Federated AI communication protocol

Václav Oujezský

Telč 2023

Overview

- Motivation for the topic
- Federated learning
- Current research projects on the given topic
- Current challenges and conclusion

Motivation

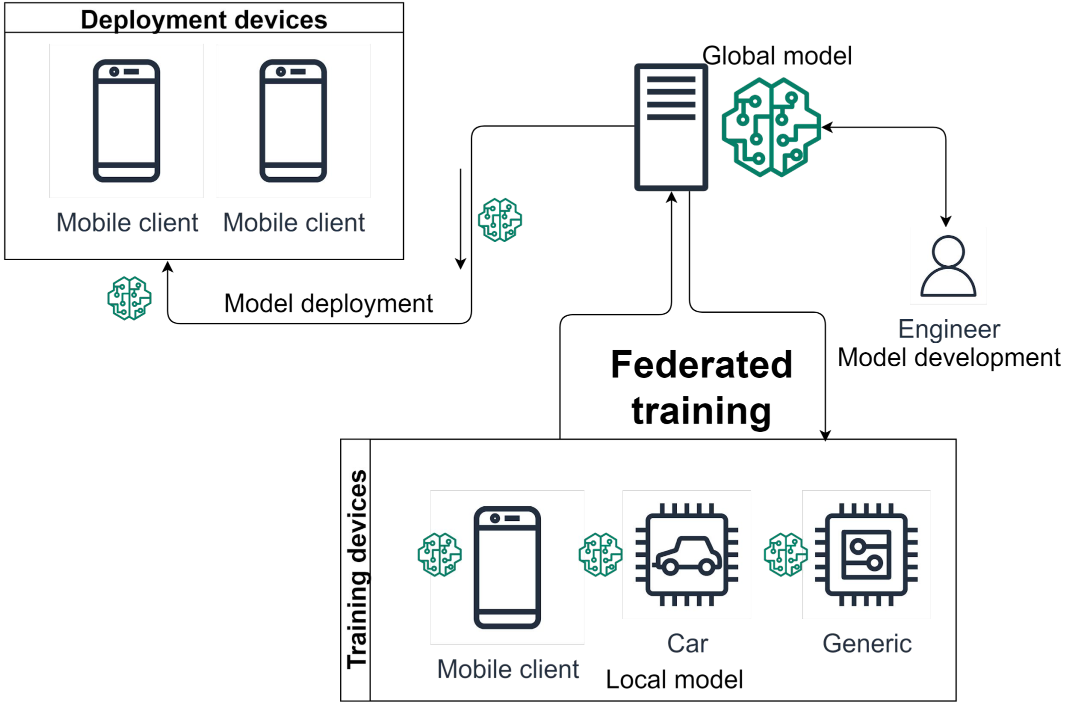
- Safety
    - Data privacy, transfer of sensitive data
- Efficiency
    - Federated learning, federated computing
- Development
    - New possibilities of use (ad hoc encryption, mobile communication, others)

Federated learning?

Data processing is increasingly done on end devices

- Improving responsiveness
- Benefits of data security
- Analytics
- Learning

Federated learning?



Federated learning?

