Title: Federated learning

URL: https://en.wikipedia.org/wiki/Federated_learning

PageID: 60992857

Categories: Category:Distributed artificial intelligence, Category:Machine learning

---

Federated learning (also known as collaborative learning) is a machine learning technique in a setting where multiple entities (often called clients) collaboratively train a model while keeping their data decentralized , rather than centrally stored. A defining characteristic of federated learning is data heterogeneity . Because client data is decentralized, data samples held by each client may not be independently and identically distributed .

Federated learning is generally concerned with and motivated by issues such as data privacy , data minimization , and data access rights. Its applications involve a variety of research areas including defence , telecommunications , the Internet of things , and pharmaceuticals .

Definition

Federated learning aims at training a machine learning algorithm, for instance deep neural networks , on multiple local datasets contained in local nodes without explicitly exchanging data samples. The general principle consists in training local models on local data samples and exchanging parameters (e.g. the weights and biases of a deep neural network) between these local nodes at some frequency to generate a global model shared by all nodes.

The main difference between federated learning and distributed learning lies in the assumptions made on the properties of the local datasets, as distributed learning originally aims at parallelizing computing power where federated learning originally aims at training on heterogeneous datasets . While distributed learning also aims at training a single model on multiple servers, a common underlying assumption is that the local datasets are independent and identically distributed (i.i.d.) and roughly have the same size. None of these hypotheses are made for federated learning; instead, the datasets are typically heterogeneous and their sizes may span several orders of magnitude. Moreover, the clients involved in federated learning may be unreliable as they are subject to more failures or drop out since they commonly rely on less powerful communication media (i.e. Wi-Fi) and battery-powered systems (i.e. smartphones and IoT devices) compared to distributed learning where nodes are typically datacenters that have powerful computational capabilities and are connected to one another with fast networks.

Mathematical formulation

The objective function for federated learning is as follows:

f (x 1 , … , x K) = 1 K ∑ i = 1 K f i (x i) {\displaystyle f(\mathbf {x} _{1},\dots ,\mathbf {x} _{K})={\dfrac {1}{K}}\sum _{i=1}^{K}f_{i}(\mathbf {x} _{i})}

where K {\displaystyle K} is the number of nodes, x i {\displaystyle \mathbf {x} _{i}} are the weights of model as viewed by node i {\displaystyle i} , and f i {\displaystyle f_{i}} is node i {\displaystyle i} 's local objective function, which describes how model weights x i {\displaystyle \mathbf {x} _{i}} conforms to node i {\displaystyle i} 's local dataset.

The goal of federated learning is to train a common model on all of the nodes' local datasets, in other words:

Optimizing the objective function f (x 1 , … , x K) {\displaystyle f(\mathbf {x} _{1},\dots ,\mathbf {x} _{K})} .

Achieving consensus on x i {\displaystyle \mathbf {x} _{i}} . In other words, x 1 , … , x K {\displaystyle \mathbf {x} _{1},\dots ,\mathbf {x} _{K}} converge to some common x {\displaystyle \mathbf {x} } at the end of the training process.

Centralized federated learning

In the centralized federated learning setting, a central server is used to orchestrate the different steps of the algorithms and coordinate all the participating nodes during the learning process. The server is responsible for the nodes selection at the beginning of the training process and for the aggregation of the received model updates. Since all the selected nodes have to send updates to a single entity, the server may become a bottleneck of the system.

Decentralized federated learning

In the decentralized federated learning setting, the nodes are able to coordinate themselves to obtain the global model. This setup prevents single point failures as the model updates are exchanged only between interconnected nodes without the orchestration of the central server. Nevertheless, the specific network topology may affect the performances of the learning process. See blockchain-based federated learning and the references therein.

Heterogeneous federated learning

An increasing number of application domains involve a large set of heterogeneous clients, e.g., mobile phones and IoT devices. Most of the existing federated learning strategies assume that local models share the same global model architecture. Recently, a new federated learning framework named HeteroFL was developed to address heterogeneous clients equipped with very different computation and communication capabilities. The HeteroFL technique can enable the training of heterogeneous local models with dynamically varying computation and non- IID data complexities while still producing a single accurate global inference model.

Main features

Iterative learning

The iterative process of federated learning is composed of a series of fundamental client-server interactions, each of which is known as a federated learning round. Each round of this process consists in transmitting the current global model state to participating nodes, training local models on these local nodes to produce a set of potential model updates at each node, and then aggregating and processing these local updates into a single global update and applying it to the global model.

In the methodology below, a central server is used for aggregation, while local nodes perform local training depending on the central server's orders. However, other strategies lead to the same results without central servers, in a peer-to-peer approach, using gossip or consensus methodologies.

