FEDERATED LEARNING

DEFINITION

"**Federated Learning (FL)** is a machine learning technique that enables multiple entities to collaboratively learn a shared model **without exchanging their local data**."

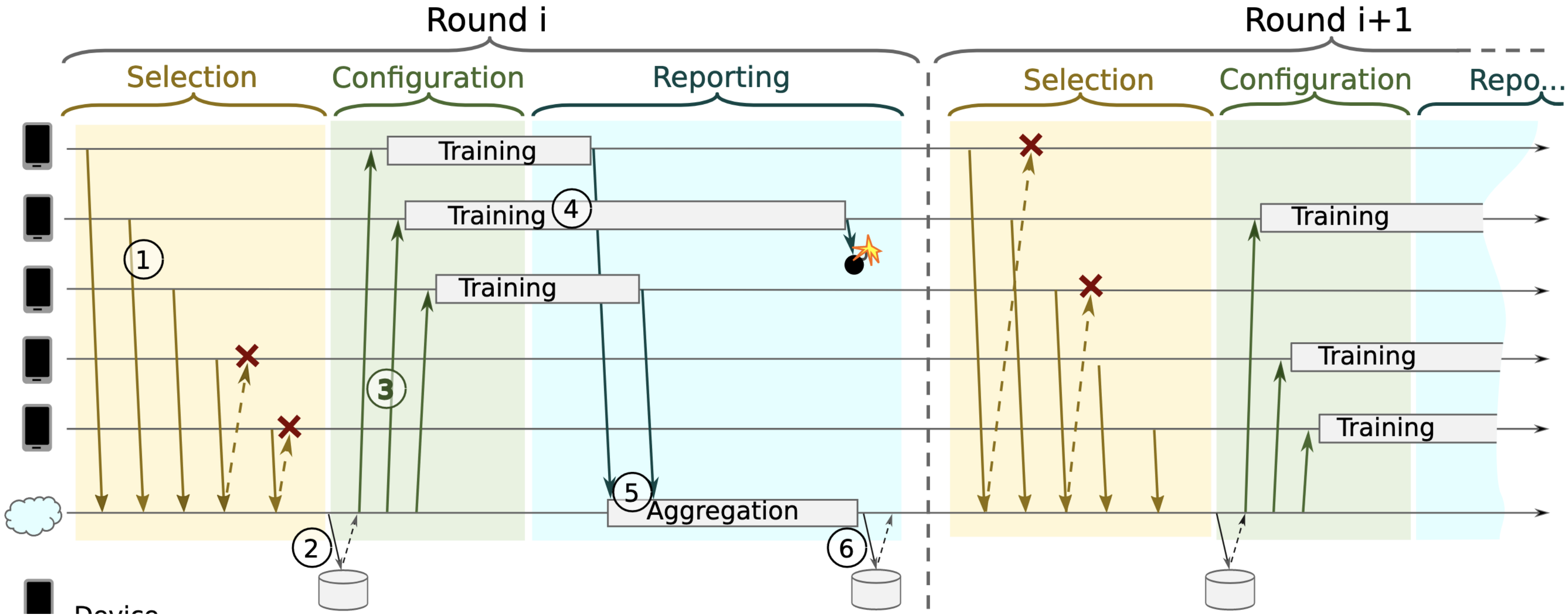
Daly, Katharine, et al. *Federated Learning in Practice: Reflections and Projections*. arXiv:2410.08892, arXiv, 11 Oct. 2024. *arXiv.org*, https://doi.org/10.48550/arXiv.2410.08892.

MOTIVATION

Federated learning may be applicable when:

- Training data cannot be shared directly due to privacy concerns.
- Decentralized compute is available to use for training.
    - Applicable to both high- and low-end compute.

GENERIC PROTOCOL



Bonawitz, Keith, et al. *Towards Federated Learning at Scale: System Design*. arXiv:1902.01046, arXiv, 22 Mar. 2019. *arXiv.org*, https://doi.org/10.48550/arXiv.1902.01046.