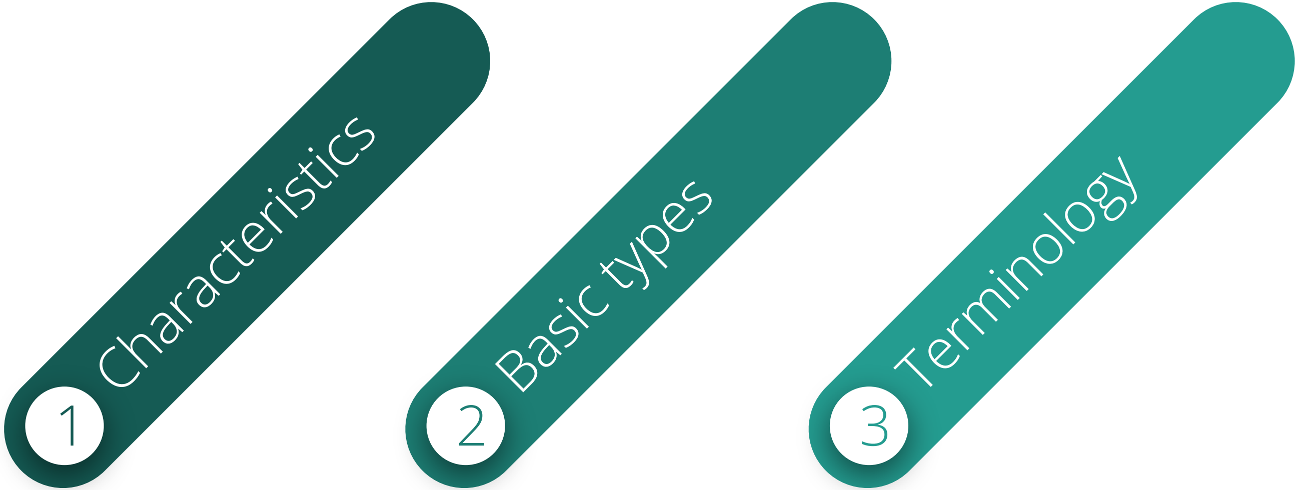
Specifically defined in 2019 in the publication

Advances and Open Problems in Federated Learning (https://arxiv.org/abs/1912.049770)

Citation:

Federated learning (FL) is a machine learning setting where many clients (e.g. mobile devices or whole organizations) collaboratively train a model under the orchestration of a central server (e.g. service provider), while keeping the training data decentralized. FL embodies the principles of focused data collection and minimization, and can mitigate many of the systemic privacy risks and costs resulting from traditional, centralized machine learning and data science approaches. Motivated by the explosive growth in FL research, this paper discusses recent advances and presents an extensive collection of open problems and challenges.

Federated learning



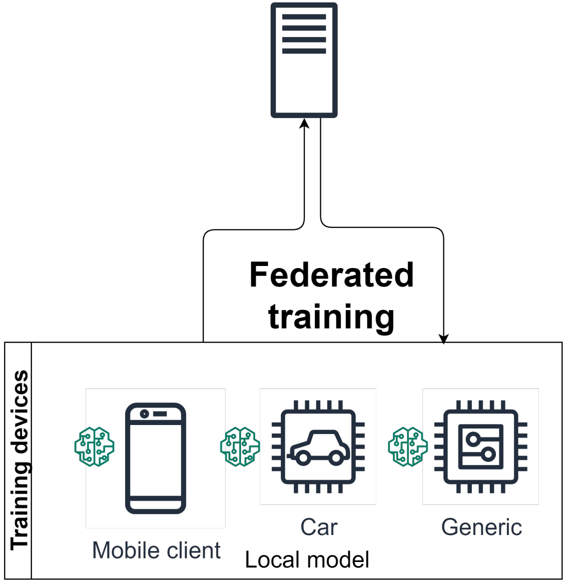
1 Characteristics

2 Basic types

3 Terminology

Federated learning – characteristic

- Data is generated locally
- Data remains decentralized
- Data is not shared and distributed
- A central orchestrator coordinates training, but never sees its own data



Federated learning – basic types

Cross-silo federated learning (between objects, institutions)

- smaller number of clients, high availability
- object identity
- each object participates in each round of learning
- computational complexity is the primary weak point

Cross-device federated learning (between end devices)

- hundreds of temporarily available clients
- no identification
- usually each client participates only once
- communication is the primary weak point

Decentralized learning

- peer-to-peer, without a centralized orchestrator

Federated learning – terminology

- **Client**
  - Computing entity holding local data (mobile device, IoT, institution, …)
- **Server**
  - Federated learning coordinator, **nowadays, more than one device**