Assuming a federated round composed by one iteration of the learning process, the learning procedure can be summarized as follows:

Initialization : according to the server inputs, a machine learning model (e.g., linear regression , neural network, boosting) is chosen to be trained on local nodes and initialized. Then, nodes are activated and wait for the central server to give the calculation tasks.

Client selection : a fraction of local nodes are selected to start training on local data. The selected nodes acquire the current statistical model while the others wait for the next federated round.

Configuration : the central server orders selected nodes to undergo training of the model on their local data in a pre-specified fashion (e.g., for some mini-batch updates of gradient descent).

Reporting : each selected node sends its local model to the server for aggregation. The central server aggregates the received models and sends back the model updates to the nodes. It also handles failures for disconnected nodes or lost model updates. The next federated round is started returning to the client selection phase.

Termination : once a pre-defined termination criterion is met (e.g., a maximum number of iterations is reached or the model accuracy is greater than a threshold) the central server aggregates the updates and finalizes the global model.

The procedure considered before assumes synchronized model updates. Recent federated learning developments introduced novel techniques to tackle asynchronicity during the training process, or training with dynamically varying models. Compared to synchronous approaches where local models are exchanged once the computations have been performed for all layers of the neural network, asynchronous ones leverage the properties of neural networks to exchange model updates as soon as the computations of a certain layer are available. These techniques are also commonly referred to as split learning and they can be applied both at training and inference time regardless of centralized or decentralized federated learning settings.

Non-IID data

In most cases, the assumption of independent and identically distributed samples across local nodes does not hold for federated learning setups. Under this setting, the performances of the training process may vary significantly according to the unbalanced local data samples as well as the particular probability distribution of the training examples (i.e., features and labels) stored at the local nodes. To further investigate the effects of non-IID data, the following description considers the main categories presented in the preprint by Peter Kairouz et al. from 2019.

The description of non-IID data relies on the analysis of the joint probability between features and labels for each node.

This allows decoupling of each contribution according to the specific distribution available at the local nodes.

The main categories for non-iid data can be summarized as follows:

Covariate shift : local nodes may store examples that have different statistical distributions compared to other nodes. An example occurs in natural language processing datasets where people typically write the same digits/letters with different stroke widths or slants.

Prior probability shift : local nodes may store labels that have different statistical distributions compared to other nodes. This can happen if datasets are regional and/or demographically partitioned. For example, datasets containing images of animals vary significantly from country to country.

Concept drift (same label, different features): local nodes may share the same labels but some of them correspond to different features at different local nodes. For example, images that depict a particular object can vary according to the weather condition in which they were captured.

Concept shift (same features, different labels): local nodes may share the same features but some of them correspond to different labels at different local nodes. For example, in natural language processing, the sentiment analysis may yield different sentiments even if the same text is observed.

Unbalanced : the amount of data available at the local nodes may vary significantly in size.

The loss in accuracy due to non-iid data can be bounded through using more sophisticated means of doing data normalization, rather than batch normalization.

Algorithmic hyper-parameters

Network topology

The way the statistical local outputs are pooled and the way the nodes communicate with each other can change from the centralized model explained in the previous section. This leads to a variety of federated learning approaches: for instance no central orchestrating server, or stochastic communication.

In particular, orchestrator-less distributed networks are one important variation. In this case, there is no central server dispatching queries to local nodes and aggregating local models. Each local node sends its outputs to several randomly-selected others, which aggregate their results locally. This restrains the number of transactions, thereby sometimes reducing training time and computing cost.

Federated learning parameters

Once the topology of the node network is chosen, one can control different parameters of the federated learning process (in addition to the machine learning model's own hyperparameters) to optimize learning:

Number of federated learning rounds: T {\displaystyle T}

Total number of nodes used in the process: K {\displaystyle K}

Fraction of nodes used at each iteration for each node: C {\displaystyle C}

Local batch size used at each learning iteration: B {\displaystyle B}

Other model-dependent parameters can also be tinkered with, such as:

Number of iterations for local training before pooling: N {\displaystyle N}

Local learning rate: η {\displaystyle \eta }

Those parameters have to be optimized depending on the constraints of the machine learning application (e.g., available computing power, available memory, bandwidth). For instance, stochastically choosing a limited fraction C {\displaystyle C} of nodes for each iteration diminishes computing cost and may prevent overfitting [citation needed] , in the same way that stochastic gradient descent can reduce overfitting.

