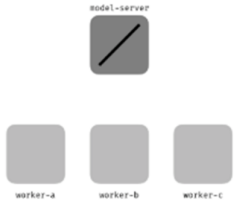
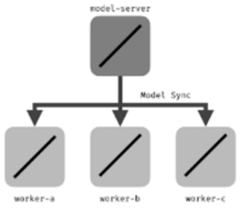
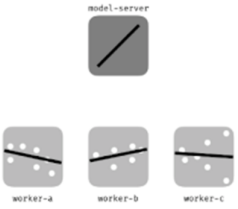
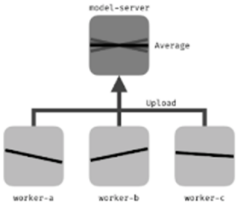
Lecture 9: Distributed Machine Learning

Assoc Prof Tham Chen Khong
Dept of Electrical & Computer Engineering (ECE)
NUS
E-mail: eletck@nus.edu.sg

Federated Learning

- Federated learning is useful when individual actors need to train models on larger datasets than their own, but cannot afford to share its data with others due to data privacy and data locality requirements (e.g. for legal, strategic or economic reasons).
- Requires good connections between local servers and minimum computational power for each node.

Overview

Step 1	Step 2	Step 3	Step 4
			
Central server chooses a statistical model to be trained	Central server transmits the initial model to several nodes	Nodes train the model locally with their own data	Central server pools model results and generate one global mode without accessing any data

source: Wikipedia

Challenges: (1) Non-Independent and Identically Distributed (Non-IID) data:
- local characteristics cannot represent global traits
(2) Communication costs, and computation capability on local nodes

Federated Learning

A formal definition of FL is presented as follows. Consider that there are N parties $\{\mathcal{F}_i\}_{i=1}^{N}$ interested in establishing and training a cooperative ML model. Each party has their respective datasets $\mathcal{D}_i$. Traditional ML approaches consist of collecting all data $\{\mathcal{D}_i\}_{i=1}^{N}$ together to form a centralized dataset $\mathbb{R}$ at one data server. The expected model $\mathcal{M}_{SUM}$ is trained by using the dataset $\mathbb{R}$. On the other hand, FL is a reform of ML process in which the participants $\mathcal{F}_i$ with data $\mathcal{D}_i$ jointly train a target model $\mathcal{M}_{FED}$ without aggregating their data. Respective data $\mathcal{D}_i$ is stored on the owner $\mathcal{F}_i$ and not exposed to others. In addition, the performance measure of the federated model $\mathcal{M}_{FED}$ is denoted as $\mathcal{V}_{FED}$, including accuracy, recall, and F1-score, *etc*, which should be a good approximation of the performance of the expected model $\mathcal{M}_{SUM}$, *i.e.*, $\mathcal{V}_{SUM}$. In order to quantify differences in performance, let δ be a non-negative real number and define the federated learning model $\mathcal{M}_{FED}$ has δ performance loss if

$$|\mathcal{V}_{SUM} - \mathcal{V}_{FED}| < \delta.$$

Specifically, the FL model hold by each party is basically the same as the ML model, and it also includes a set of parameters w_i which is learned based on the respective training dataset $\mathcal{D}_i$ [10]. A training sample j typically contains both the input of FL model and the expected output. For example, in the case of image recognition, the input is the pixel of the image, and the expected output is the correct label. The learning process is facilitated by defining a loss function on parameter vector w for every data sample j. The loss function represents the error of the model in relation to the training data. For each dataset $\mathcal{D}_i$ at party $\mathcal{F}_i$, the loss function on the collection of training samples can be defined as follow [11],

$$F_i(w) = \frac{1}{|\mathcal{D}_i|} \sum_{j \in \mathcal{D}_i} f_j(w),$$

where $f_j(w)$ denotes the loss function of the sample j with the given model parameter vector w and $|\cdot|$ represents the size of the set. In FL, it is important to define the global loss function since multiple parties are jointly training a global statistical model without sharing a dataset. The common global loss function on all the distributed datasets is given by,

$$F_g(w) = \sum_{i=1}^{N} \eta_i F_i(w),$$

where η_i indicates the relative impact of each party on the global model. In addition, $\eta_i > 0$ and $\sum_{i=1}^{N} \eta_i = 1$. This term η can be flexibly defined to improve training efficiency. The natural setting is averaging between parties, *i.e.*, $\eta = 1/N$.