Federated learning

1 Algorithm

2 Variations

3 Frameworks

Federated learning – basic algorithm

- **Client**
  - The current **state of the model** is **sent** to the client from the central element
  - The client trains it with local data
  - The principle of training with the SGD (Stochastic Gradient Descent) method or its derivation as a function of evaluation
  - The result is **model parameter weights** that are **sent** to the central element
- **Server**
  - It obtains parameter weights from clients and performs averaging (Federated Average) and updates the original model
  - The next procedure already depends on the specific solution (cross-silo, cross-device)

Federated learning – basic principles

- **Optimization of the objective function for the purpose of convergence of weights**

$$f(w_1, \ldots, w_K) = \frac{1}{K} \sum_{i=1}^{K} f_i(w_i)$$

- **Hyperparameters**
    - number of epochs, learning rate, batch size

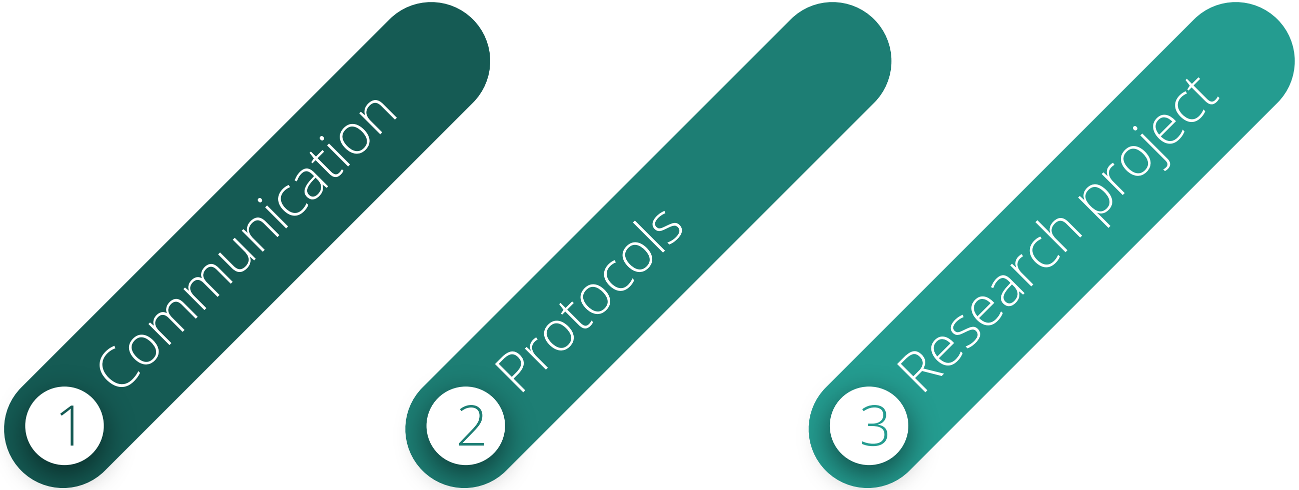
Federated learning – list of variations

- **Federated stochastic gradient descent (FedSGD)**
  - transpose SGD, swap gradients, random number of clients, gradients are averaged and gradient step performed
- **Federated averaging (FedAvg)**
  - generalization of FedSGD, exchange of updated weights, not gradients
- **Federated Learning with Dynamic Regularization (FedDyn)**
  - 2021(Acar, et al.), the issue of heterogeneous data distribution. It addresses the dilemma of the difference between device error minimization and global error minimization. Using dynamic control to converge local and global error.
- **Hybrid Federated Dual Coordinate Ascent (HyFDCA)**
  - 2022 (Overman et al.), solves the problems of hybrid federated learning, where each client solves only a certain subset of data samples (features, samples).
- **KafkaFed**
  - An example of a design of a federated learning algorithm using Apache Kafka as a **communication medium**

Federated learning – frameworks

- **TensorFlow Federated** – opensource Google Brain 2019, use TFLite flatbuffer files, mostly focused on simulations, still in development, FedAvg integration
- **FederatedAITechnologyEnabler(FATE)** – Webank, the first open source at an industrial level – in the area of credit risk control, object detection and anti-money laundering
- **IBM Federated Learning –** possible to use for free, tools for implementation in real situations
- **PySyft –** the open source solution OpenMined, originally using PyTorch, then also using TensorFlow
- **DeepLearning4J** – opensource Konduit for JVM, Python, C++
- **Visioner** - based on TensorFlow, uses inference graph* files, org.tensorflow:tensorflow-android)
- **Flower** – uses TensorFlow, possibility of use on AWS, Azure, Android, iOS, Raspberry, Nvidia Jetson
- **FedML** – cross platform solution