GENERIC ALGORITHM

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 $t = 1, 2, \ldots$ **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**
      $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
    $m_t \leftarrow \sum_{k \in S_t} n_k$
    $w_{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{m_t} w_{t+1}^k$ // *Erratum*[4]

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

Model is **initialized**, typically with random parameters, sometimes with pre-trained model

Algorithm from: McMahan, H. Brendan, et al. *Communication-Efficient Learning of Deep Networks from Decentralized Data*. arXiv:1602.05629, arXiv, 26 Jan. 2023. *arXiv.org*, https://doi.org/10.48550/arXiv.1602.05629.

GENERIC ALGORITHM

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**

$w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$

$m_t \leftarrow \sum_{k \in S_t} n_k$

$w_{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{m_t} w_{t+1}^k$ *// Erratum*[4]

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

Many "rounds" of training, as clients gather new data

Algorithm from: McMahan, H. Brendan, et al. *Communication-Efficient Learning of Deep Networks from Decentralized Data*. arXiv:1602.05629, arXiv, 26 Jan. 2023. *arXiv.org*, https://doi.org/10.48550/arXiv.1602.05629.

GENERIC ALGORITHM

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 $t = 1, 2, \ldots$ **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**
      $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
    $m_t \leftarrow \sum_{k \in S_t} n_k$
    $w_{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{m_t} w_{t+1}^k$ *// Erratum*[4]

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

A **subset** of clients are selected for training

Algorithm from: McMahan, H. Brendan, et al. *Communication-Efficient Learning of Deep Networks from Decentralized Data*. arXiv:1602.05629, arXiv, 26 Jan. 2023. *arXiv.org*, https://doi.org/10.48550/arXiv.1602.05629.

GENERIC ALGORITHM

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 $t = 1, 2, \ldots$ **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**
      $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
    $m_t \leftarrow \sum_{k \in S_t} n_k$
    $w_{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{m_t} w_{t+1}^k$ // *Erratum*[4]

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

All selected clients are given the **current version** of the model (may be initial, may be from previous rounds)

Algorithm from: McMahan, H. Brendan, et al. *Communication-Efficient Learning of Deep Networks from Decentralized Data*. arXiv:1602.05629, arXiv, 26 Jan. 2023. *arXiv.org*, https://doi.org/10.48550/arXiv.1602.05629.

GENERIC ALGORITHM

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 $t = 1, 2, \ldots$ **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**
      $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
    $m_t \leftarrow \sum_{k \in S_t} n_k$
    $w_{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{m_t} w_{t+1}^k$ // *Erratum*[4]

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

Client trains model on multiple batches of **local data** and returns **trained model** to server.

Algorithm from: McMahan, H. Brendan, et al. *Communication-Efficient Learning of Deep Networks from Decentralized Data*. arXiv:1602.05629, arXiv, 26 Jan. 2023. *arXiv.org*, https://doi.org/10.48550/arXiv.1602.05629.

GENERIC ALGORITHM

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**
      $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
    $m_t \leftarrow \sum_{k \in S_t} n_k$
    $w_{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{m_t} w_{t+1}^k$ *// Erratum*[4]

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

Counting the number of training samples across all clients.

Algorithm from: McMahan, H. Brendan, et al. *Communication-Efficient Learning of Deep Networks from Decentralized Data*. arXiv:1602.05629, arXiv, 26 Jan. 2023. *arXiv.org*, https://doi.org/10.48550/arXiv.1602.05629.

GENERIC ALGORITHM

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**
      $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
    $m_t \leftarrow \sum_{k \in S_t} n_k$
    $w_{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{m_t} w_{t+1}^k$ *// Erratum*[4]

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

Model parameters are **updated** through a **weighted average** of client model parameters.

