Foundation of Cloud IoT Edge ML

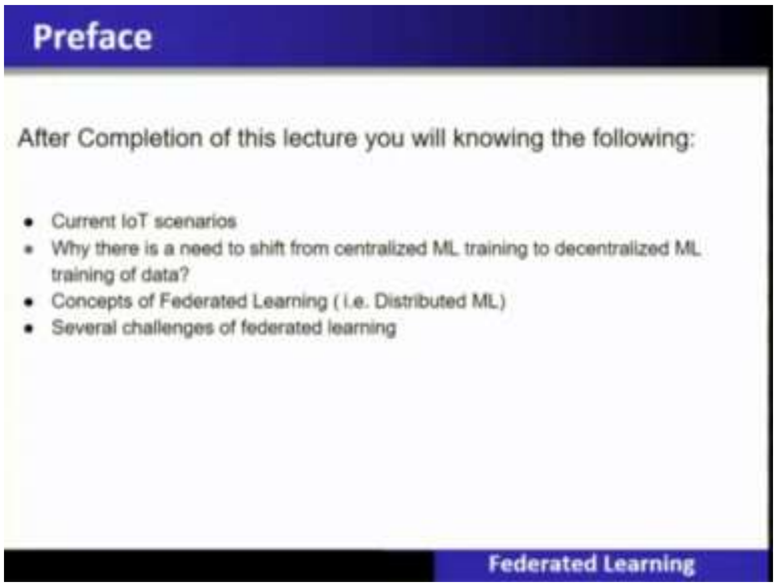
Professor Rajiv Misra
Department of Computer Science and Engineering
Indian Institute of Technology, Patna

Lecture No. 21

Introduction to Federated Learning at IoT Edge

I am Doctor Rajiv Misra from IIT, Patna. The topic of today's lecture is Introduction to Federated Learning at IoT Edge.

(Refer Slide Time: 0:23)



Preface

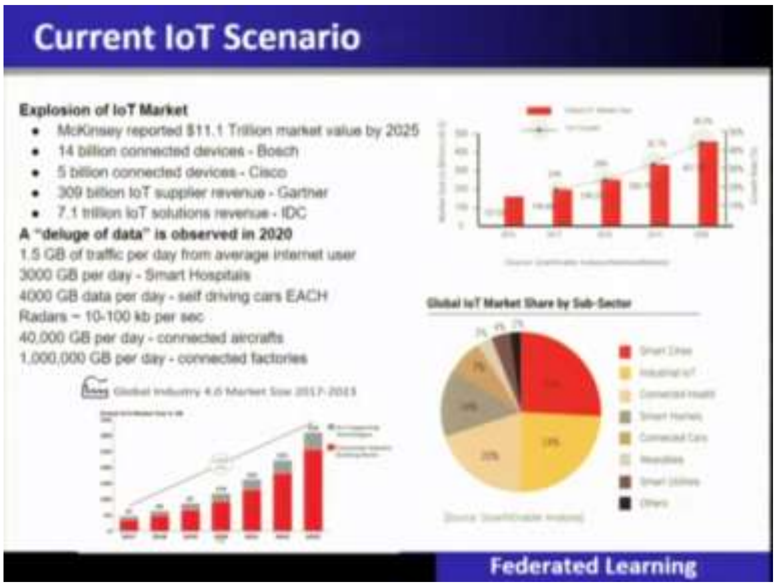
After Completion of this lecture you will knowing the following:

- Current IoT scenarios
- Why there is a need to shift from centralized ML training to decentralized ML training of data?
- Concepts of Federated Learning (i.e. Distributed ML)
- Several challenges of federated learning

Federated Learning

In this lecture we will cover the following details. First is, we will show the current IoT scenarios. Then, why there is a need to shift from centralized machine learning training to decentralize machine learning training of an IoT data. Then, we will also discuss the concepts in this domain, which is called a distributed machine learning. That is a federated learning concepts. Then we will also cover several challenges of federated learning. Besides this, we will also cover few applications of federated learning, which is at the use.

(Refer Slide Time: 1:06)



Current IoT Scenario

Explosion of IoT Market

- McKinsey reported $11.1 Trillion market value by 2025
- 14 billion connected devices - Bosch
- 5 billion connected devices - Cisco
- 309 billion IoT supplier revenue - Gartner
- 7.1 trillion IoT solutions revenue - IDC

A "deluge of data" is observed in 2020

1.5 GB of traffic per day from average internet user
3000 GB per day - Smart Hospitals
4000 GB data per day - self driving cars EACH
Radars ~ 10-100 kb per sec
40,000 GB per day - connected aircrafts
1,000,000 GB per day - connected factories

Global Industry 4.0 Market Size 2017-2023
Global IoT Market Share by Sub-Sector

Federated Learning

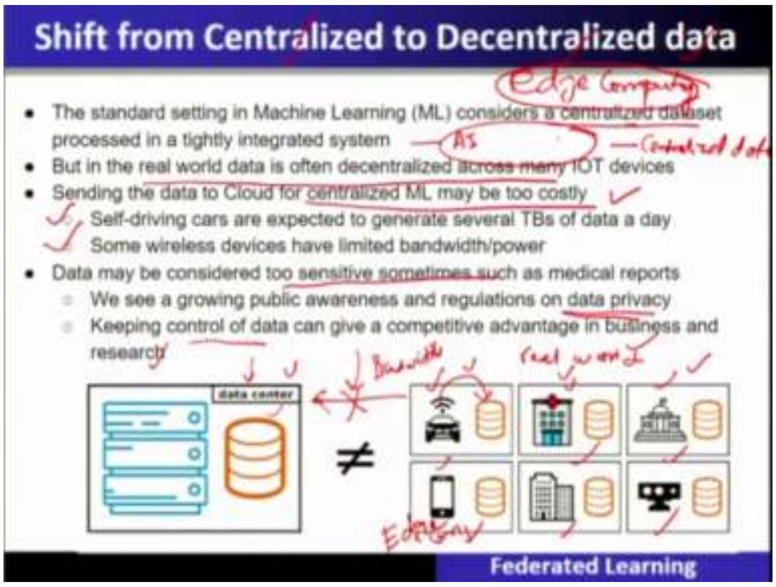
Let us talk about the current IoT scenario. So, there is a explosion of IoT market and various forecasting agencies such as McKinsey reported 11.1 trillion dollar market, the value of an IoT use cases by 2025. Bosch company who is working with IoT technologies. So, they are expecting that 14 billion connected devices will be there. Cisco is expecting that 5 billion connected devices. Gartner also projected 309, 309 billion IoT supplier revenues and IDC is a forecasting company also suggested that 7.1 trillion IoT solution revenue will be exploding this IoT market.

So, this growth, if you see about the global IoT market size that is continuously growing over 2017, that was 24 percent of the US dollar size. Now it has gone in 2020. That is 38.3 percent and that is ever growing, if you go by this forecasting agency like McKinsey. Now there is a deluge of data that is over helm of that particular data which is observed in 2020. So, why this so much of data is being observed that is in this IoT scenario?

So, what we are going to see is 1.5 gigabytes of traffic per day is originated because of this IoT growth from an average internet user. Similarly, with the advent of smart hospitals, this particular deluge of data is 300 gigabytes per day. So, deluge of data is also in the domain, which is expected called self-driving car that is 4,000 gigabytes per day.

Radars also are significantly adding this particular data traffic. Connected aircrafts, connected factories are also able to generate so much of particular data and this all is seen as deluge of data that is observed 2020 and also being forecasted as they are soaring revenues in an IoT market by 2025.

(Refer Slide Time: 3:53)



Shift from Centralized to Decentralized data

- The standard setting in Machine Learning (ML) considers a centralized dataset processed in a tightly integrated system
- But in the real world data is often decentralized across many IOT devices
- Sending the data to Cloud for centralized ML may be too costly
  - Self-driving cars are expected to generate several TBs of data a day
  - Some wireless devices have limited bandwidth/power
- Data may be considered too sensitive sometimes such as medical reports
  - We see a growing public awareness and regulations on data privacy
  - Keeping control of data can give a competitive advantage in business and research

So, let us see what are the key players, what are the key technologies which are also shifting in this particular line to tap this or facilitate this explosion of an IoT market and also to deal with deluge of data. So, the first aspect which we will discuss in this lecture is that we are going to shift from centralized data to decentralize data. Let us understand a particular scenario in this shift.

So, the standard setting in machine learning considers centralized data processed in a tightly integrated system. That is what we have so far seen and that was happening in the cloud. That is AI in the cloud. And therefore, the entire dataset has to be moved to the cloud for performing the machine learning. So, therefore, this is standard setting in machine learning considers the centralized dataset processed in a tightly integrated system. That is, it requires, that is AI required so far centralized data.