The goal of the learning problem is to find the optimal parameter that minimizes the global loss function $F_g(w)$. It is presented in formula form,

$$w^* = \arg\min_{w} F_g(w).$$

Federated Averaging (FedAvg)
McMahan 2016

Considering that FL is designed to adapt to various scenarios, the objective function may be appropriate depending on the application. However, a closed-form solution is almost impossible to find with most FL models due to their inherent complexity. A canonical federated averaging algorithm (FedAvg) based on gradient-descent techniques is presented in the study from McMahan *et al.* [12], which is widely used in FL systems. In general, the coordinator has the initial FL model and is responsible for aggregation. Distributed participants know the optimizer settings and can upload information that does not affect privacy. The specific architecture of FL will be discussed in the next subsection. Each participant uses their local data to perform one step (or multiple steps) of gradient descent on the current model parameter $\bar{w}(t)$ according to the following formula,

update local weights

$$\forall i, w_i(t+1) = \bar{w}(t) - \gamma \nabla F_i(\bar{w}_i(t)),$$

where γ denotes a fixed learning rate of each gradient descent. After receiving the local parameters from participants, the central coordinator updates the global model using a weighted average, *i.e.*,

update global weights

$$\bar{w}_g(t+1) = \sum_{i=1}^{N} \frac{n_i}{n} w_i(t+1),$$

where n_i indicates the number of training data samples of the i-th participant has and n denotes the total number of samples contained in all the datasets. Finally, the coordinator sends the aggregated model weights $\bar{w}_g(t+1)$ back to the participants. The aggregation process is performed at a predetermined interval or iteration round. Additionally, FL leverages privacy-preserving techniques to prevent the leakage of gradients or model weights.

H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," CoRR, vol. abs/1602.05629, 2016.

FedSGD (prior to FedAvg)

1. Compute local gradients at node i

$$g_i = \nabla F_i(\bar{w}(t))$$

2. All nodes i send g_i to central coordinator
3. Update global model

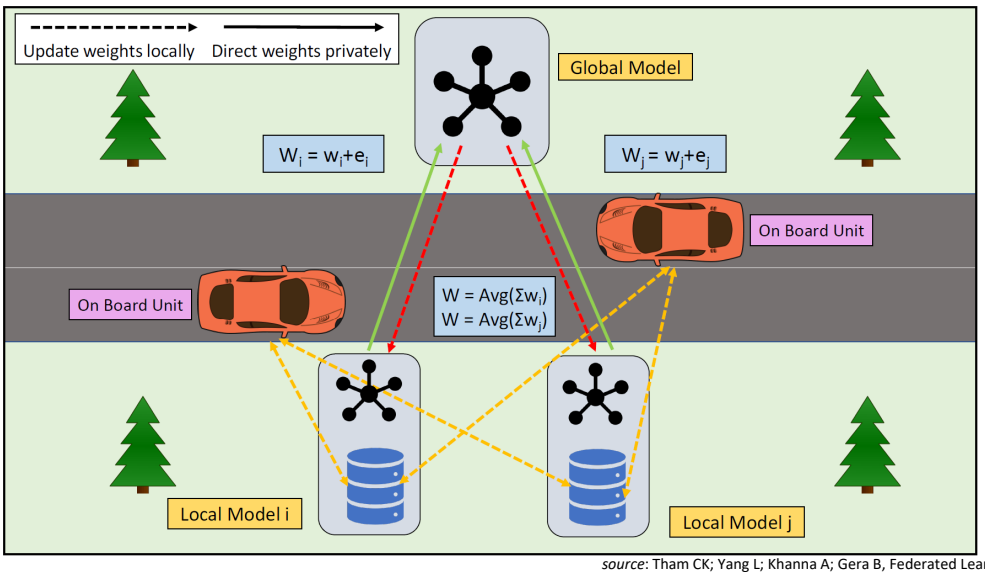
$$\bar{w}_g(t+1) = \bar{w}_g(t) - \gamma \sum_{i=1}^{N} \frac{n_i}{n} g_i$$

4. Send updated global model to all nodes i

Method	Local	Upload	Global	Download
FedSGD	gradient computation	local gradients	gradient averaging	updated model
FedAvg	training	local models	model averaging	updated model

Internet of Vehicles (IoV) example

- e.g. anomaly detection



source: Tham CK; Yang L; Khanna A; Gera B, Federated Learning for Anomaly Detection in Vehicular Networks, in Proceedings of IEEE Vehicular Technology Conference - VTCSpring 2023, Florence, Italy, Jun 2023

FedAvg
with local SGD

Algorithm 1 Federated Averaging with SGD

Server:
Initialize weights W
for each Round t in 1 to T **do**
  Selected Clients = RandomSelection(N Clients)
  for each Client k in Selected Devices in parallel **do**
    W_k = ClientUpdate(k, W)
  end for
  $W = 1/k * \sum_{k=1}^{K} W_k$
end for
ClientUpdate(k, **W**):
Batches = Split data D_k into batches of size B
for each Epoch e in 1 to E **do**
  for each Batch b in Batches **do**
    Compute error gradient ∇_W
    $W = W - \eta * \nabla_W$
  end for
end for
return W

source: Khanna & Tham, submitted, 2022

Federated Learning: Client-Server Model

- Step 1: In the process of setting up a client-server-based learning system, the coordinator creates an initial model and sends it to each participant. Those participants who join later can access the latest global model.
- Step 2: Each participant trains a local model based on their respective dataset.
- Step 3: Updates of model parameters are sent to the central coordinator.
- Step 4: The coordinator combines the model updates using specific aggregation algorithms.
- Step 5: The combined model is sent back to the corresponding participant.