Federated learning



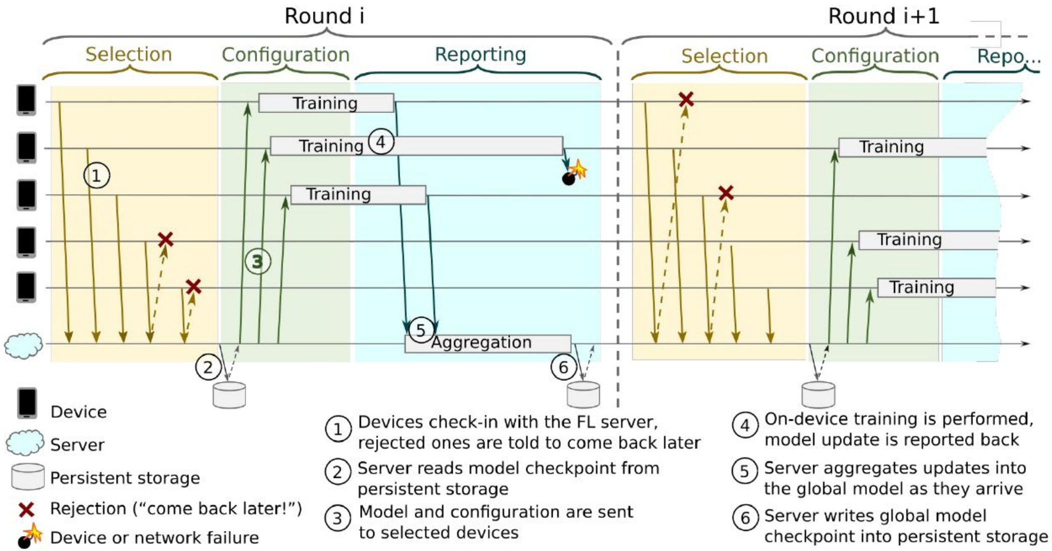
1 Communication

2 Protocols

3 Research project

Federated learning – communication

An example cross-device federated learning protocol



Device

Server

Persistent storage

✕ Rejection (“come back later!”)

Device or network failure

1. Devices check-in with the FL server, rejected ones are told to come back later
2. Server reads model checkpoint from persistent storage
3. Model and configuration are sent to selected devices
4. On-device training is performed, model update is reported back
5. Server aggregates updates into the global model as they arrive
6. Server writes global model checkpoint into persistent storage

Bonawitz, et. al. ***Towards Federated Learning at Scale: System Design.*** *MLSys 2019.*