But in the real world, this is often decentralized across many IoT devices because the data is generated not at one place, but through various devices which are separated or which are geographically distributed across the globe. And therefore, this particular real world data is decentralized across the globe and that is being generated by an IoT scenario or IoT devices.

Now sending the entire data to the cloud for centralized AI or the machine learning may become too costly to support or to continue this scale which is being projected by an IoT market. For example, the new kind of innovation way which is in the form of in IoT, which is in the form of self-driving car is expected to generate several trillion bytes of data per day. Similarly, new wireless devices also have the limited bandwidth and the power to transfer that particular data.

So, one side of this particular scenario is our growing number of devices, that is IoT devices connecting to the cloud through internet. Whereas, these wireless communication and these devices sometimes have the limited bandwidth or the power. So, data may be considered also sometimes too sensitive such as medical reports or let us say mobile data, which is highly personalized data.

Now if you see this growing public awareness and the regulations on data privacy, it is not easy to transfer the data yet you are going to protect or preserve the privacy without the consent of the people. So, therefore, to perform these artificial intelligence or machine learning, you have to keep the control of the data and can give a competitive edge in the business research if this particular protection is ensure. So, therefore, protection will build the trust of the people's data and yet you are going to use that particular data with a control and now foster the research and the business in this particular scenario.

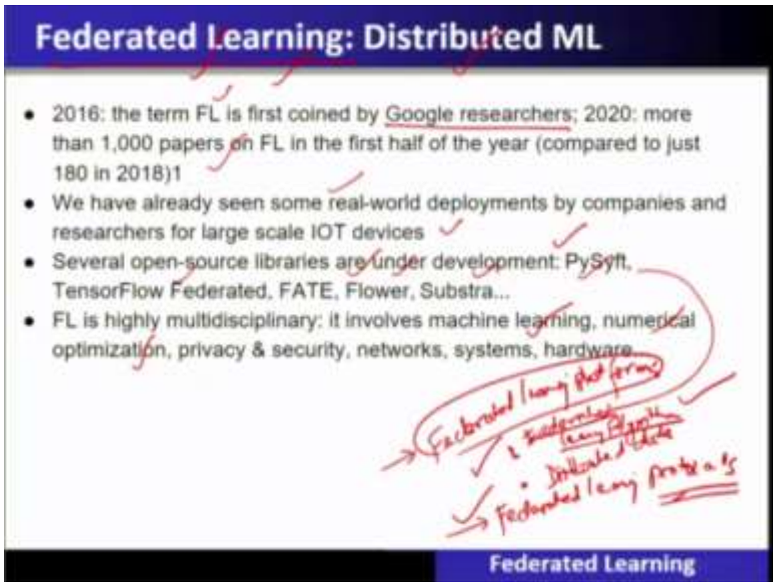
So, here the scenario is shown over here is that there are several IoT devices which are generating the data and often sending to the cloud. That is called the centralized data. But this particular scenario is called centralized data. And centralized data is being consumed and processed in a place of a cloud that is called data centre that we have already covered.

Now, but if you see the real world, real world devices such as connected cars, hospitals, smart hospitals, government organizations, mobile phones carried by the people and so on various other factories where IoT devices are embedded or computers. So, these particular devices, they are IoT market. They are generating the data not at one place but from different places. That is that particular data is decentralized data. Now how do you fill the gap? So, one way for centralized data is to ship all the data towards the cloud, which requires the lot of bandwidth

In a state, what you can do is you can stop tipping the data and perform the artificial intelligence or machine learning on a decentralized data thereby. So, data need not to move heavily on the internet but will be stored close by those devices that is, the source and this is called the edge or a network edge. So, that means the data has to be stored and processed at the network edge and this becomes the concept of edge computing.

So, decentralized data is a need and this particular model of decentralization of that particular data for performing various analytics and applying the machine learning is requires a technology, which is called the edge computing.

(Refer Slide Time: 10:14)



Federated Learning: Distributed ML

- 2016: the term FL is first coined by Google researchers; 2020: more than 1,000 papers on FL in the first half of the year (compared to just 180 in 2018)1
- We have already seen some real-world deployments by companies and researchers for large scale IOT devices
- Several open-source libraries are under development: PySyft, TensorFlow Federated, FATE, Flower, Substra...
- FL is highly multidisciplinary: it involves machine learning, numerical optimization, privacy & security, networks, systems, hardware

Federated learning platforms
• Federated learning algorithms
• Distributed data
Federated learning protocols

Federated Learning

Now let us see in the machine learning paradigm how this distributed data we are going to use that particular model for machine learning. So, that kind of machine learning is called a distributed machine learning in contrast to the centralized data, I am performing the machine learning. So, if the data is distributed and if you are doing the machine learning with a distributed data, then this becomes a distributed machine learning.

We will see one type of machine learning that is in this particular domain that machine learning, which is a distributed machine learning, is done through a very popular model of a machine learning that is called federated learning. So, federated learning algorithms and the protocols both we are going to discuss in this particular lecture. So, the topic of this particular lecture is going to be very important development towards edge computing. So, let us see the details about the federated learning, which is nothing but a concept of a distributed machine learning.

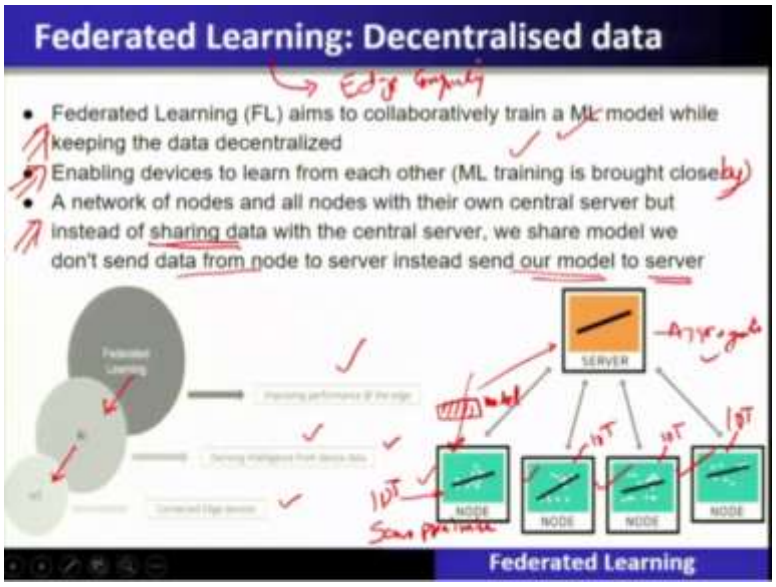
So, 2016 the term federated learning is first coined by the company that is the Google where the researchers, they have published more than 1000 papers in federated learning of their contributions. Now we have already seen some of the real-world gadgets and the deployments by the companies and researchers for large scale IoT devices using that distributed machine learning concept and also called as federated learning.

So, several open-source libraries therefore are also under the development. Lot of resources are available in this particular area that is called a federated learning, PySyft, Tensorflow federated, then flower and so on. There are many other platforms for federated learning platforms are available. These platforms will provide the customers to bring their federated learning algorithm and run on a particular scenario which is called a distributed data. So, these particular federated learning platforms, they provide the protocols that is federated learning protocols.

So, these protocols we are going to cover in this particular lecture session. And also, we will cover one very important federated learning algorithms. And these particular platforms which are available as open-source that you can refer and see that particular libraries due to the scope that due to the time constraint, we are not going to cover that platforms. So, we are going to cover rated learning algorithms and federated learning protocols.

So, federated learning is highly multidisciplinary. It involves machine learning, numerical optimization, privacy, security, network systems and hardware. So, we are going to cover these aspect of federated learning, which is becoming multidisciplinary and very popular. So, some aspects from this particular angle we are going to cover.

(Refer Slide Time: 14:07)



Federated Learning: Decentralised data

- Federated Learning (FL) aims to collaboratively train a ML model while keeping the data decentralized
- Enabling devices to learn from each other (ML training is brought closeby)
- A network of nodes and all nodes with their own central server but instead of sharing data with the central server, we share model we don't send data from node to server instead send our model to server

Federated Learning

So, federated learning means that performing the machine learning with the federated with a decentralized data. So, federated learning aims to collectively train the machine learning models while keeping the data decentralized. That means data will remain wherever it is generated or it is kept to the satisfaction of privacy and the locality or localization. Data may not have to move away from that premise to the cloud for performing this distributed learning. And that particular kind of concept is called federated learning.

So, federated learning cons technologies aim to collectively train the machine learning model while keeping the data decentralized. This is very, very important development. We will see how that is possible using federated learning. Federated learning also enable the devices to learn from each other. There is a machine learning is brought very close by. So, then next thing is that federated learning in al learning and network of nodes or all the node with their own central server. But in instead of sharing the data with the central server, we will share the model. We do not send the data from one node to the server in a state, we will only send the model and that is acceptable in this particular learning power lens.