Technical limitations

Federated learning requires frequent communication between nodes during the learning process. Thus, it requires not only enough local computing power and memory, but also high bandwidth connections to be able to exchange parameters of the machine learning model. However, the technology also avoids data communication , which can require significant resources before starting centralized machine learning. Nevertheless, the devices typically employed in federated learning are communication-constrained, for example IoT devices or smartphones are generally connected to Wi-Fi networks, thus, even if the models are commonly less expensive to be transmitted compared to raw data, federated learning mechanisms may not be suitable in their general form.

Federated learning raises several statistical challenges:

Heterogeneity between the different local datasets: each node may have some bias with respect to the general population, and the size of the datasets may vary significantly;

Temporal heterogeneity: each local dataset's distribution may vary with time;

Interoperability of each node's dataset is a prerequisite;

Each node's dataset may require regular curations;

Hiding training data might allow attackers to inject backdoors into the global model;

Lack of access to global training data makes it harder to identify unwanted biases entering the training e.g. age, gender, sexual orientation;

Partial or total loss of model updates due to node failures affecting the global model;

Lack of annotations or labels on the client side.

Heterogeneity between processing platforms

Variations

A number of different algorithms for federated optimization have been proposed.

Federated stochastic gradient descent (FedSGD)

Stochastic gradient descent is an approach used in deep learning , where gradients are computed on a random subset of the total dataset and then used to make one step of the gradient descent..

Federated stochastic gradient descent is the analog of this algorithm to the federated setting, but uses a random subset of the nodes, each node using all its data. The server averages the gradients in proportion to the number of training data on each node, and uses the average to make a gradient

descent step.

Federated averaging (FedAvg)

Federated averaging (FedAvg) is a generalization of FedSGD which allows nodes to do more than one batch update on local data and exchange updated weights rather than gradients. This reduces communication and is equivalent to averaging the weights if all nodes start with the same weights. It does not seem to hurt the resulting averaged model's performance compared to FedSGD. FedAvg variations have been proposed based on adaptive optimizers such as ADAM and AdaGrad , and tend to outperform FedAvg.

Federated learning with dynamic regularization (FedDyn)

Federated learning methods suffer when node datasets are distributed heterogeneously, because then minimizing the node losses is not the same as minimizing the global loss. In 2021, Acar et al. introduced a solution called FedDyn, which dynamically regularizes each node loss function so that they converge to the global loss. Since the local losses are aligned, FedDyn is robust to the different heterogeneity levels and so it can safely perform full minimization in each device. In theory, FedDyn converges to the optimal (a stationary point for nonconvex losses) by being agnostic to the heterogeneity levels. These claims are verified with extensive experiments on various datasets.

Besides reducing communication, it is also beneficial to reduce local computation. To do this, FedDynOneGD modifies FedDyn to calculate only one gradient per node per round, regularizes it and updates the global model with it. Hence, the computational complexity is linear in local dataset size. Moreover, gradient computation can be parallelized on each node, unlike successive SGD steps. In theory, FedDynOneGD achieves the same convergence guarantees as in FedDyn with less local computation.

Personalized federated learning by pruning (Sub-FedAvg)

Federated learning methods have poor global performance under non-IID settings. This motivates clients to yield personalized models in federation. To change this, Vahidian et al. recently introduced the algorithm Sub-FedAvg which does hybrid pruning (structured and unstructured pruning) with averaging on the intersection of clients' drawn subnetworks. This simultaneously addresses communication efficiency, resource constraints and personalized models accuracies.

Sub-FedAvg also extends the " lottery ticket hypothesis " of centrally trained neural networks to federated learning, with the research question: “Do winning tickets exist for clients' neural networks being trained in federated learning? If so, how to effectively draw the personalized subnetworks for each client?” Sub-FedAvg experimentally answers "yes" and proposes two algorithms to effectively draw the personalized subnetworks.

Dynamic aggregation - inverse distance aggregation

IDA (Inverse Distance Aggregation) is a novel adaptive weighting approach for federated learning nodes based on meta-information which handles unbalanced and non-iid data. It uses the distance of the model parameters as a strategy to minimize the effect of outliers and improve the model's convergence rate.

Hybrid federated dual coordinate ascent (HyFDCA)

Very few methods for hybrid federated learning, where clients only hold subsets of both features and samples, exist. Yet, this scenario is very important in practical settings. Hybrid Federated Dual Coordinate Ascent (HyFDCA) is a novel algorithm proposed in 2024 that solves convex problems in the hybrid FL setting. This algorithm extends CoCoA, a primal-dual distributed optimization algorithm introduced by Jaggi et al. (2014) and Smith et al. (2017), to the case where both samples and features are partitioned across clients.