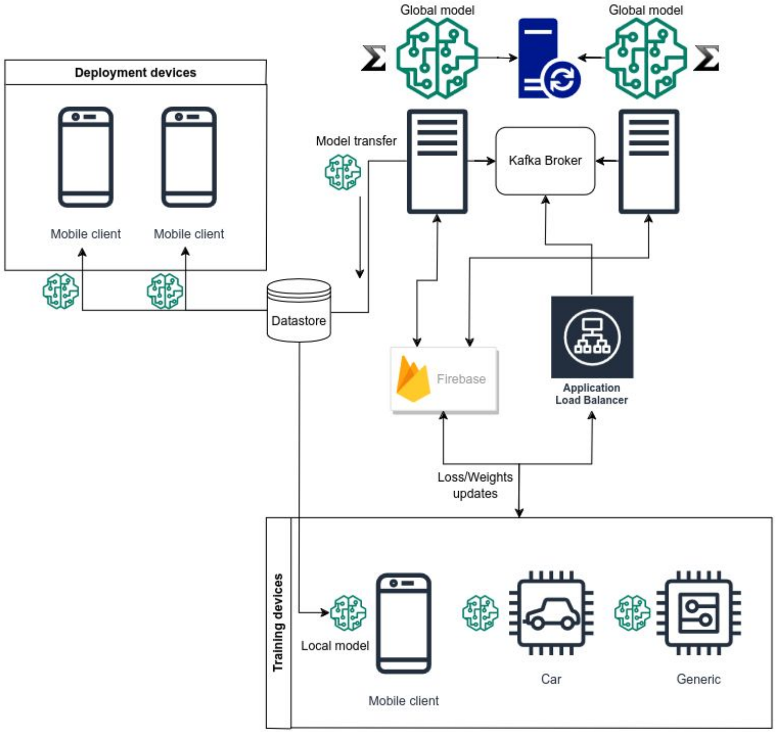
Federated learning – communication

Transmission of updated data

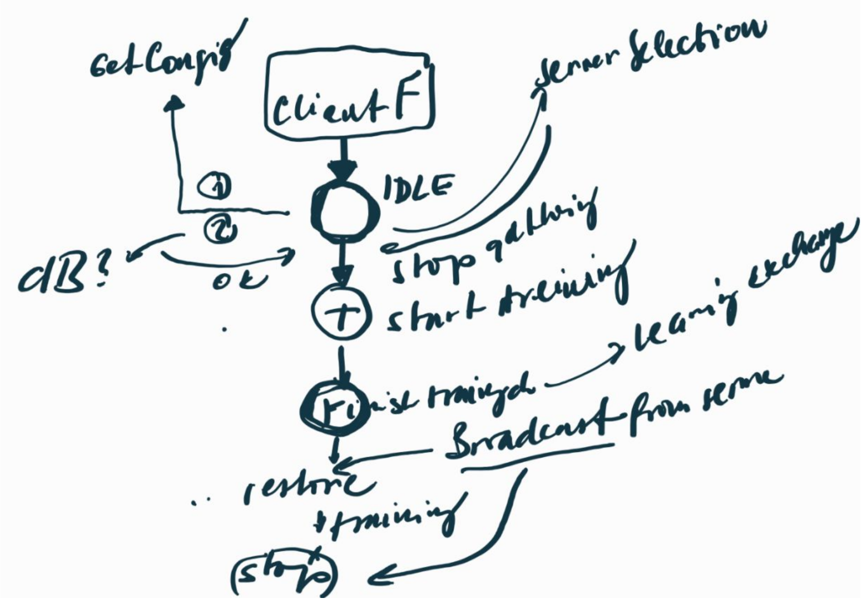
- most often gRPC (HTTP/2 and protocol buffer, Google 2015)
- also REST HTTP API (Ktor, Retrofit2, others)
- or Apache Kafka, dynamic databases like Google Firebase

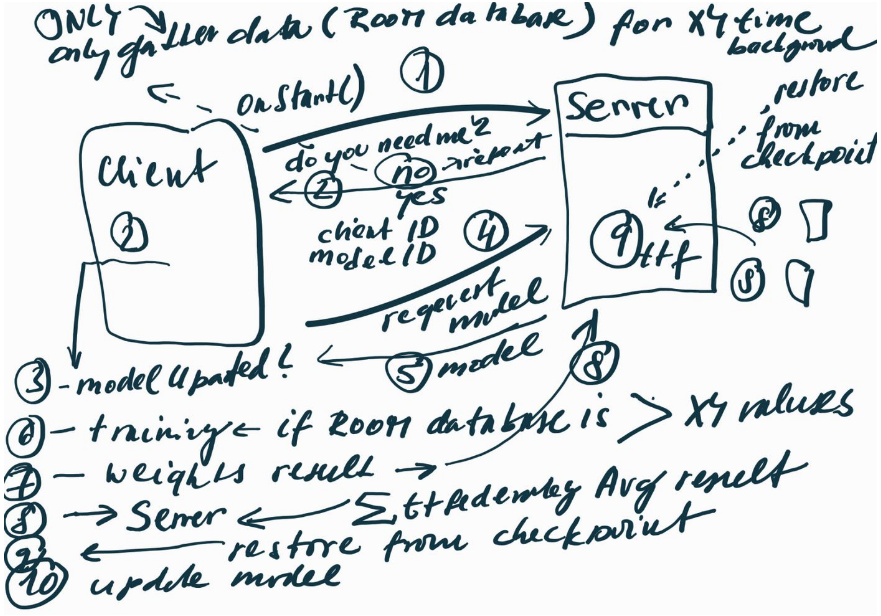
https://www.tensorflow.org/federated/tutorials/building_your_own_federated_learning_algorithm

