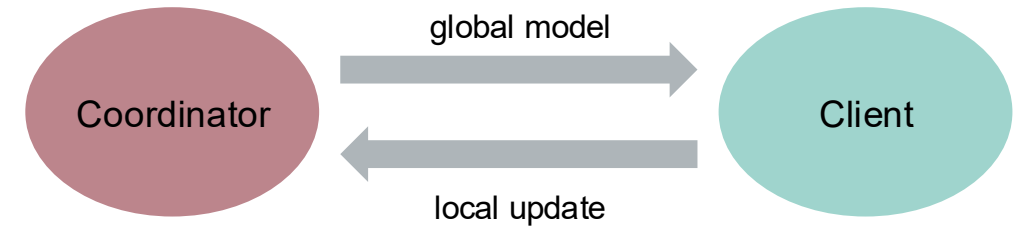
Vertical FL: split by features



IMPORTANT ASPECTS

Communication efficiency

- Each round sends the model to the clients and their updates back
- $\text{Cost} \approx \text{model size} \times \text{clients per round} \times \text{number of rounds}$
- Levers: fewer rounds, sampling of clients, compressed updates
- Trade-off: less traffic vs. slower convergence



Update size per client and round (schematic)

Full update (32-bit floats)



Quantized to 8 bit (4 × smaller)



Illustrative – real savings depend on model, task and method

IMPORTANT ASPECTS

Security & privacy

Raw data stays local – but model updates can still reveal information about it.

Risk

Updates leak training data (gradient inversion, membership inference)

The server sees every individual update

Malicious clients poison or backdoor the model

Mitigation

Differential privacy: clip updates and add calibrated noise

Secure aggregation: the server only learns the sum (also homomorphic encryption)

Robust aggregation (median, trimmed mean), client authentication, anomaly detection

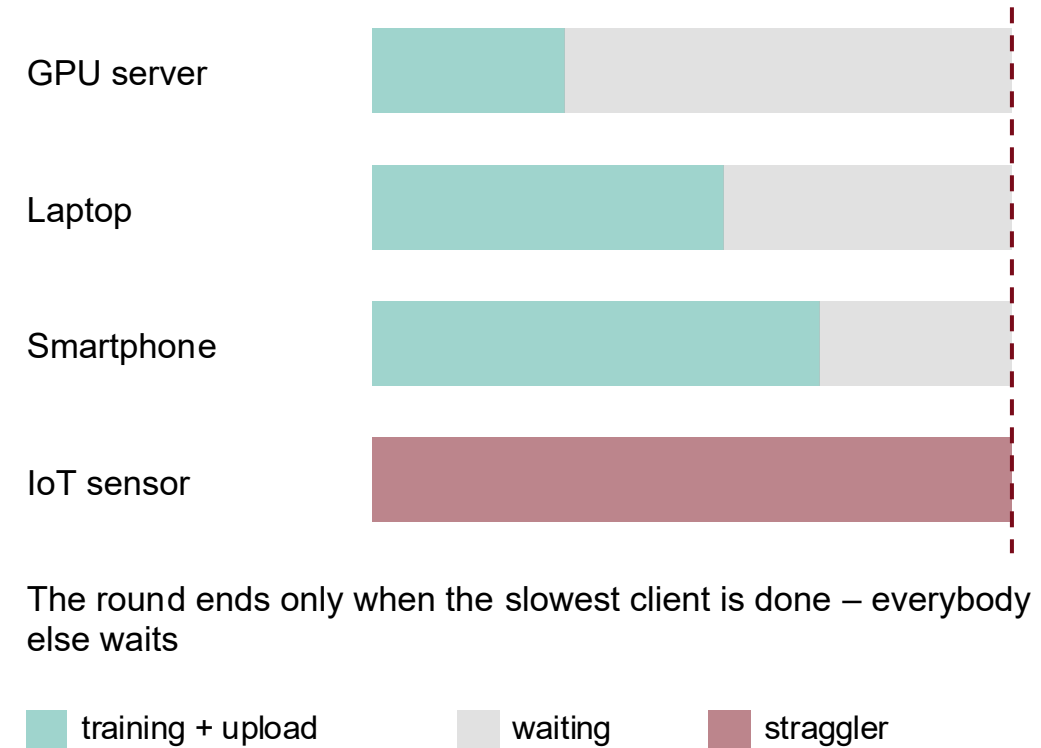
Trade-off: stronger protection costs accuracy, computation and communication