HyFDCA claims several improvement over existing algorithms:

HyFDCA is a provably convergent primal-dual algorithm for hybrid FL in at least the following settings. Hybrid Federated Setting with Complete Client Participation Horizontal Federated Setting with Random Subsets of Available Clients The authors show HyFDCA enjoys a convergence rate of $O(1/t)$ which matches the convergence rate of FedAvg (see below). Vertical Federated Setting

with Incomplete Client Participation The authors show HyFDCA enjoys a convergence rate of O (log(t)／t) whereas FedBCD exhibits a slower O (1／sqrt(t)) convergence rate and requires full client participation.

Hybrid Federated Setting with Complete Client Participation

Horizontal Federated Setting with Random Subsets of Available Clients The authors show HyFDCA enjoys a convergence rate of O (1／t) which matches the convergence rate of FedAvg (see below).

The authors show HyFDCA enjoys a convergence rate of O (1／t) which matches the convergence rate of FedAvg (see below).

Vertical Federated Setting with Incomplete Client Participation The authors show HyFDCA enjoys a convergence rate of O (log(t)／t) whereas FedBCD exhibits a slower O (1／sqrt(t)) convergence rate and requires full client participation.

The authors show HyFDCA enjoys a convergence rate of O (log(t)／t) whereas FedBCD exhibits a slower O (1／sqrt(t)) convergence rate and requires full client participation.

HyFDCA provides the privacy steps that ensure privacy of client data in the primal-dual setting. These principles apply to future efforts in developing primal-dual algorithms for FL.

HyFDCA empirically outperforms HyFEM and FedAvg in loss function value and validation accuracy across a multitude of problem settings and datasets (see below for more details). The authors also introduce a hyperparameter selection framework for FL with competing metrics using ideas from multiobjective optimization.

There is only one other algorithm that focuses on hybrid FL, HyFEM proposed by Zhang et al. (2020). This algorithm uses a feature matching formulation that balances clients building accurate local models and the server learning an accurate global model. This requires a matching regularizer constant that must be tuned based on user goals and results in disparate local and global models. Furthermore, the convergence results provided for HyFEM only prove convergence of the matching formulation not of the original global problem. This work is substantially different than HyFDCA's approach which uses data on local clients to build a global model that converges to the same solution as if the model was trained centrally. Furthermore, the local and global models are synchronized and do not require the adjustment of a matching parameter between local and global models. However, HyFEM is suitable for a vast array of architectures including deep learning architectures, whereas HyFDCA is designed for convex problems like logistic regression and support vector machines.

HyFDCA is empirically benchmarked against the aforementioned HyFEM as well as the popular FedAvg in solving convex problem (specifically classification problems) for several popular datasets (MNIST, Covtype, and News20). The authors found HyFDCA converges to a lower loss value and higher validation accuracy in less overall time in 33 of 36 comparisons examined and 36 of 36 comparisons examined with respect to the number of outer iterations. Lastly, HyFDCA only requires tuning of one hyperparameter, the number of inner iterations, as opposed to FedAvg (which requires tuning three) or HyFEM (which requires tuning four). In addition to FedAvg and HyFEM being quite difficult to optimize hyperparameters in turn greatly affecting convergence, HyFDCA's single hyperparameter allows for simpler practical implementations and hyperparameter selection methodologies.

Federated tree■based ensemble methods

While most federated■learning research focuses on gradient■based models, ensemble trees have also been adapted. Cotorobai et al. (2025) introduced a privacy■preserving framework for training Random Forest classifiers across multiple institutions without sharing raw data, achieving predictive performance within 9 % of centralized baselines on healthcare benchmarks .

Current research topics

Federated learning has started to emerge as an important research topic in 2015 and 2016, with the first publications on federated averaging in telecommunication settings. Before that, in a thesis work titled "A Framework for Multi-source Prefetching Through Adaptive Weight", an approach to

aggregate predictions from multiple models trained at three location of a request response cycle with was proposed. Another important aspect of active research is the reduction of the communication burden during the federated learning process. In 2017 and 2018, publications have emphasized the development of resource allocation strategies, especially to reduce communication requirements between nodes with gossip algorithms as well as on the characterization of the robustness to differential privacy attacks. Other research activities focus on the reduction of the bandwidth during training through sparsification and quantization methods, where the machine learning models are sparsified and/or compressed before they are shared with other nodes. Developing ultra-light DNN architectures is essential for device-/edge- learning and recent work recognises both the energy efficiency requirements for future federated learning and the need to compress deep learning, especially during learning.

Recent research advancements are starting to consider real-world propagating channels as in previous implementations ideal channels were assumed. Another active direction of research is to develop Federated learning for training heterogeneous local models with varying computation complexities and producing a single powerful global inference model.