Federated learning – communication



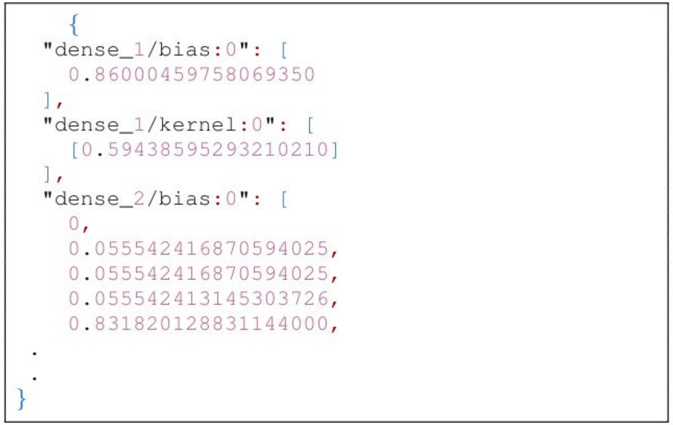
Federated learning – communication protocol, ideas



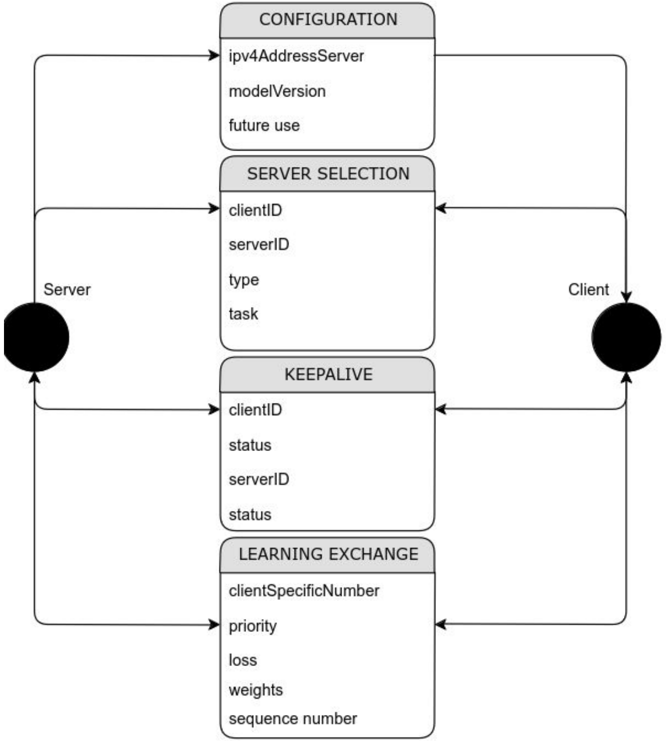


Federated learning – proposed communication protocol

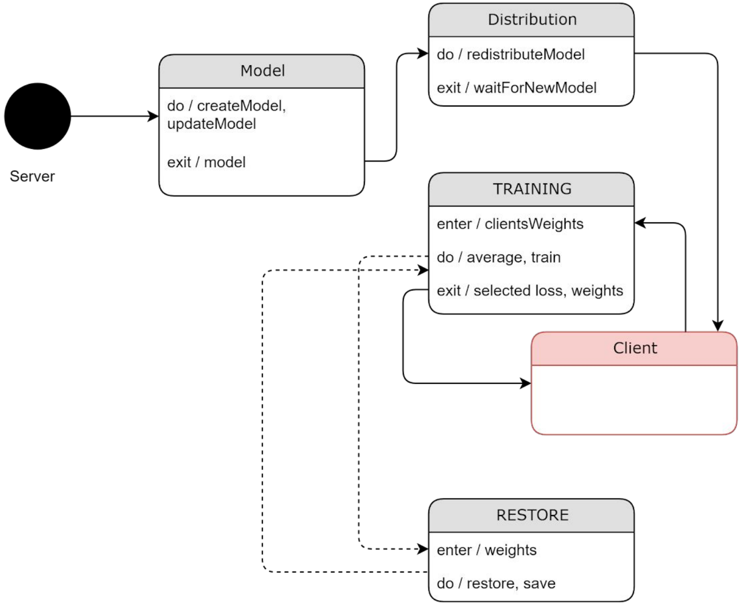
Compatible with both, Apache Kafka and Realtime database
Using a text format for the transmission of messages
Automatic selection of central nodes
Seemless communication