IMPORTANT ASPECTS

System heterogeneity

- Clients differ in compute, memory, connectivity and availability
- Synchronous rounds wait for the slowest client (straggler); devices drop out
- Cross-device (phones, sensors) vs. cross-silo (hospitals, utilities)
- Remedies: client selection, deadlines, asynchronous updates

One communication round (schematic)



EXAMPLE USE CASES

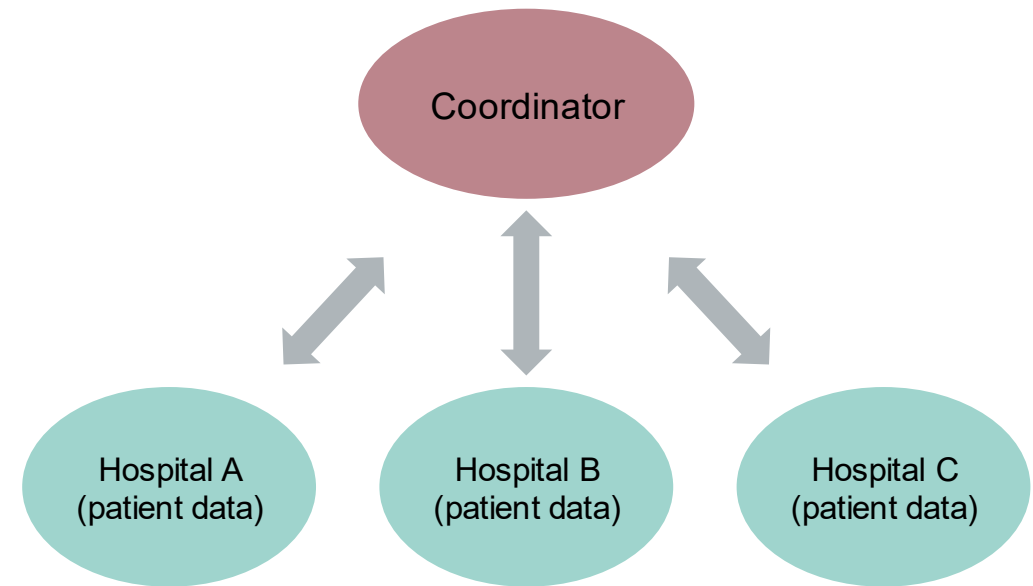
Where federated learning is already applied



EXAMPLE USE CASES

Healthcare

- Patient data cannot leave the hospital – yet no single site has enough data
- Cross-silo FL: a few reliable partners train locally, a coordinator aggregates
- Example: COVID-19 outcome prediction (“EXAM”) with 20 institutions worldwide