A learning framework named Assisted learning was recently developed to improve each agent's learning capabilities without transmitting private data, models, and even learning objectives. Compared with Federated learning that often requires a central controller to orchestrate the learning and optimization, Assisted learning [dead link] aims to provide protocols for the agents to optimize and learn among themselves without a global model.

Use cases

Federated learning typically applies when individual actors need to train models on larger datasets than their own, but cannot afford to share the data in itself with others (e.g., for legal, strategic or economic reasons). The technology yet requires good connections between local servers and minimum computational power for each node.

Transportation: self-driving cars

Self-driving cars encapsulate many machine learning technologies to function: computer vision for analyzing obstacles, machine learning for adapting their pace to the environment (e.g., bumpiness of the road). Due to the potential high number of self-driving cars and the need for them to quickly respond to real world situations, traditional cloud approach may generate safety risks. Federated learning can represent a solution for limiting volume of data transfer and accelerating learning processes.

Industry 4.0: smart manufacturing

In Industry 4.0 , there is a widespread adoption of machine learning techniques to improve the efficiency and effectiveness of industrial process while guaranteeing a high level of safety. Nevertheless, privacy of sensitive data for industries and manufacturing companies is of paramount importance. Federated learning algorithms can be applied to these problems as they do not disclose any sensitive data. In addition, FL also implemented for PM2.5 prediction to support Smart city sensing applications.

Medicine: digital health

Federated learning seeks to address the problem of data governance and privacy by training algorithms collaboratively without exchanging the data itself. Today's standard approach of centralizing data from multiple centers comes at the cost of critical concerns regarding patient privacy and data protection. To solve this problem, the ability to train machine learning models at scale across multiple medical institutions without moving the data is a critical technology. Nature Digital Medicine published the paper "The Future of Digital Health with Federated Learning" in September 2020, in which the authors explore how federated learning may provide a solution for the future of digital health, and highlight the challenges and considerations that need to be addressed. Recently, a collaboration of 20 different institutions around the world validated the utility of training AI models using federated learning. In a paper published in Nature Medicine "Federated learning for predicting clinical outcomes in patients with COVID-19", they showcased the accuracy and generalizability of a federated AI model for the prediction of oxygen needs in patients with

COVID-19 infections. Furthermore, in a published paper "A Systematic Review of Federated Learning in the Healthcare Area: From the Perspective of Data Properties and Applications", the authors trying to provide a set of challenges on FL challenges on medical data-centric perspective.

A coalition from industry and academia has developed MedPerf, an open source platform that enables validation of medical AI models in real world data. The platform relies technically on federated evaluation of AI models aiming to alleviate concerns of patient privacy and conceptually on diverse benchmark committees to build the specifications of neutral clinically impactful benchmarks.

Robotics

Robotics includes a wide range of applications of machine learning methods: from perception and decision-making to control. As robotic technologies have been increasingly deployed from simple and repetitive tasks (e.g. repetitive manipulation) to complex and unpredictable tasks (e.g. autonomous navigation), the need for machine learning grows. Federated Learning provides a solution to improve over conventional machine learning training methods. In the paper, mobile robots learned navigation over diverse environments using the FL-based method, helping generalization. In the paper, Federated Learning is applied to improve multi-robot navigation under limited communication bandwidth scenarios, which is a current challenge in real-world learning-based robotic tasks. In the paper, Federated Learning is used to learn Vision-based navigation, helping better sim-to-real transfer.

Biometrics

Federated Learning (FL) is transforming biometric recognition by enabling collaborative model training across distributed data sources while preserving privacy. By eliminating the need to share sensitive biometric templates like fingerprints, facial images, and iris scans, FL addresses privacy concerns and regulatory constraints, allowing for improved model accuracy and generalizability. It mitigates challenges of data fragmentation by leveraging scattered datasets, making it particularly effective for diverse biometric applications such as facial and iris recognition. However, FL faces challenges, including model and data heterogeneity, computational overhead, and vulnerability to security threats like inference attacks. Future directions include developing personalized FL frameworks, enhancing system efficiency, and expanding FL applications to biometric presentation attack detection (PAD) and quality assessment, fostering innovation and robust solutions in privacy-sensitive environments.

References

External links

"Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016" at eur-lex.europa.eu. Retrieved October 18, 2019.

"Data minimisation and privacy-preserving techniques in AI systems" Archived 2020-07-23 at the Wayback Machine at UK Information Commissioners Office. Retrieved July 22, 2020

"Realising the Potential of Data Whilst Preserving Privacy with EyA and Conclave from R3" at eya.global. Retrieved March 31, 2022.