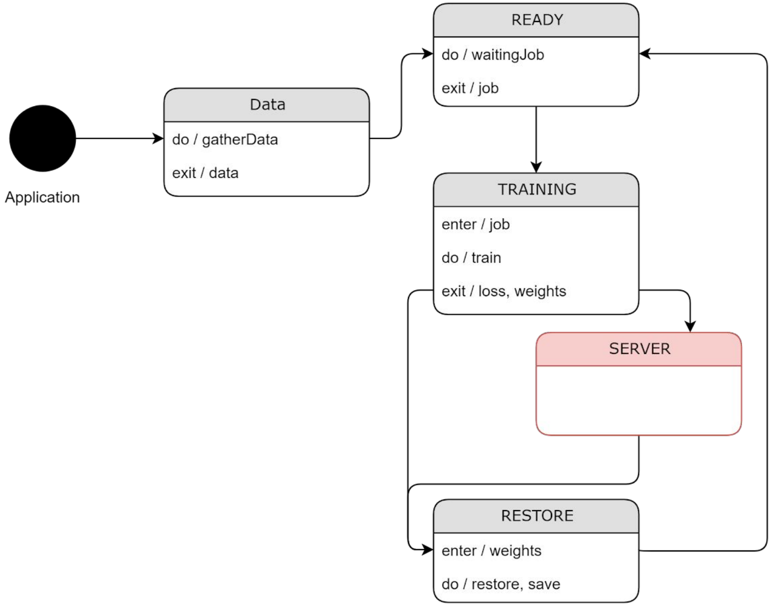


```
    {
  "dense_1/bias:0": [
    0.86000459758069350
  ],
  "dense_1/kernel:0": [
    [0.59438595293210210]
  ],
  "dense_2/bias:0": [
    0,
    0.055542416870594025,
    0.055542416870594025,
    0.055542413145303726,
    0.831820128831144000,
 .
 .
}
```



Federated learning – proposed communication protocol





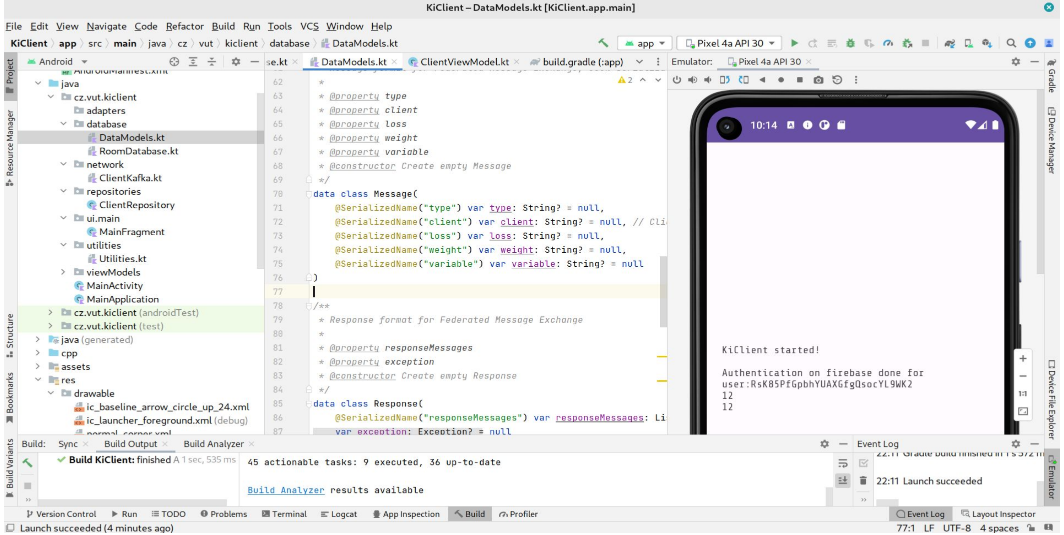
Federated learning – proposed communication protocol