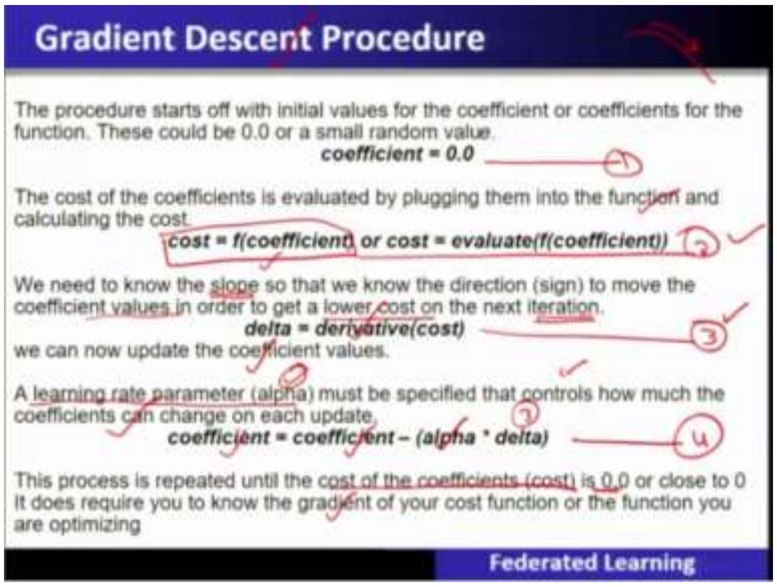
So, take this example here, you have one end that is called connected as devices. Then, these edge devices are having that model of decentralized data. Now you can apply these machine learning models that is AI using federated learning to derive the intelligence from the device data without moving the data to the cloud. So, that particular concept of an AI qualified learning will improve the performance at the edge.

So, the edge that means that part of training or machine learning which earlier used to happen at the cloud, now will be performed either the edge where these devices are operating. So, take this particular example, the nodes are the edge devices where the IoT is connected to or is being supported. Now, these nodes are given, they are the edge nodes very close by to that particular IoT, maybe in the same prim, same premise. So, it is in the same premise these IoT and these edge nodes. So, IoT will generate the data and it is available at the edge nodes.

Now if you bring the machine learning model, if you can bring the machine learning algorithm and train the model using this IoT data at the node itself that is at the edge nodes and then the trained model will be sent to the servers and server will do the aggregation of this trained model. So, that is the architecture of federated learning that is about the distributed data, how that is all done through the, through the federated learning that we are going to see.

So, therefore, therefore federated learning is the federated learning will give the way or the concepts to the edge computing. So, with the help of federated learning, the edge computing is becoming more mature.

(Refer Slide Time: 18:22)



Gradient Descent Procedure

The procedure starts off with initial values for the coefficient or coefficients for the function. These could be 0.0 or a small random value.

coefficient = 0.0 ①

The cost of the coefficients is evaluated by plugging them into the function and calculating the cost.

cost = f(coefficient) or cost = evaluate(f(coefficient)) ②

We need to know the slope so that we know the direction (sign) to move the coefficient values in order to get a lower cost on the next iteration.

delta = derivative(cost) ③

we can now update the coefficient values.

A learning rate parameter (alpha) must be specified that controls how much the coefficients can change on each update.

*coefficient = coefficient – (alpha * delta)* ④

This process is repeated until the cost of the coefficients (cost) is 0.0 or close to 0
It does require you to know the gradient of your cost function or the function you are optimizing

Federated Learning

Let us go ahead and see the growth. Now to do this aggregated learning or learning at the nodes and aggregated learning at the server. So, that learning part is done through a particular process or the algorithm and that is popularly known as gradient descent. So, you can see on one side you have the machine learning algorithm and the other side you have the data. So, data and machine learning both cannot be used to train the system. For that you require a glue, you require another technique and that is called the gradient descent.

So, using gradient descent and you can use the data or the gradient descent will use the data and train the machine learning model. So, training has to be happen using an algorithm that is called a gradient descent. Let us understand about that procedure. So, gradient descent procedure starts off with the initial values of a coefficient for the function and this could be in a slice to 0 or a very small random number. Let us assume that this particular coefficient is in a slice to 0. That becomes the equation number 1.

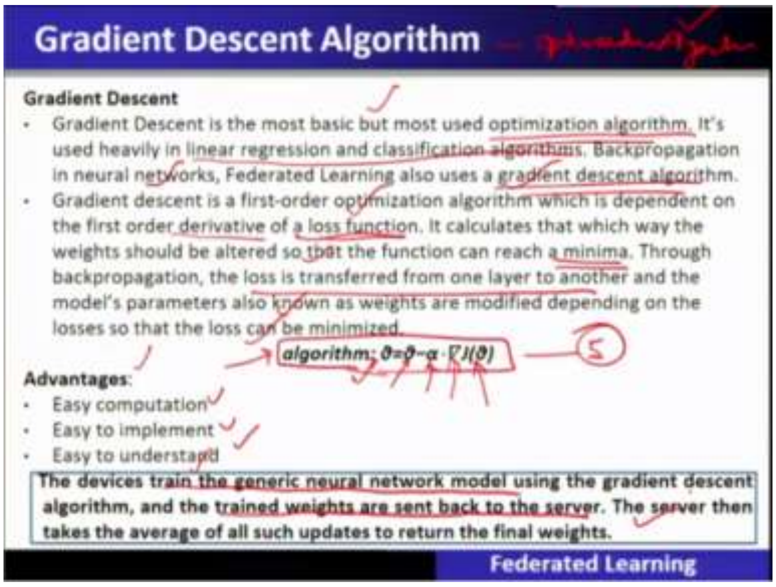
Then the cost of coefficient is evaluated by plugging them in a function and calculating the cost. So, cost of this particular coefficient is a function. So, either that will be computed with the help of a function or it will be evaluated applying this particular function of that coefficients and that will calculate the cost. Now we need to know the slope gradient descent works. Descent means it will go to an optimal value. That means it will, it has to find out a converts optimization for that, it has to now work around the slope so that you know that direction or the sign of moving the coefficient values in order to get the lower cost. So, this cost function has to be reduced to the minimal.

So, how to go ahead to find out the optimal values or the iterations. So, this procedure is well defined using the method called gradient descent. So, gradient descent will use the derivative of that particular cost and that becomes the delta. So, delta is nothing but the derivative of the cost function, which I have already told that it is derived out of the coefficient by applying a function and that becomes a cost of cost function. So, the cost and the derivative of that particular cost will become the delta. So, it has to be differentiatable.

So, we can now update the coefficient values and the learning rate parameter. We have to also fix that is called the alpha, which is specified by the designers. So, learning rate parameter, which is called the alpha must be specified that controls how much coffee she can change on each update. So, the coefficient, so the fourth equation using 1, 2, 3, it becomes the update of coefficient.

So, coefficient will be updated by plugging the learning rate that we have already explained and the data which we have calculated out of step number 3. So, this process of update is done in one iteration. So, the process is repeated until the cost of this coefficient becomes 0 or very close to 0. So, it does require to know the gradient of your cost function or a function you are optimizing.

(Refer Slide Time: 22:41)



Gradient Descent Algorithm

Gradient Descent

- Gradient Descent is the most basic but most used optimization algorithm. It's used heavily in linear regression and classification algorithms. Backpropagation in neural networks, Federated Learning also uses a gradient descent algorithm.
- Gradient descent is a first-order optimization algorithm which is dependent on the first order derivative of a loss function. It calculates that which way the weights should be altered so that the function can reach a minima. Through backpropagation, the loss is transferred from one layer to another and the model's parameters also known as weights are modified depending on the losses so that the loss can be minimized.

algorithm: $\theta = \theta - \alpha \cdot \nabla J(\theta)$

Advantages:

- Easy computation
- Easy to implement
- Easy to understand

The devices train the generic neural network model using the gradient descent algorithm, and the trained weights are sent back to the server. The server then takes the average of all such updates to return the final weights.