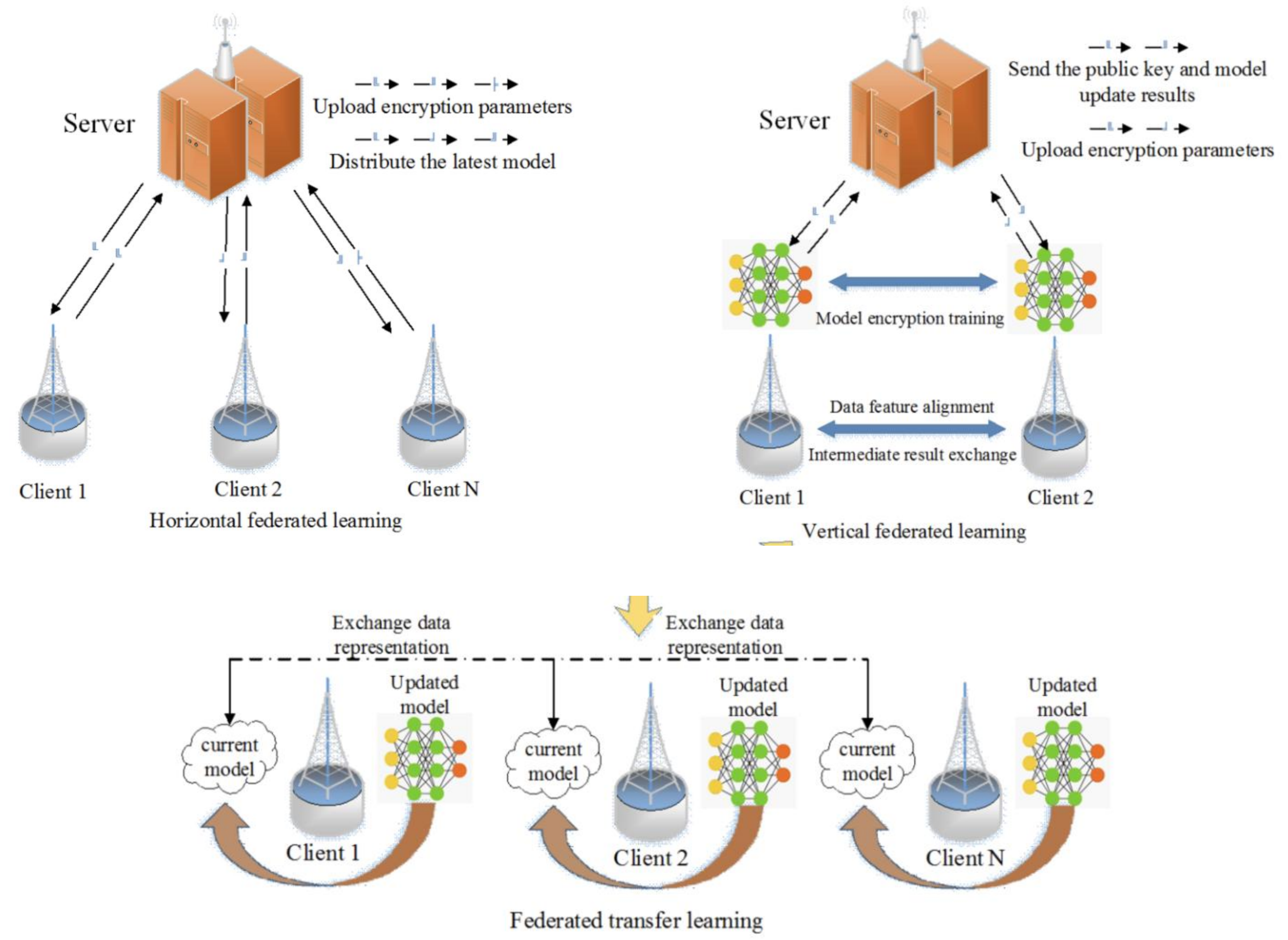
(Weighted average based on number of training samples the client used.)

Algorithm from: McMahan, H. Brendan, et al. *Communication-Efficient Learning of Deep Networks from Decentralized Data*. arXiv:1602.05629, arXiv, 26 Jan. 2023. *arXiv.org*, https://doi.org/10.48550/arXiv.1602.05629.

Federated learning is a **simple** concept with **many** variations.

