CS-E4740 - Federated Learning

L1 - From ML to FL

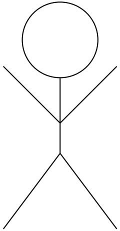
Assoc. Prof. Alexander Jung

Spring 2026

Calendar **Glossary** **Book** **GitHub**

Table of Contents

The Right Song Can Save the Day



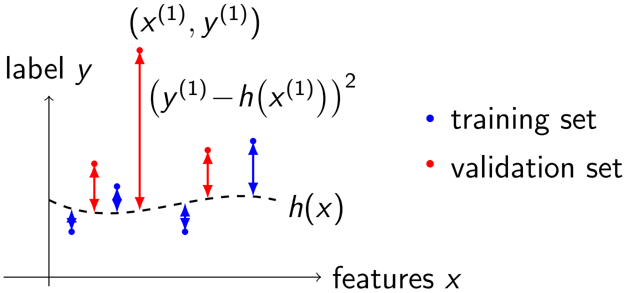
user: Alex Jung
mood: tired
evening

$\xrightarrow{h(x)}$ **Bohemian Rhapsody**

How do we get a good hypothesis map $h(x)$?

Wang, M., Wu, J., Yan, H. (2023). "Effect of music therapy on older adults with depression: A systematic review and meta-analysis." *Complementary Therapies in Clinical Practice* https://doi.org/10.1016/j.ctcp.2023.101809

Empirical risk minimization (ERM)



Learn $h \in \mathcal{H}$ by min. average loss (empirical risk),

$$\min_{h \in \mathcal{H}} \frac{1}{m} \sum_{r=1}^{m} L\left((\mathbf{x}, y), h\right).$$

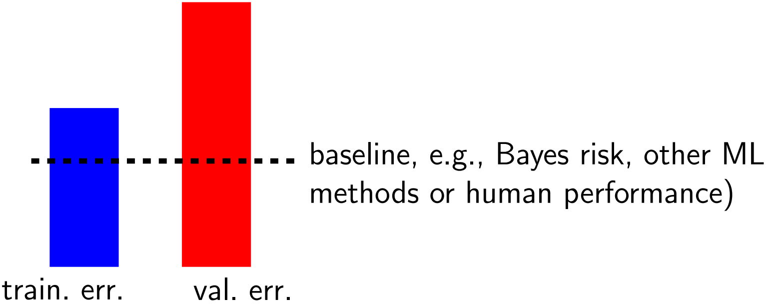
Different choices for $\mathcal{H}$ and loss L yield different ML methods.

see Chapters 3,4 of AJ, "Machine Learning: The Basics," Springer, 2022. https://mlbook.cs.aalto.fi

ML with Python

```
X, y = read_data()
model = SGDRegressor()
model.fit(X, y)
```

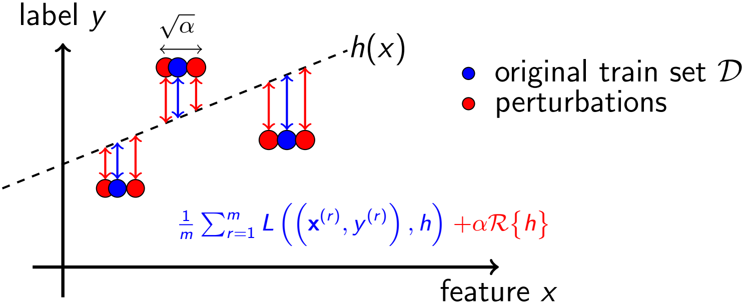
Applied ML - Trial and Error



ML diagnosis by comparing training error with validation error and a a baseline.

see Chapter 6 of AJ, "Machine Learning: The Basics," Springer, 2022. https://mlbook.cs.aalto.fi

Applied ML - Regularization



Start with large $\mathcal{H}$, then shrink it via (combinations of)

- data augmentation, e.g., $\mathbf{x} \mapsto \mathbf{x} + \mathcal{N}(0, \alpha)$,
- adding penalty term to loss function, e.g., $\ldots + \alpha \|\mathbf{w}\|_2^2$,
- **constraining** model parameters, e.g., $\|\mathbf{w}\|_2 \leq 1$.

---

see Chapter 7 of AJ, "Machine Learning: The Basics," Springer, 2022.
https://mlbook.cs.aalto.fi

Table of Contents

What is FL?

FL trains ML models over a network of devices.