Federated Learning

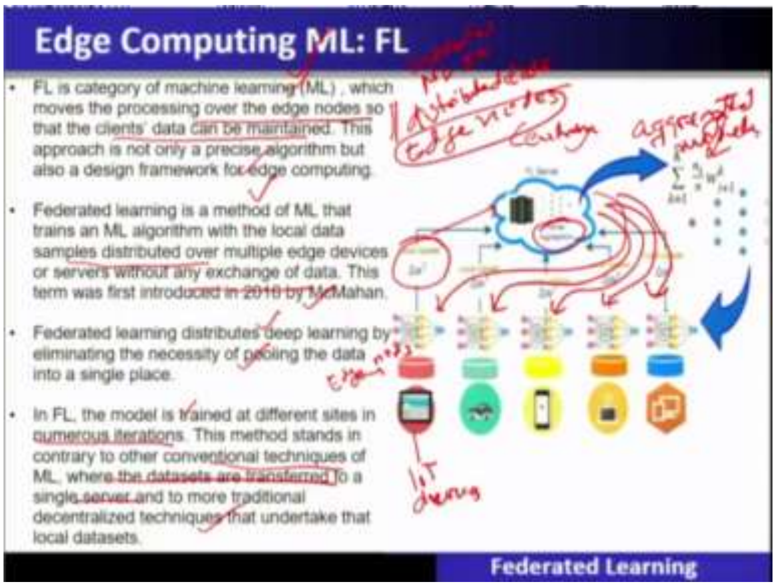
So, the gradient descent algorithm, let us briefly cover it. So, gradient descent is the most basic but most used optimization algorithm. So, gradient descent is an optimization algorithm. So, it is, itis used heavily in the linear regression and the classification activities. So, back propagation in the neural network. The federated learning is also uses this gradient descent, whether it is a back propagation in the neural network and federated learning both users gradient descent as an optimization algorithm.

So, what are we optimizing? We are optimizing the learning of a model. So, gradient descent is the first order optimization algorithm which is dependent upon the first order derivatives of a loss function. So, it calculates that which way the weight should be altered so that the function can reach the minima and through the back propagation the losses transferred from one layer to another layer and the models parameter also known as the weights are modified depending upon the losses so that losses can be minimized.

Therefore, the algorithm can be specified or executing several iterations and it can be very briefly explained using only one equation. So, again, that equation we have already explained. So, this is the coefficient is updated with the help of learning rate and the derivative of the loss function. So, the advantages we have seen in this optimization algorithm, which requires to train with the decentralized data, we are going to see the protocol of federated learning, how that is using the gradient descent for this method. So, advantage is that it is easy to compute, easy to implement, easy to understand.

Now once having understood this gradient descent optimization algorithm, then we have to go for the next step to understand the entire federated learning algorithm. So, the devices train the generic neural network model using gradient descent algorithm and thereby, the trained waves are sent to the server. The server then takes the average of all the updates to return the final weights.

(Refer Slide Time: 25:28)



Edge Computing ML: FL

- FL is category of machine learning (ML) , which moves the processing over the edge nodes so that the clients' data can be maintained. This approach is not only a precise algorithm but also a design framework for edge computing.
- Federated learning is a method of ML that trains an ML algorithm with the local data samples distributed over multiple edge devices or servers without any exchange of data. This term was first introduced in 2016 by McMahan.
- Federated learning distributes deep learning by eliminating the necessity of pooling the data into a single place.
- In FL, the model is trained at different sites in numerous iterations. This method stands in contrary to other conventional techniques of ML, where the datasets are transferred to a single server and to more traditional decentralized techniques that undertake that local datasets.

Federated Learning

So, let us see before we go and discuss the federated learning algorithm, let us see about this edge computing with the help of federated learning to perform the machine learning. So, federated learning is a category of machine learning algorithm which moves the processing over the edge nodes. What do you mean by moving the processing or the edge node?

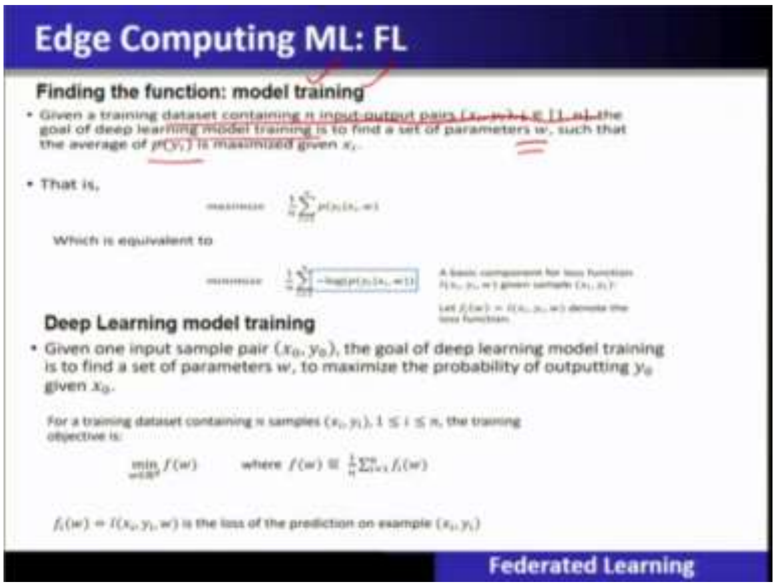
So, earlier the processing in the centralized data. So, that means moving the processing to the client, to the edge nodes so that the clients can, client’s data can be maintained there itself without movement. So, this gives a birth to the decentralized distributed data, performing the machine learning on distributed data or also you can say itis a distributed machine learning. So, if you are performing the machine learning on those distributed nodes, they are called edge nodes, because they will be able to perform the computation. This approach is not only a precise algorithm but also a design of a framework for edge computing that we will explain.

Now federated learning is a method of machine learning that trains the machine learning model with the local data samples distributed or multiple edge devices or the server without exchange of this particular data. And this term federated learning was coined by by McMahan in 2016. So, federated learning distributes the deep learning by eliminating the necessity of pulling the data in one single place.

So, learning the model is trained at different sites in numerous iterations. This method is stands in contrary to the conventional techniques of a machine learning where dataset is transferred to a single server that is the cloud and to a more traditional decentralized techniques that undertake that local dataset.

So, for example, these are all IoT devices, they are sending the data to an, to the edge nodes. Now with the federated learning, which is being coordinated, learning is being performed where the data is and then it will send only the parameters, not the data. So, data will stay wherever it is being generated. And it will be sent to the centralized server that is, which is running the federated learning server for model aggregation. So, it will do the model aggregation, this is the formula for model aggregation and then this particular server will again push back these particular updates of the aggregated model parameters.

(Refer Slide Time: 28:56)



Edge Computing ML: FL

Finding the function: model training

- Given a training dataset containing n input-output pairs $(x_i, y_i), i \in [1, n]$, the goal of deep learning model training is to find a set of parameters w, such that the average of $p(y_i)$ is maximized given x_i.
- That is,

$$\text{maximize} \quad \frac{1}{n}\sum_{i=1}^{n} p(y_i|x_i, w)$$

Which is equivalent to

$$\text{minimize} \quad \frac{1}{n}\sum_{i=1}^{n} \boxed{-\log(p(y_i|x_i, w))}$$

A basic component for loss function $l(x_i, y_i, w)$ given sample (x_i, y_i):

Let $f_i(w) = l(x_i, y_i, w)$ denote the loss function.

Deep Learning model training

- Given one input sample pair (x_0, y_0), the goal of deep learning model training is to find a set of parameters w, to maximize the probability of outputting y_0 given x_0.

For a training dataset containing n samples $(x_i, y_i), 1 \le i \le n$, the training objective is:

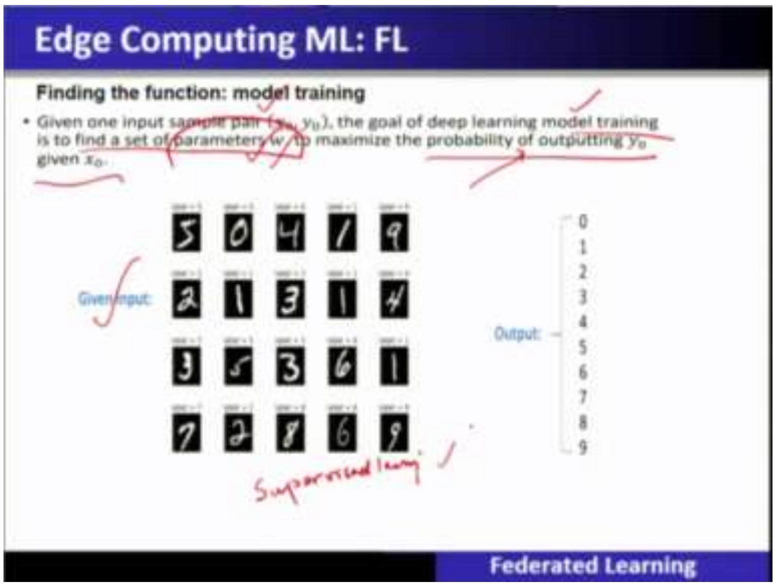
$$\min_{w \in \mathbb{R}^d} f(w) \qquad \text{where } f(w) \equiv \frac{1}{n}\sum_{i=1}^{n} f_i(w)$$

$f_i(w) = l(x_i, y_i, w)$ is the loss of the prediction on example (x_i, y_i)

Federated Learning

So, therefore, let us see what are the building blocks or governing equations or governing algorithms which will perform the model training in a decentralized way at the edge nodes, that is wherever this IoT is functioning. So, thereby in the model training you have to find a function for that model training. So, given a training dataset which comprises of n input and output pair, let us say that xi and yi and which belongs to domain of that dataset. And the goal of this deep learning model training is to find a set of parameter w says that the average of p of yi becomes maximized given an xi.