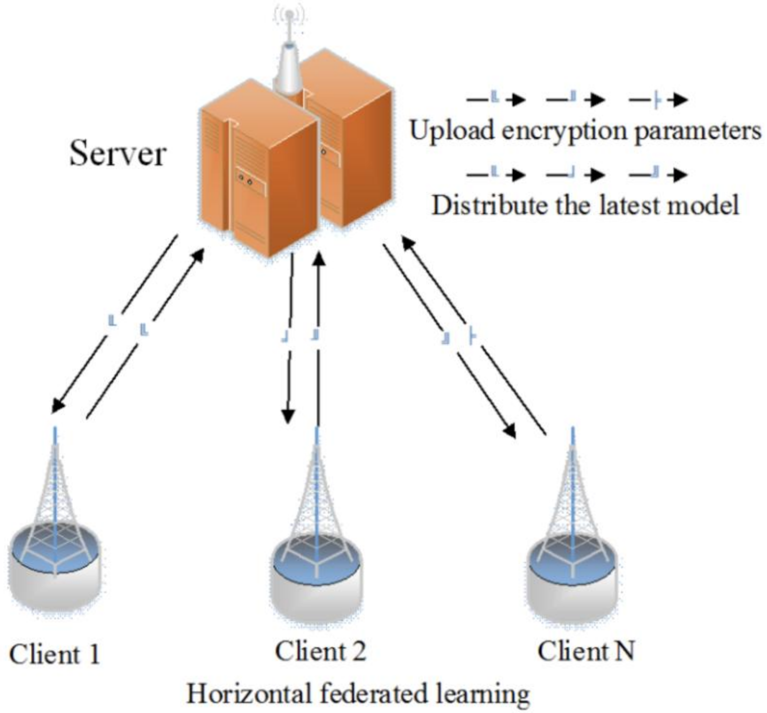
These variations start with the **system architecture itself.**

PARADIGMS



Wen, Jie, et al. “A Survey on Federated Learning: Challenges and Applications.” *International Journal of Machine Learning and Cybernetics*, vol. 14, no. 2, Feb. 2023, pp. 513–35. *DOI.org (Crossref)*, https://doi.org/10.1007/s13042-022-01647-y.

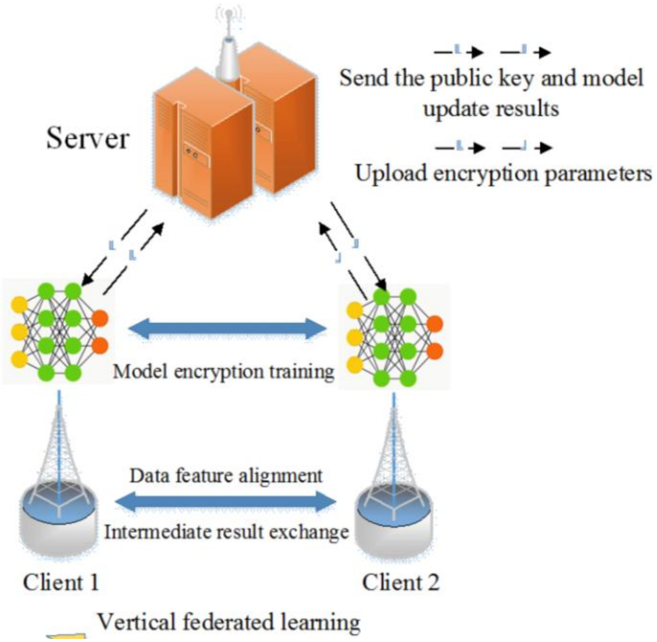
HORIZONTAL FEDERATED LEARNING



- Model sent from server to clients.
- Clients train the model on local data.
- Clients send their models back to the server.
- Server aggregates learning.
- Best for **homogenous data**.

Wen, Jie, et al. “A Survey on Federated Learning: Challenges and Applications.” *International Journal of Machine Learning and Cybernetics*, vol. 14, no. 2, Feb. 2023, pp. 513–35. *DOI.org (Crossref)*, https://doi.org/10.1007/s13042-022-01647-y.