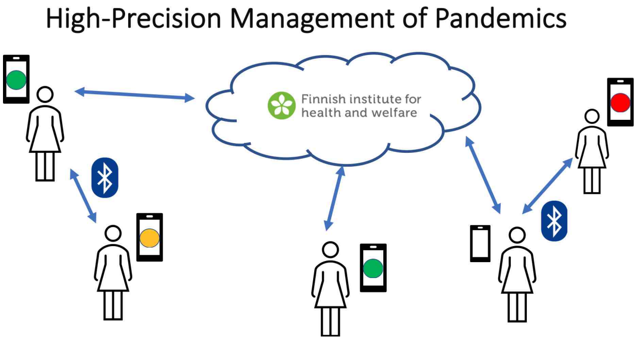


Figure: A hypothetical FL system for pandemic forecasting. Smartphones train personalized models based on their observations (e.g., audio recordings of coughing) as well as public health-care data.

Devices

We use the term device broadly.

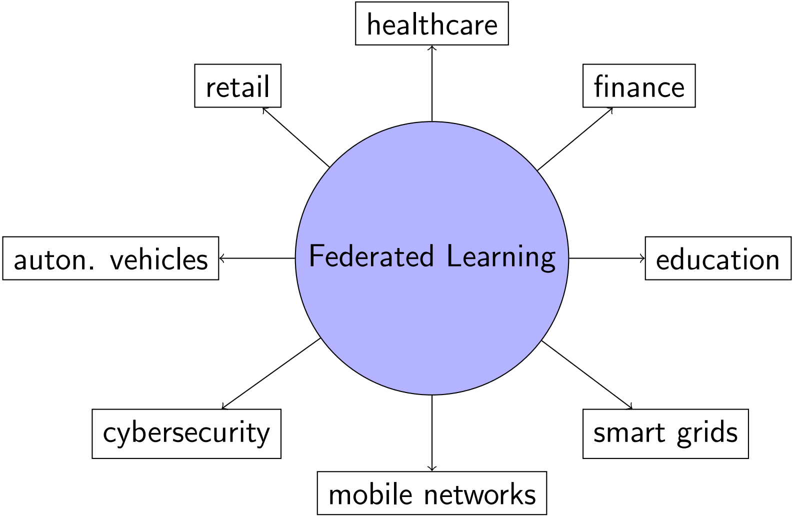
It is any computational system that is able to

- access data,
- train a model, and
- communicate with other devices.

Key Characteristics of FL

- No centralized data collection (no single point of failure)).
- Each device trains a tailored model (high-precision).
- Scalability: more devices yield more compute and data.
- No raw data is shared (privacy-friendly).

FL Applications



FL in Healthcare

- Turn smartphone into personal health-care advisor.
- Smartphone app uses FL to train personalized model.
- Combine personal data with public health-care data.

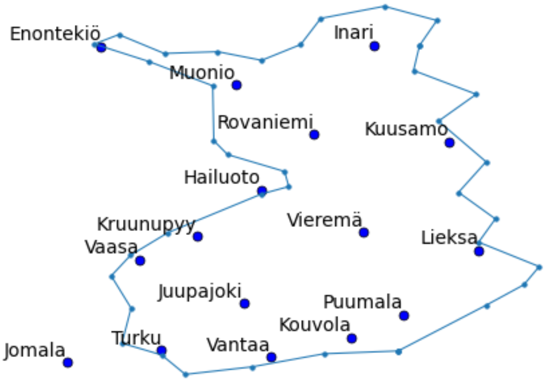
Key Reference: Rieke, N., et al. *The future of digital health with federated learning.* Nature Medicine, 2020.

FL in Finance

FL can help financial institutions to improve

- **Fraud detection.** N. F. Aurna, et.al., "Federated Learning-Based Credit Card Fraud Detection: Performance Analysis with Sampling Methods and Deep Learning Algorithms," 2023,
- **Risk assessment.** W. Li, et.al., "Personal Credit Evaluation Model Based on Federated Learning," 2024