(Refer Slide Time: 29:48)

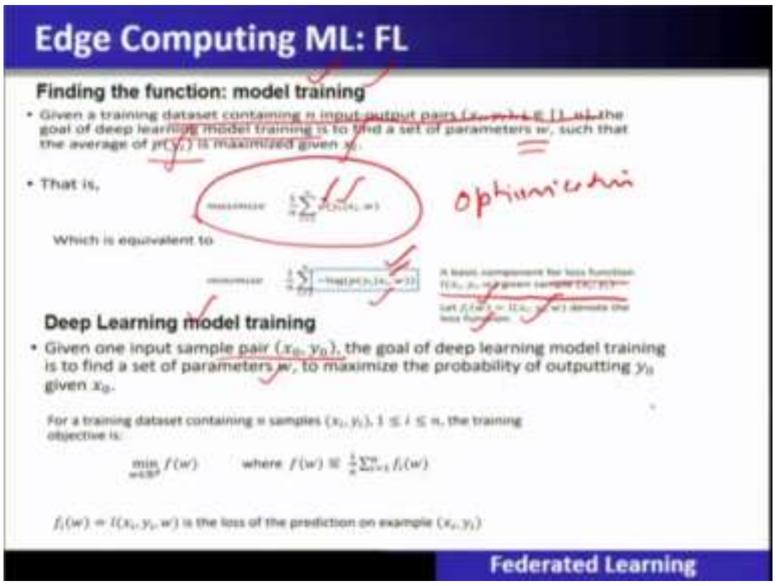


Edge Computing ML: FL

Finding the function: model training

- Given one input sample pair (x_0, y_0), the goal of deep learning model training is to find a set of parameters w, to maximize the probability of outputting y_0 given x_0.

Federated Learning



Edge Computing ML: FL

Finding the function: model training

- Given a training dataset containing n input-output pairs $(x_i, y_i), i \in [1, n]$, the goal of deep learning model training is to find a set of parameters w, such that the average of $p(y_i)$ is maximized given x_i.
- That is,

$$\text{maximize} \quad \frac{1}{n}\sum_{i=1}^{n} p(y_i | x_i, w)$$

optimization

Which is equivalent to

$$\text{minimize} \quad \frac{1}{n}\sum_{i=1}^{n} \boxed{-\log(p(y_i | x_i, w))}$$

A basic component for loss function $l(x_i, y_i, w)$ given sample (x_i, y_i)

Let $f_i(w) = l(x_i, y_i, w)$ denote the loss function.

Deep Learning model training

- Given one input sample pair (x_0, y_0), the goal of deep learning model training is to find a set of parameters w, to maximize the probability of outputting y_0 given x_0.

For a training dataset containing n samples (x_i, y_i), $1 \leq i \leq n$, the training objective is:

$$\min_{w \in \mathbb{R}^d} f(w) \qquad \text{where } f(w) \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} f_i(w)$$

$f_i(w) = l(x_i, y_i, w)$ is the loss of the prediction on example (x_i, y_i)

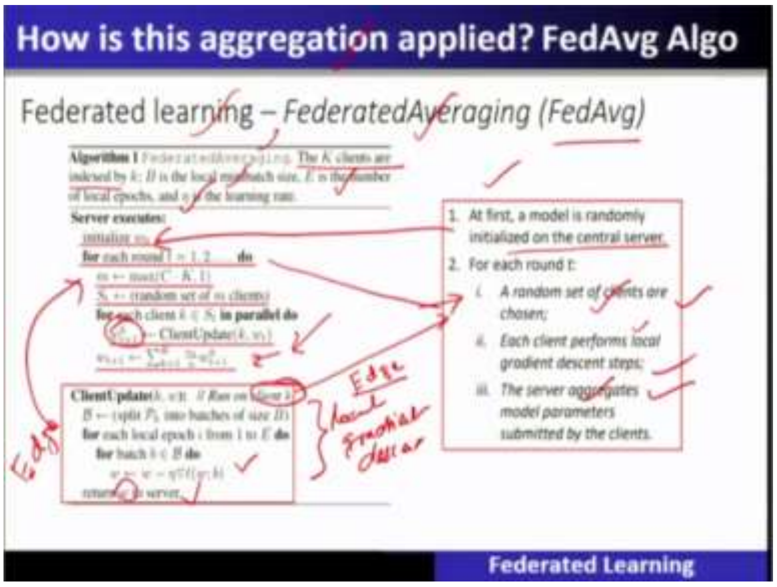
Federated Learning

So, this means that if you are given a input sample of a dataset that is let us say x0 and y0, the goal of the deep learning model training is to find a set of parameter w to maximize the probability of outputting y0 in giving x0. So, given the input dataset, let us say this is the trained dataset or it is labelled dataset. This will be the example of supervised learning. So, this particular supervised learning will train the model and it will collect the deep learning model training parameters such as w with the scope that it will maximize the probability of outputting y0.

Means that what are the parameters when your model is perfectly trained with the supervised learning of a dataset. Therefore, you have to find a particular probability function which will maximize the value of y given xi and that is being expressed the maximization, that is an optimization in the model training process. So, that means you want to maximize the probability of maximum values of yi learn given the value of x. So, which is equivalent minimizing the error. So, the error function is written in this following manner that is minus log of p of yi given the values of x and w.

So, this basic component for the loss function is given with a sample dataset and let f w be the function of these particular w and thereby, it will be easy to calculate the loss function by applying the log values to it and to minimize it. So, deep learning model training in this particular method, using this method you are given an input sample x0, y0 and the goal of the deep learning model training is to find the set of parameter w to maximize the probability of outputting y0 given x0.

(Refer Slide Time: 32:27)



How is this aggregation applied? FedAvg Algo

Federated learning – *FederatedAveraging (FedAvg)*

```
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, ... 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
      $w_{t+1}^k \leftarrow$ ClientUpdate(k, $w_t$)
    $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$

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
```

1. At first, a model is randomly initialized on the central server.
2. For each round t:
   i. *A random set of clients are chosen;*
   ii. *Each client performs local gradient descent steps;*
   iii. *The server aggregates model parameters submitted by the clients.*

So, let us see that aggregation, how the server does this aggregation of the models which are trained at different decentralized data sites. And then what is the governing algorithm which will do this aggregation method. So, very important algorithm is called federated averaging fed average is properly known as and it has given the lot of use for federated learning using this particular algorithm.

So, algorithm goes like this, this algorithm is called federated averaging. This particular algorithm does has two parts. One is called the server, the other is called the clients. So, clients will perform using federated averaging, which is a particular algorithm of performing the federated learning. So, federated learning algorithm called federated averaging comprises of two components. One will run at the client, client means all the devices will be running this particular algorithm called client update.

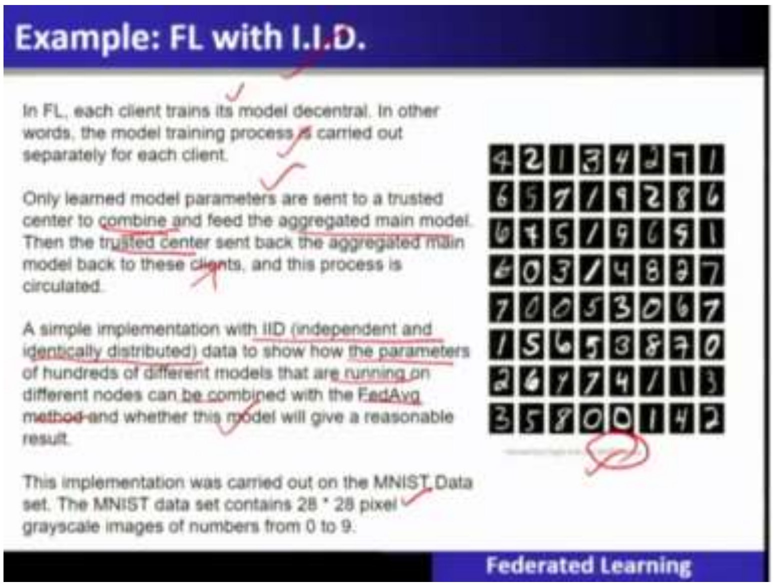
So, what it will do, it will split the Pk into the batches of size B and for each epoch i from 1 to E, it will perform in the batches. And you know that these iterations, what we have discussed in the previous slide is that coefficients or the parameters, weights we call it with a learning rate and the derivative, this particular things will be returned to the server after being computed this weight w at the client's end.