J. Michalek, V. Skorpil and V. Oujezsky, "Federated Learning on Android - Highlights from Recent Developments," *2022 14th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, Valencia, Spain, 2022, pp. 27-30, doi: 10.1109/ICUMT57764.2022.9943382.

J. Michalek, V. Oujezsky, and V. Skorpil An Android Federated Learning Framework for Emergency Management Applications, 2023, ICUMT.

Federated learning – research project

Android federated learning framework for crisis management applications



```
 *
 * @property type
 * @property client
 * @property loss
 * @property weight
 * @property variable
 * @constructor Create empty Message
 */
data class Message(
    @SerializedName("type") var type: String? = null,
    @SerializedName("client") var client: String? = null, // Cli
    @SerializedName("loss") var loss: String? = null,
    @SerializedName("weight") var weight: String? = null,
    @SerializedName("variable") var variable: String? = null
)

/**
 * Response format for Federated Message Exchange
 *
 * @property responseMessages
 * @property exception
 * @constructor Create empty Response
 */
data class Response(
    @SerializedName("responseMessages") var responseMessages: Li
    var exception: Exception? = null
```

```
KiClient started!

Authentication on firebase done for
user:RsK85PfGpbhYUAXGfgQsocYL9WK2
12
12
```

```
Build KiClient: finished At 1 sec, 535 ms
45 actionable tasks: 9 executed, 36 up-to-date

Build Analyzer results available
```

```
22:11 Gradle build finished in 1 s 572 ms
22:11 Launch succeeded
```

Launch succeeded (4 minutes ago)

Telč 2023

Current challenges

Communication problems

- Security risks
    - Transfer of model, weights
- Principles of data transfer, protocols, synchronization synchronization
- Decentralized learning

Telč 2023

Thank you

oujezsky@vut.cz

This presentation is based upon the grant of the Ministry of the Interior of the Czech Republic, Open challenges in security research, VK01030152, ``Android federated learning framework for emergency management applications ''.

MINISTERSTVO VNITRA
ČESKÉ REPUBLIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
department of telecommunications

Federated learning – appendix

Federated learning – FedSGD

- **Federated stochastic gradient descent (FedSGD)**
  - transpose SGD, swap gradients, random number of clients, gradients are averaged and gradient step performed

$$F_k(w) = \frac{1}{\boxed{n_k}} \sum_{\boxed{i\in P_k}} \boxed{f_i(w)}$$

Client Data Count

A set of client data

Loss $l(x_i, y_i; w)$ on the data sample with the parameters of the weights

$$\boxed{g_k} = \nabla F_k(w_t)$$

The server aggregates the gradients and applies the update

$$w_{t+1} \leftarrow \boxed{w_t} - \boxed{\eta} \sum_{k=1}^{K} \frac{n_k}{n} g_k$$

Model weights in the communication round

training rate

Federated learning – FedAvg

- **Federated averaging (FedAvg)**
  - generalization of FedSGD, exchange of updated weights, not gradients

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

One step GD locally on the client

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

The server updates the model with a weighted average of the weights

In practice, calculations are performed on clients in batches of data