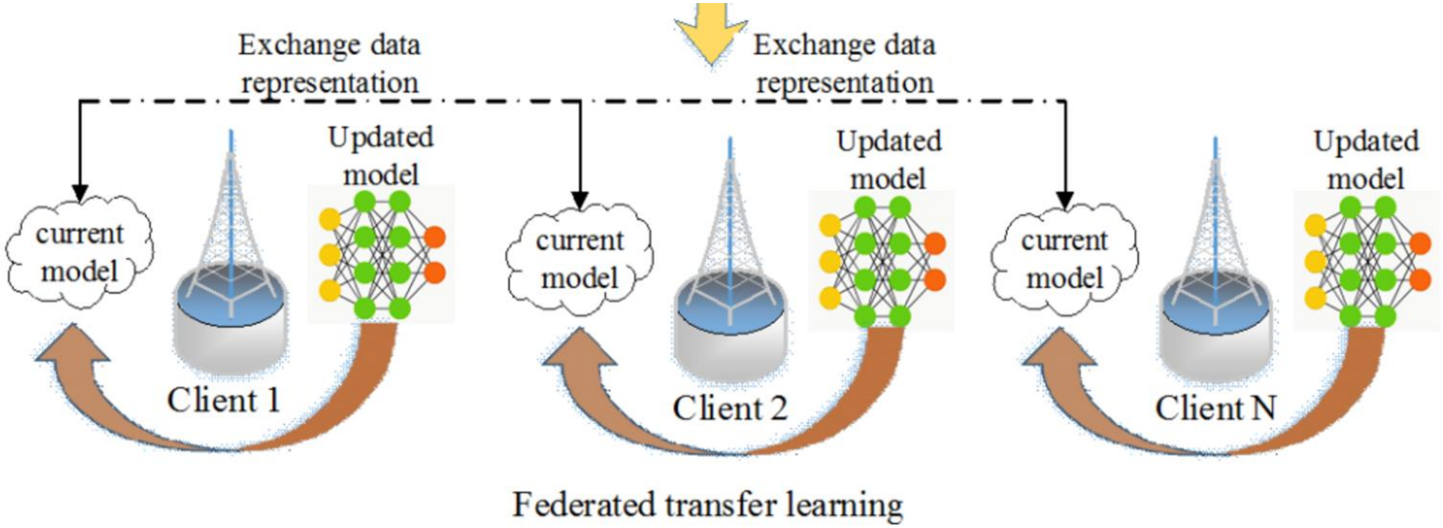
VERTICAL FEDERATED LEARNING



- Model sent from server to clients.
- Clients with **heterogenous data** coordinate feature overlap.
- Joint training with encrypted data.

Wen, Jie, et al. “A Survey on Federated Learning: Challenges and Applications.” *International Journal of Machine Learning and Cybernetics*, vol. 14, no. 2, Feb. 2023, pp. 513–35. *DOI.org (Crossref)*, https://doi.org/10.1007/s13042-022-01647-y.

FEDERATED TRANSFER LEARNING



- Model is passed from client to client and **trained on the way**.
- Can be executed **without** a server/orchestrator.
- Training similar to traditional ML, doesn't require aggregation

Wen, Jie, et al. "A Survey on Federated Learning: Challenges and Applications." *International Journal of Machine Learning and Cybernetics*, vol. 14, no. 2, Feb. 2023, pp. 513–35. *DOI.org (Crossref)*, https://doi.org/10.1007/s13042-022-01647-y.

FURTHER VARIATIONS

- Aggregation
  - Variants on traditional federated averaging
  - Gradient-based
  - Introducing attention
  - Adaptive federated optimization
- Privacy
  - Secure multi-party computation
  - Differential privacy
  - Homomorphic encryption
- Communication Efficiency
  - Model Compression
  - Federated Dropout
  - Structured & Sketched Updates

Additionally: Data/Model Heterogeneity, Client Selection Strategies

THANK YOU