Server, what it will do? Initially, when none of the clients have sent the data or weights back. So, it has to initialize with weight 0 and for every round or for every iterations what it will do, it will maximize the C K and one and it will also search the random set of M clients and then it will update the clients, client update when comes here in the server, it will be then perform the averaging of these weights and this is called the federated learning and these newly calculated weights are being again fed back to the client. Let us understand.

So, at the first the model is randomly initialized on the central server that we have already shown. So, at first, the model is randomly initialized on the central server. Then, for each round t, a random set of clients are choosing not all the clients. So, some k clients are randomly chosen by the server and each client performs local gradient descent steps. So, here the local gradient steps are being executed on every edge nodes. So, they are at the edge.

So, edge nodes will execute the gradient descent the steps and then the computed weights that is w is sent to the server. And the server will aggregate the model parameter submitted by the client here. So, the client update will submit its weight and then it will be aggregated and then updated and so on. So, this is a very simple way of a federated learning. So, federated learning will choose, will select randomly the k clients. So, not all the clients, randomly k clients are being chosen maybe and B is the local back size which is required by the clients or the edge nodes and E is the number of locally epoch and there is a learning rate. These parameters are needed.

(Refer Slide Time: 37:21)



Example: FL with I.I.D.

In FL, each client trains its model decentral. In other words, the model training process is carried out separately for each client.

Only learned model parameters are sent to a trusted center to combine and feed the aggregated main model. Then the trusted center sent back the aggregated main model back to these clients, and this process is circulated.

A simple implementation with IID (independent and identically distributed) data to show how the parameters of hundreds of different models that are running on different nodes can be combined with the FedAvg method and whether this model will give a reasonable result.

This implementation was carried out on the MNIST Data set. The MNIST data set contains 28 * 28 pixel grayscale images of numbers from 0 to 9.

Federated Learning

Now let us see implementation experience of a federated learning with I.I.D data. So, in federated learning each client trains its model in a decentralized way. In other words, the model training process carried out separately at each client that we have explained in the previous slides, only the learn model parameters are sent to the server to aggregate and feed the aggregated main model back. Then the trusted server will send back the aggregated main model back to these particular client. And this process is continue.

Now we will see an experience of this federated learning implementation with with I.I.D data. The full form of I.I.D is Independent and Identically Distributed.

So, Independent and Identically Distributed data, I.I.D data implementation is shown here in this particular example. So, we will see that how the parameters of hundreds of different models that are running on different node can be combined using federated averaging algorithm method, which we have seen one type of federated learning algorithm, which I have already discussed in the previous slides.

So, whether this model that is federated averaging based training will give the reasonable results or not. So, let us see this experience. So, this particular dataset, let us see that itis a public dataset which is widely used called MNIST dataset. Itis a handwritten digits. So, this dataset will contain the images in the grey scale of handwritten digit from 0 to 9.

(Refer Slide Time: 39:29)

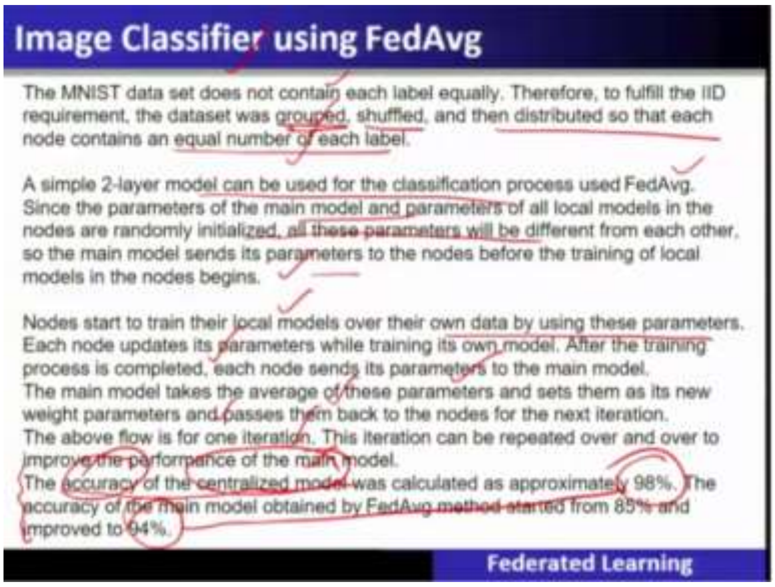


Image Classifier using FedAvg

The MNIST data set does not contain each label equally. Therefore, to fulfill the IID requirement, the dataset was grouped, shuffled, and then distributed so that each node contains an equal number of each label.

A simple 2-layer model can be used for the classification process used FedAvg. Since the parameters of the main model and parameters of all local models in the nodes are randomly initialized, all these parameters will be different from each other, so the main model sends its parameters to the nodes before the training of local models in the nodes begins.

Nodes start to train their local models over their own data by using these parameters. Each node updates its parameters while training its own model. After the training process is completed, each node sends its parameters to the main model.
The main model takes the average of these parameters and sets them as its new weight parameters and passes them back to the nodes for the next iteration.
The above flow is for one iteration. This iteration can be repeated over and over to improve the performance of the main model.
The accuracy of the centralized model was calculated as approximately 98%. The accuracy of the main model obtained by FedAvg method started from 85% and improved to 94%.

Federated Learning

So, let us see how you can build the same classifiers which centralized learning also has already done this experience. So, MNIST dataset does not contain each label equally therefore, to fulfil the independent and identically distributed requirement, the dataset was grouped, shuffled, and then distributed so that each node contains an equal number of each label. So, this is very important.

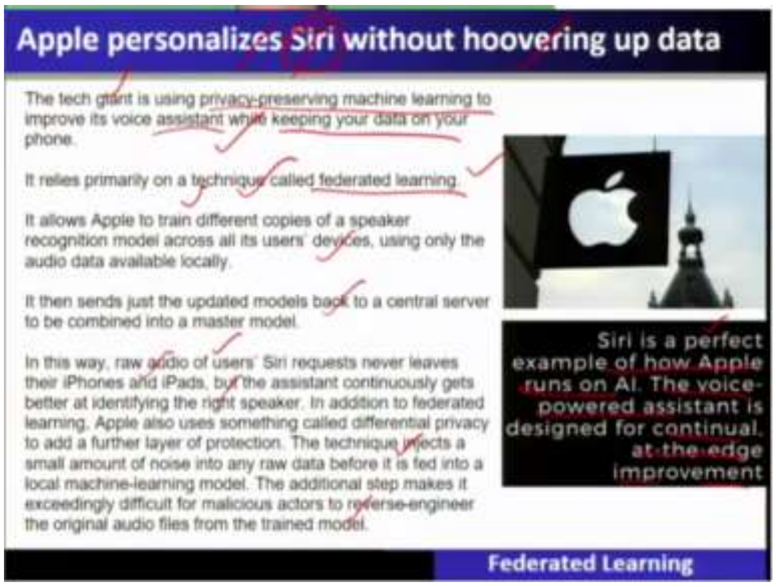
So, the very basic requirement of independent and identically distributed is to group them, shuffle them, and then distribute so that each node contains an equal, almost equal number of equal number of each label. So, here you can use a very simple two-layer model for the classification process and applying the federated learning. Now these parameters of the main model and the parameters of all the local models are randomly initialized that we have already explained. So, these parameters will be different from each and the main model will send its parameter to the node before the training of a local data begins.

Node start to train their local models over their own data by using these parameters and each node updates its parameter while training its own model. And after the training processes complete the node sense its parameter to the to the main model. Main model takes the average of these parameter and set them its new weight and passes back to the nodes for the next iteration.

The above flow is for one iteration and this iteration can be repeated over and over to improve the main model. So, what we have seen is that let us see important comparison that if you use the same dataset in a deep learning model for the classification purpose, the accuracy of the centralized model in a similar implementation is giving 98 percent accuracy in a centralized model. What about here in federated learning using federated average accuracy is obtained 85 percent and can be improved up to 94 percent.

So, you can see that you can achieve similar accuracy, almost similar accuracy in this distributed manner that is using five learning without even sending the data to the central cloud. So, this particular development is quite serious and long lasting in the sense that it will join the wave of innovation for IoT use cases and the growth using the edge computing.

(Refer Slide Time: 42:38)



Apple personalizes Siri without hoovering up data

The tech giant is using privacy-preserving machine learning to improve its voice assistant while keeping your data on your phone.

It relies primarily on a technique called federated learning.

It allows Apple to train different copies of a speaker recognition model across all its users' devices, using only the audio data available locally.