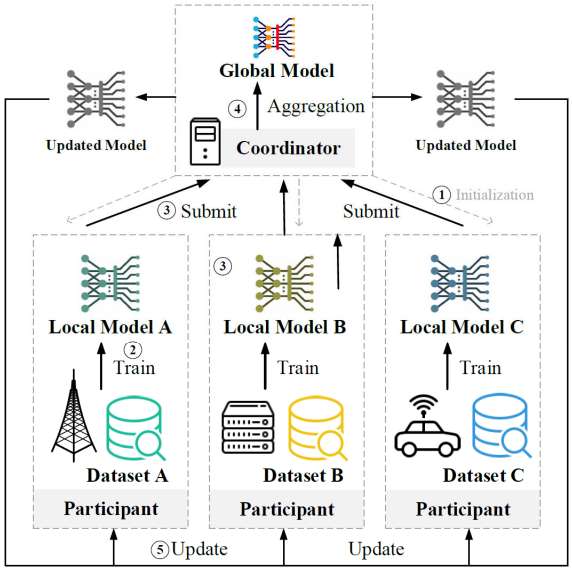


Fig. 1. An Example of Federated Learning Architecture: Client-Server Model.

Federated Learning: Peer-to-Peer Model

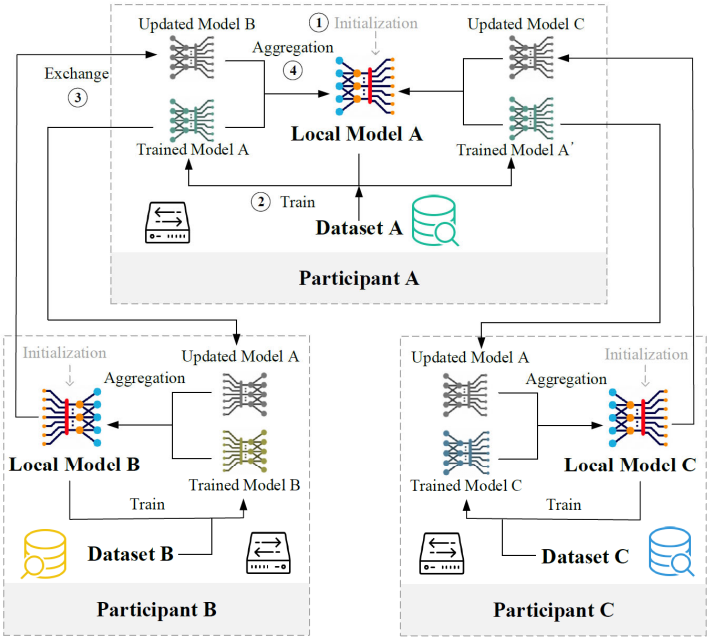
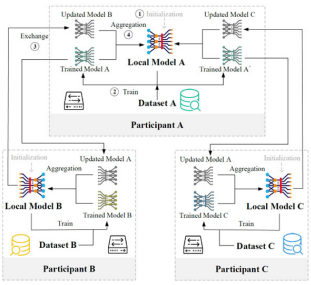


Fig. 2. An Example of Federated Learning Architecture: Peer-to-Peer Model.

Federated Learning: Peer-to-Peer Model



The peer-to-peer based FL architecture does not require a coordinator as illustrated in Figure 2. Participants can directly communicate with each other without relying on a third party. Therefore, each participant in the architecture is equal and can initiate a model exchange request with anyone else. As there is no central server, participants must agree in advance on the order in which model should be sent and received. Common transfer modes are cyclic transfer and random transfer. The peer-to-peer model is suitable and important for specific scenarios. For example, multiple banks jointly develop an ML-based attack detection model. With FL, there is no need to establish a central authority between banks to manage and store all attack patterns. The attack record is only held at the server of the attacked bank, but the detection experience can be shared with other participants through model parameters. The FL procedure of the peer-to-peer model is simpler than that of the client-server model.

- Step 1: Each participant initializes their local model depending on its needs.
- Step 2: Train the local model based on the respective dataset.
- Step 3: Create a model exchange request to other participants and send local model parameters.
- Step 4: Aggregate the model received from other participants into the local model.

The termination conditions of the process can be designed by participants according to their needs. This architecture further guarantees security since there is no centralized server. However, it requires more communication resources and potentially increased computation for more messages.

Thank You

Questions?

Assoc Prof Tham Chen Khong

E-mail: eletck@nus.edu.sg