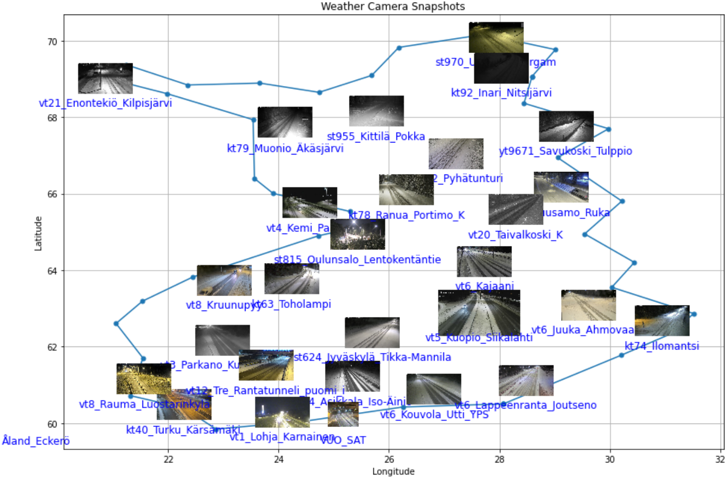
FL at FMI



Train a separate model for each Finnish Meteorological Institute (FMI) weather station

Python script for reproducing the Fig.:

FL for Finnish Road Safety



Train separate model for each camera operated by FinTraffic

Python script for reproducing the Fig.:

The Internet of Things (IoT) is Growing

IoT connections (billion)

IoT	2023	2029	CAGR
Wide-area IoT	3.6	7.2	12%
Cellular IoT	3.4	6.7	12%
Short-range IoT	12.1	31.6	17%
Total	**15.7**	**38.8**	**16%**

Note: Based on rounded figures. Cellular IoT figures are also included in the figures for wide-area IoT.

Figure: Some IoT statistics from .

The IoT - A Global FL System

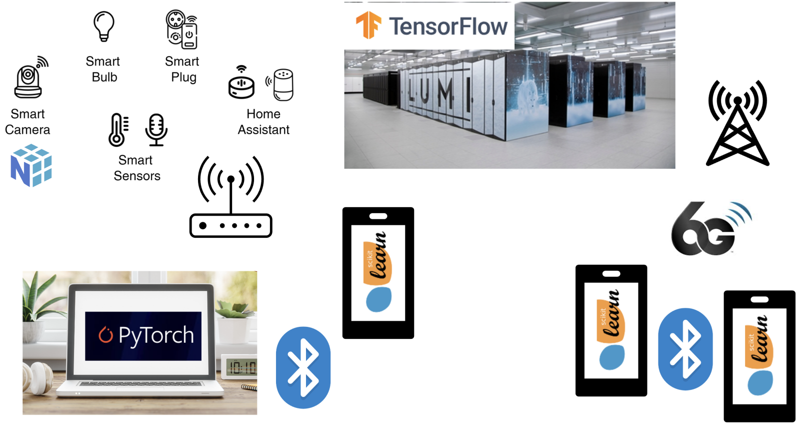


Table of Contents

From ML to FL

- Basic ML: Train a single model $\mathcal{H}$ by minimizing average loss on a single dataset
- FL: Train a separate model $\mathcal{H}^{(i)}$ for each node i of an interconnected FL system.

From ML to FL

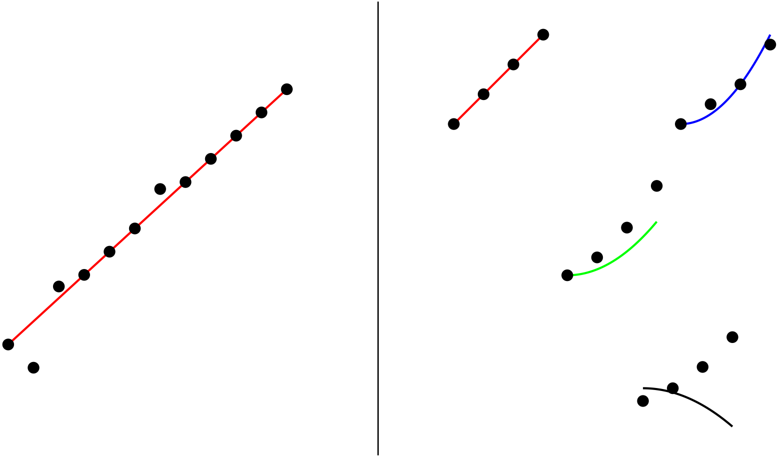


Figure: Left: A ML method uses a single dataset to train a single model. Right: FL methods train ML models from decentralized data.