It then sends just the updated models back to a central server to be combined into a master model.

In this way, raw audio of users' Siri requests never leaves their iPhones and iPads, but the assistant continuously gets better at identifying the right speaker. In addition to federated learning, Apple also uses something called differential privacy to add a further layer of protection. The technique injects a small amount of noise into any raw data before it is fed into a local machine-learning model. The additional step makes it exceedingly difficult for malicious actors to reverse-engineer the original audio files from the trained model.

Siri is a perfect example of how Apple runs on AI. The voice-powered assistant is designed for continual, at-the-edge improvement

Federated Learning

Now let us see some more use cases of these federated learning. So, you might be using the Apple mobile phone and Apple Mobile phone is running a voice assistant called Siri. That

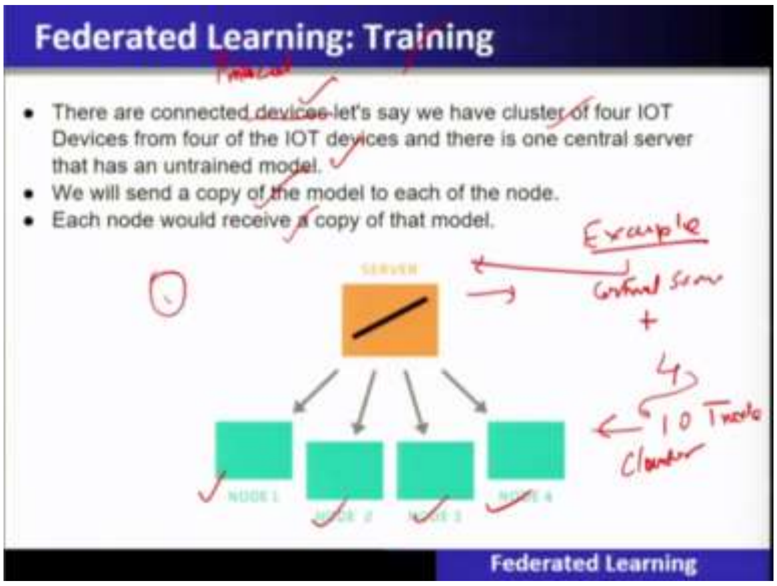
Siri is now using your personalized data to optimize your phone configurations without hovering up the data. So, your data protection is there is still it can use your data which is staying still in the mobile phone but training the model with the help of federated learning.

So, the tech giant like Apple is using privacy preserving machine learning to improve the voice assistant that is Siri while keeping your data on your phone. So, data is not moving, yet that particular voice assistant Siri will be now learning with your data which is there in your mobile phone. It relies primarily on the technical federated learning that we have already explained. So, it allows the apple to train different copies of the speaker recognition model across the user's devices using only the audio data available locally.

So, Siri is the perfect example how the Apple runs, a privacy preserving AI that is in the form of voice powered assistant is designed for continual at the edge improvement. So, this, it sends the data, it sends just the updated models back to the server, not the data. In this manner, the raw audio of the users Siri request never leaves the iPhone or iPad, but the assistant continuously get better at identifying the right speaker call differential privacy to further layer of protection.

So, the technique ingest, injects a small amount of noise in the local machine learning model and additional steps makes it difficult for malicious actors to reverse engineer the original data from the trained model.

(Refer Slide Time: 44:51)



Federated Learning: Training

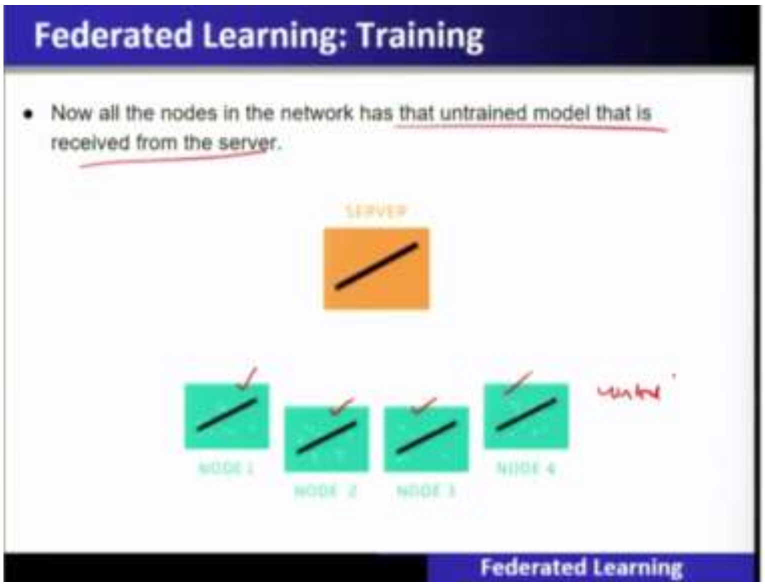
- There are connected devices let's say we have cluster of four IOT Devices from four of the IOT devices and there is one central server that has an untrained model.
- We will send a copy of the model to each of the node.
- Each node would receive a copy of that model.

So, those kind of privacy issues are already taken care in the federated learning. So, federated learning model training protocol, let us go ahead and see. So, we have so far seen the federated learning algorithm and federated learning optimization algorithm that is called gradient descent. We have also seen one use case of a federated learning that is in the form of Apple’s Siri, how that is being learning without moving the data and that is called machine learning or privacy preserving machine learning that is using the federated learning.

So, federated learning protocol, let us see how all these steps are done. Let us see about the training. So, in the training there are nodes, these nodes are having connected with the connected devices and let us say that you have a cluster of 4 IoT devices and there is one central server in this example, you have one central server plus you have 4 IoT clusters, node cluster. So, there is one central server and initially it is having untrained model and we will send the copy of this untrained model to each node and each node will receive the copy of that model. This is the step number 1.

So, let us see how the training will happen when the server is sending the model that is untrained model to the nodes.

(Refer Slide Time: 47:00)

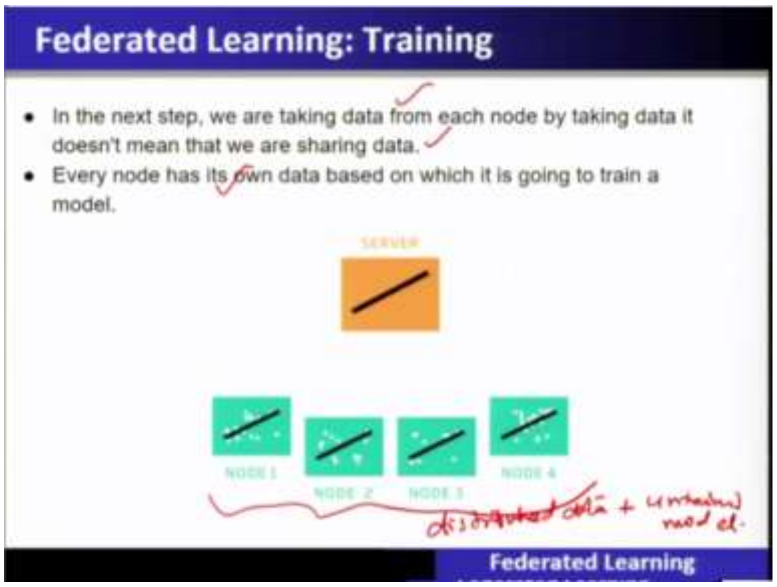


Federated Learning: Training

- Now all the nodes in the network has that untrained model that is received from the server.

All the node in the network has untrained model that is received by the servers.

(Refer Slide Time: 47:13)

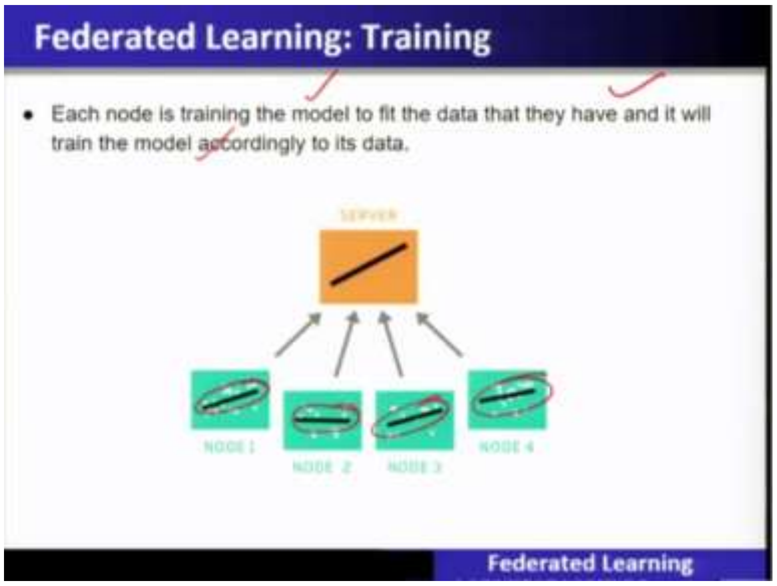


Federated Learning: Training

- In the next step, we are taking data from each node by taking data it doesn't mean that we are sharing data.
- Every node has its own data based on which it is going to train a model.