EXAM study vs. models trained at a single site

20

institutions

+16 %

average AUC

+38 %

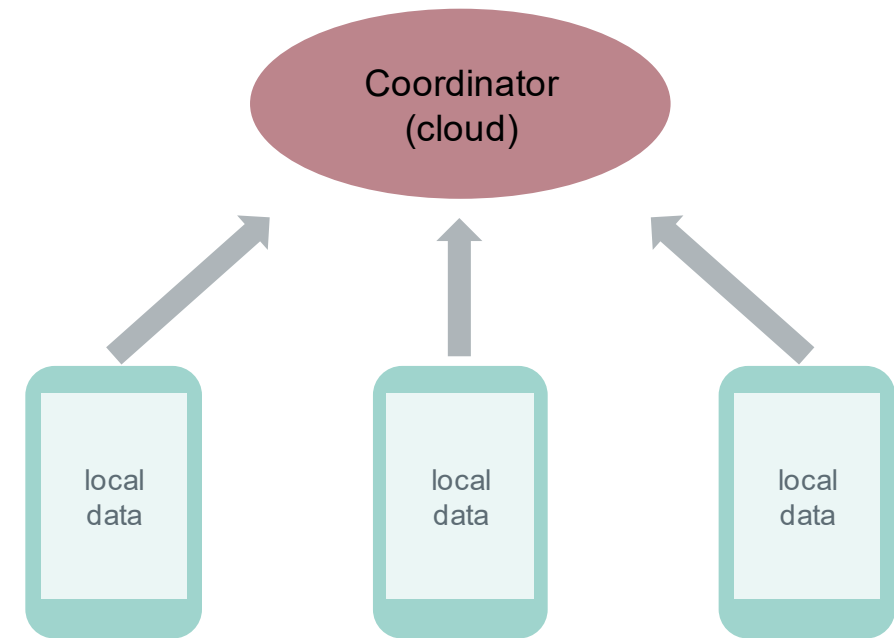
generalizability

*Source: Dayan et al., “Federated learning for predicting clinical outcomes in patients with COVID-19”, Nature Medicine 27 (2021)
<https://doi.org/10.1038/s41591-021-01506-3>*

EXAMPLE USE CASES

Mobile & edge devices

- Origin of FL: next-word prediction in Google's Gboard keyboard
- Typing data stays on the phone – only small model updates are uploaded
- Training runs only when the phone is idle, charging and on Wi-Fi
- Also IoT, wearables, smart home: many clients, little data each



Only small model updates leave the phone – never the typing data

Training starts only if the phone is:

idle

charging

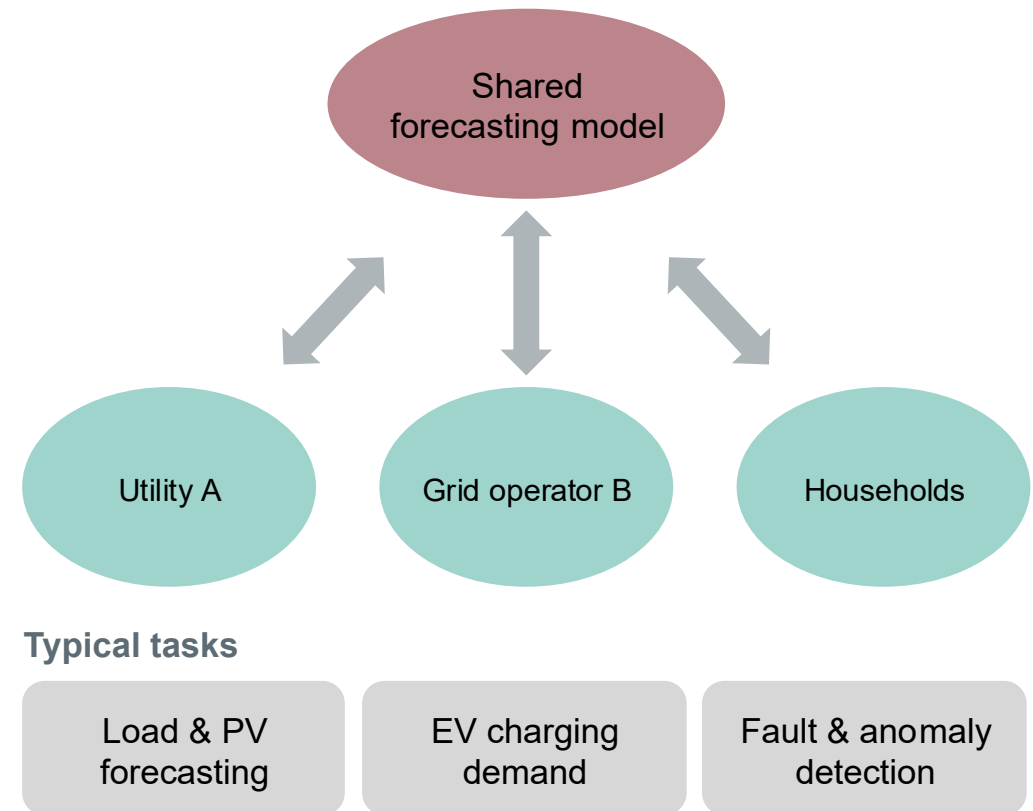
on Wi-Fi

Source: Google AI Blog, "Federated Learning: Collaborative Machine Learning without Centralized Training Data" (2017)

EXAMPLE USE CASES

Energy sector

- Smart-meter, PV generation, ... are sensitive – and spread over many owners
- Learn jointly across utilities and grid operators without exchanging raw data
- Cross-silo (utilities, grid operators) and cross-device (smart meters)
- Data differ by region, climate and consumer behaviour (non-IID)



EXAMPLE USE CASES

When to use federated learning – and when not to

Use FL when ...

- **Data can't leave its owner** – GDPR, confidentiality, bandwidth or cost
- **Several parties hold similar data**, but none has enough on its own
- **Data is created at the edge** in large volumes: phones, sensors, smart meters
- **The model can be aggregated**, e.g. neural networks or linear models

Prefer centralised ML when ...

- **The data can be pooled** – central training is simpler and usually more accurate
- **One party already has enough data** for the task
- **Clients lack compute or connectivity** to train locally and join rounds
- **Client data differs so much** that one shared model helps nobody

Rule of thumb: if you can centralise the data, do so – use FL when you can't.

KEY TAKEAWAYS

1

Data stays local, models travel

Clients train on their own data; only model parameters go to the coordinator.

2

Learning happens in rounds

Local training → aggregation (e.g. FedAvg) → new global model sent back to all clients.

3

Aggregation needs combinable models

Neural networks and linear models suit FL best: their parameters can be averaged.

4

Real-world data brings challenges

Non-IID data, communication cost and slow or dropped clients.

5

Use FL when data can't be centralised

Already applied in many different domains.

FEDERATED LEARNING FRAMEWORKS

Tools to get started



FEDERATED LEARNING FRAMEWORKS

FATE


TensorFlow
federated


NVIDIA FLARE

 Flower

OpenFL
intel.

 PySyft

 FedML

Federated
Learning 

FEDERATED LEARNING FRAMEWORKS

Framework	Maintainer	Typical use / strengths
TensorFlow Federated	Google	Integrated with TensorFlow; simulation and research on FL algorithms
Flower (flwr)	Flower Labs	Framework-agnostic (PyTorch, TensorFlow, ...); simple, extensible API
NVIDIA FLARE	NVIDIA	SDK for secure, scalable FL across organisations (cross-silo), enterprise- ready
FedML	TensorOpera AI	Research and deployment platform; many data-partition protocols and non-IID methods

Choose by: ML framework · simulation vs. real deployment · cross-device vs. cross-silo · built-in privacy features · open source ...

More options (FATE, OpenFL, PySyft, ...)

QUESTIONS & DISCUSSION

10 minutes



RESOURCES AND REFERENCES



FEDERATED LEARNING

- **Book**

Federated Learning with Python: Design and implement a federated learning system and develop applications using existing frameworks von Kiyoshi Nakayama, George Jenö

- **Pages**

- <https://research.google/blog/federated-learning-collaborative-machine-learning-without-centralized-training-data/>
- <https://github.com/weimingwill/awesome-federated-learning>
- <https://github.com/poga/awesome-federated-learning>
- <https://flower.ai/docs/framework/tutorial-series-what-is-federated-learning.html>

THANK YOU!

Questions? Get in touch – sarah.reisenbauer@ait.ac.at