ML with Python

```
X, y = read_data()
model = SGDRegressor()
model.fit(X, y)
```

FL with Python

IP: 192.168.0.1

```
model = SGDRegressor()
y_hat =recv_preds(192.168.0.3)
X, y = read_data()
Xa,ya=augment_data(X, y, y_hat)
model.fit(Xa,ya)
```

IP: 192.168.0.2

```
X,y = read_data()
model=LinearRegression()
model.fit(X, y)
```

IP: 192.168.0.3

```
model=DecisionTree()
y_hat =recv_preds(192.168.0.2)
X, y = read_data()
Xa,ya=augment_data(X, y, y_hat)
model.fit(Xa,ya)
```

Table of Contents

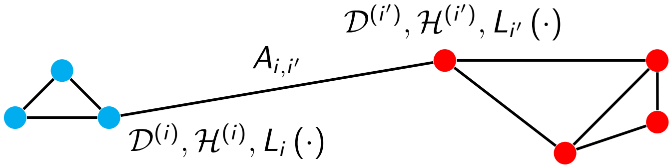
A (“Real-World”) FL System

Abstracting Away Details

To analyze an FL system, we (need to) ignore many details:

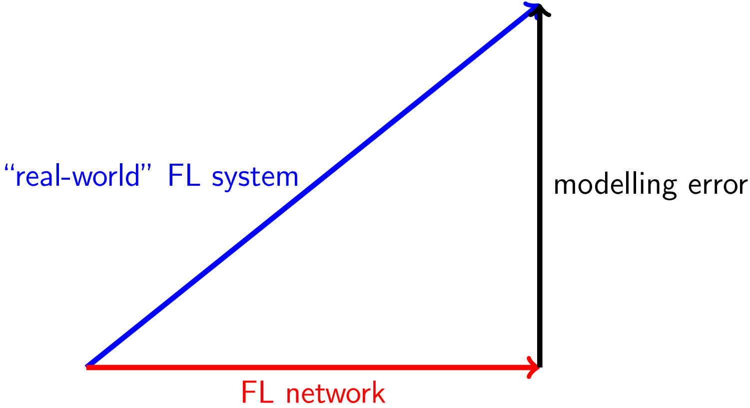
- physical properties of communication links
- low-level communication protocols
- hardware configuration of devices
- operating systems of devices
- scientific computing software (Python packages)

An FL Network



- FL network consists of devices, denoted $i = 1, \ldots, n$.
- Some i, i' connected by edge with the weight $A_{i,i'} > 0$.
- Device i **generates data** $\mathcal{D}^{(i)}$ and **trains model** $\mathcal{H}^{(i)}$.
- Data $\mathcal{D}^{(i)}$ used to construct loss func. $L_i(\cdot)$.

FL network is an Approximation



A Precise Definition

An FL network consists of:

- a finite set of **nodes**, denoted as $\mathcal{V} := \{1, \ldots, n\}$
- a **local model** $\mathcal{H}^{(i)}$ at each node $i \in \mathcal{V}$
- a **local loss function** $L_i(\cdot)$ at each node $i \in \mathcal{V}$
- a set of undirected **edges**, denoted as $\mathcal{E}$
- a positive **edge weight** $A_{i,i'} > 0$ for each edge $\{i, i'\} \in \mathcal{E}$

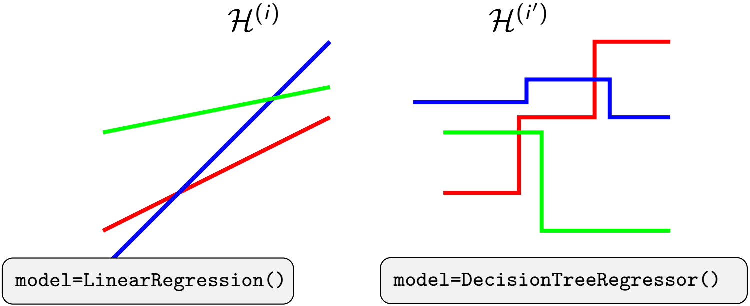
We represent the nodes $\mathcal{V}$, edges $\mathcal{E}$, and edge weights $A_{i,i'}$ of the FL network as an **undirected weighted graph** $\mathcal{G}$.

Nodes of an FL Network

- Consider an FL system with finite number n of devices.
- We index devices as $i = 1, \ldots, n$.
- These indices form the set of nodes $\mathcal{V}$ in an FL network.
- Each node $i \in \mathcal{V}$ **represents** a physical device.
- We use "device i" and "node i" interchangeably.

Local models

- Consider an FL system with devices $i = 1, \ldots, n$.
- Each device trains local (personal) model $\mathcal{H}^{(i)}$.
- Devices might use (very) different local models.
- We use local model parameters $\mathbf{w}^{(i)}$ for parametric $\mathcal{H}^{(i)}$.



Local Loss functions

- Consider device i, training its local model $\mathcal{H}^{(i)}$.
- *To train a model* is to learn a useful hypothesis $h^{(i)} \in \mathcal{H}^{(i)}$.
- We measure usefulness of $h^{(i)}$ by a local loss function

$$L_i(\cdot) : \mathcal{H}^{(i)} \rightarrow \mathbb{R} : h^{(i)} \mapsto L_i\left(h^{(i)}\right)$$

- Different devices might use different loss functions.

Local Loss Functions of an FL Network - ctd.

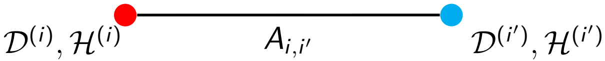
- FL methods use different constructions of loss funcs.
- for param. models $\mathcal{H}^{(i)}$, with parameters $\mathbf{w}^{(i)} \in \mathbb{R}^d$, use
$$L_i(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R} : \mathbf{w}^{(i)} \mapsto L_i\left(\mathbf{w}^{(i)}\right)$$
- can use average loss on local dataset
$$L_i\left(\mathbf{w}^{(i)}\right) := \frac{1}{m_i} \sum_{r=1}^{m_i} \left(y^{(i,r)} - \left(\mathbf{w}^{(i)}\right)^T \mathbf{x}^{(i,r)}\right)^2$$
- use reward signals to estimate loss (federated reinf. learning)

Edges of an FL network

- FL network consists of **undirected weighted** edges $\mathcal{E}$.
- $\{i, i'\} \in \mathcal{E}$ signifies a **similarity** between devices i and i'.
- We **quantify similarity using edge weight** $A_{i,i'} > 0$.
- Notion of similarity depends on FL application .
- We will primarily treat edges as a **design choice**.

Effect of Placing an Edge

We will design FL algorithms that are based on an FL network.



Placing an edge $\{i, i'\} \in \mathcal{E}$ between devices i, i' has two consequences on FL algorithms:

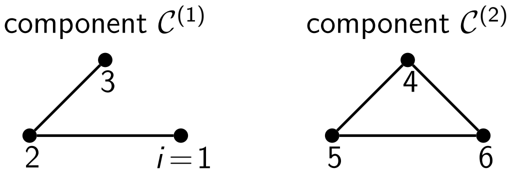
- We must communicate results of computations between devices i, i' ($A_{i,i'} \approx$ channel capacity).
- The local models at i, i' are forced to be similar.

Connectivity of an FL Network

Consider an FL network with graph $\mathcal{G}$. We define:

- $\mathcal{G}$ is **connected** if there is a path between any $i, i' \in \mathcal{V}$.
- A **component** $\mathcal{C} \subseteq \mathcal{V}$ is a connected subgraph with no edges between $\mathcal{C}$ and $\mathcal{V} \setminus \mathcal{C}$.
- The **neighborhood** of $i \in \mathcal{V}$ is $\mathcal{N}^{(i)} := \{i' \in \mathcal{V} : \{i, i'\} \in \mathcal{E}\}$.
- The **weighted node degree** of i is $d^{(i)} := \sum_{i' \in \mathcal{N}^{(i)}} A_{i,i'}$.
- The **maximum node degree** is $d_{\rm max} := \max_{i \in \mathcal{V}} d^{(i)}$.

Connectivity of an FL Network - Example

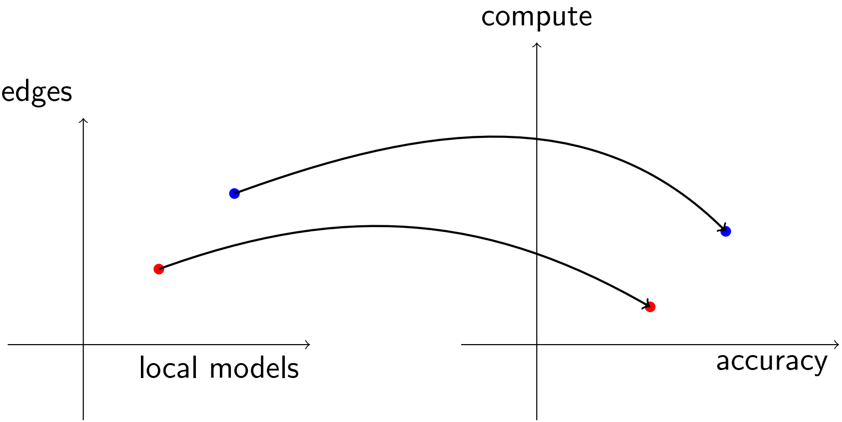


- FL network with graph $\mathcal{G}$ containing $n=6$ nodes.
- Uniform edge-weights, $A_{i,i'}=1$ for all $\{i,i'\}\in\mathcal{E}$.
- Two components $\mathcal{C}^{(1)}=\{1,2,3\},\mathcal{C}^{(2)}=\{4,5,6\}$.
- $d^{(1)}=1$, $\mathcal{N}^{(2)}=\{1,3\}$, $d_{\max}=2$.

Design Choices

- Each FL network involves design choices for
  - **Nodes.** Which devices should be included?
  - **Local models and loss functions.** What type of models should devices use, and how should we evaluate them?
  - **Edges.** Which devices should be collaborating and to what extent?
- These choices determine the **computational and statistical properties** of FL algorithms.
- Trade-offs between **comp. complexity, accuracy, robustness, explainability, and privacy-prot.**

Design Space and Objectives



FL network

We represent a FL system by a weighted undirected graph

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A}),$$

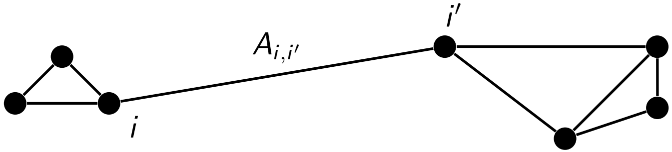
where

- nodes $i \in \mathcal{V}$ represent participating devices
- edges $\{i, i'\} \in \mathcal{E}$ indicate possible collaboration,
- edge weight $A_{i,i'} \geq 0$ quantifies amount of collaboration

Unless stated otherwise,

- $A_{i,i} = 0$ for all $i \in \mathcal{V}$ (no self-loops)
- and $\mathcal{V} = \{1, 2, \ldots, n\}$ with some $n \in \mathbb{N}$ (finiteness).

A Small FL network



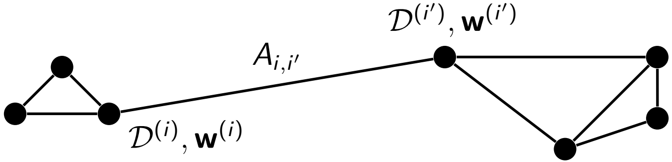
From FL network to FL system

Each node $i \in \mathcal{V}$,

- can access local dataset $\mathcal{D}^{(i)}$,
- maintains model parameters $\mathbf{w}^{(i)}$
- sends/receives messages from neighbors $\mathcal{N}^{(i)}$.

An FL algorithm specifies *when* and *how* these model parameters are updated.

Connected Nodes have Similar Model parameters



We use a large edge weight $A_{i,i'}$, to

- enforce similar local model parameters $\mathbf{w}^{(i)}, \mathbf{w}^{(i')}$
- reflect similarity between local datasets $\mathcal{D}^{(i)}, \mathcal{D}^{(i')}$

L2 "FL Design Principle" discusses graph learning/design.

FL Algorithms

Each node i uses current model parameters $\mathbf{w}^{(1,t)}, \ldots, \mathbf{w}^{(n,t)}$ to compute new model parameters $\mathbf{w}^{(i,t+1)}$,

$$\mathbf{w}^{(i,t+1)} = \mathcal{F}^{(i)}\big(\mathbf{w}^{(1,t)}, \ldots, \mathbf{w}^{(n,t)}\big) \text{ at time instants } t = 0, 1, \ldots.$$

The node-wise operator $\mathcal{F}^{(i)}$ involves

- local model updates (e.g., via gradient steps)
- sharing model parameters across edges of FL network.

What's Next?

L2-"FL Design Principle" introduces generalized total variation minimization (GTVMin) as our main design principle for FL algorithms.

‘

We use GTVMin to guess useful choices for the node-wise update operator $\mathcal{F}^{(i)}$ that define an FL algorithm.

References

- AJ, “Machine Learning: The Basics,” Springer, 2022.
- AJ, “Federated Learning: From Theory to Practice,” Springer, 2026.
- AJ et.al., “The Aalto Dictionary of Machine Learning,” github repo, 2026.