So, in the next step we are taking the data from each node by taking the data it does not mean that we are sharing the data to anyone else other than that node. So, every node has its own data based on which is going to train a model. So, this is called the distributed data plus untrained model.

(Refer Slide Time: 47:46)

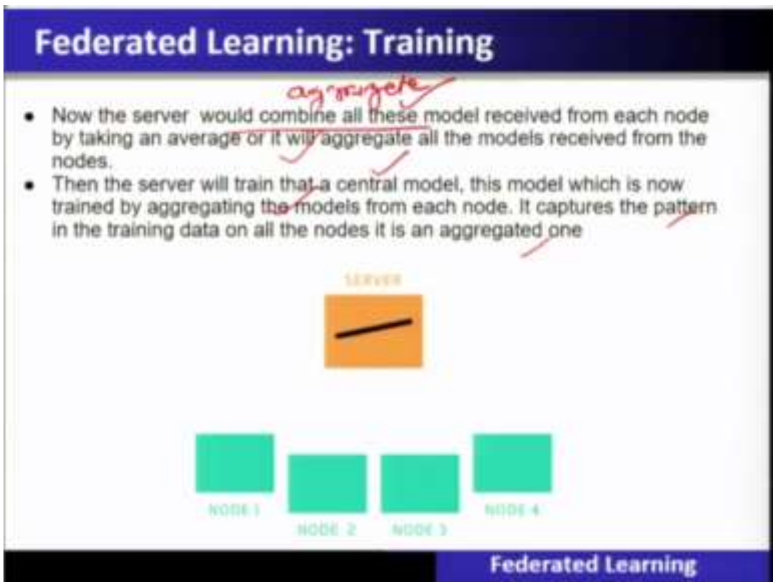


Federated Learning: Training

- Each node is training the model to fit the data that they have and it will train the model accordingly to its data.

So, each node is training the model to fit the data and that they have and it will train the model as according to its data. So, it will train the model. So, this is the picture of the model which you derive that is in the form of w, which we have seen in the fire rated averaging algorithm. So, now the server and will send that this particular train model not the data to the server.

(Refer Slide Time: 48:08)

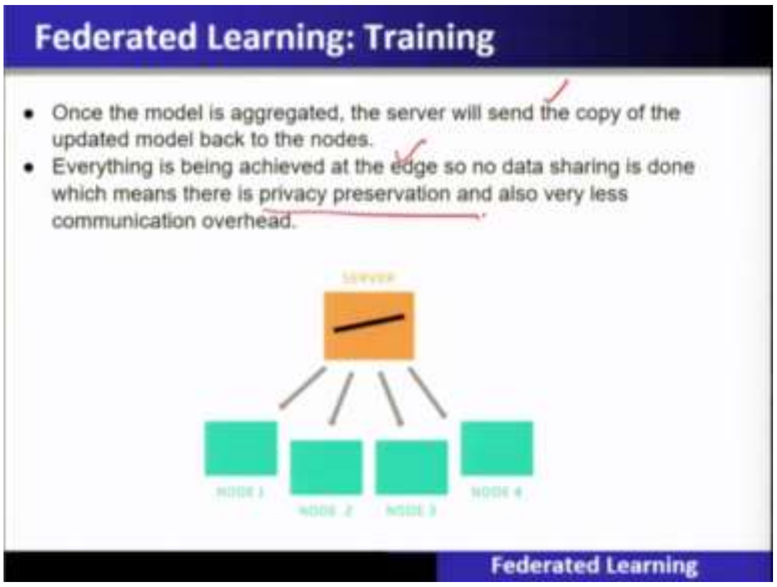


Federated Learning: Training

- Now the server would combine all these model received from each node by taking an average or it will aggregate all the models received from the nodes.
- Then the server will train that a central model, this model which is now trained by aggregating the models from each node. It captures the pattern in the training data on all the nodes it is an aggregated one

Now the server will combine that is also called aggregation aggregate. The model receive from each node by taking the average of all the model receives from the nodes, the server will train the central model with this model which is now trained by aggregating the model from each node. It captures the pattern in the training data on all the nodes it is aggregated.

(Refer Slide Time: 48:46)



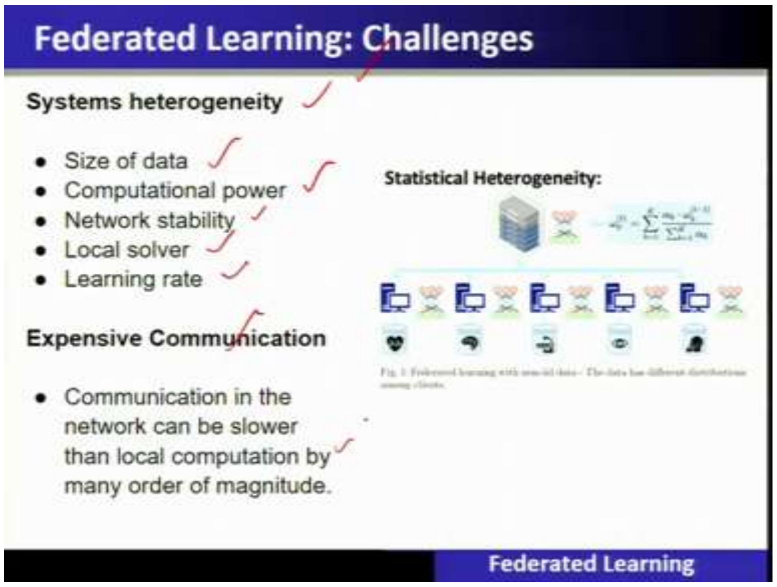
Federated Learning: Training

- Once the model is aggregated, the server will send the copy of the updated model back to the nodes.
- Everything is being achieved at the edge so no data sharing is done which means there is privacy preservation and also very less communication overhead.

SERVER
NODE 1
NODE 2
NODE 3
NODE 4
Federated Learning

Once the model is aggregated, the server will send the copy of the updated model back to the nodes and everything is being achieved at the node. So, no data sharing in this particular process of learning is done in the federated learning, means that privacy preservation and also very less communication overhead is seen. Yet the accuracy is almost similar to the central centralized machine learning we have so far seen in this particular implementation experience.

(Refer Slide Time: 49:21)



Federated Learning: Challenges

Systems heterogeneity

- Size of data
- Computational power
- Network stability
- Local solver
- Learning rate

Expensive Communication

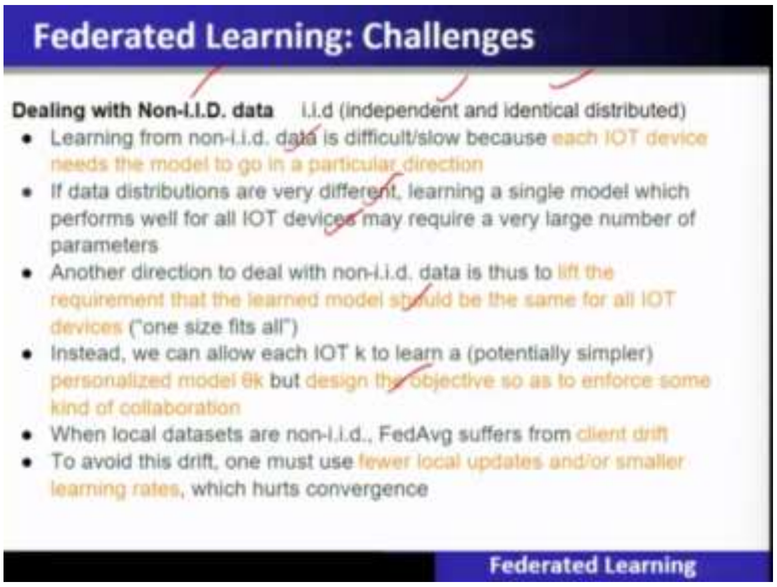
- Communication in the network can be slower than local computation by many order of magnitude.

Statistical Heterogeneity:

Federated Learning

So, federated learning challenges, let us see that. So, these are the following challenges we will have. The system's heterogeneity, the size of the data varies from different nodes. Configuration power also varies from different in different node. At the client's end. Network stability is also one of the issues. And local solvers and learning rate, these are all different parameters which brings up the heterogeneity. So, federated learning with all such systems heterogeneity becomes a challenge and also an expensive communication. That is communication. The network can be slower than than the local communication by the orders of the magnitude.

(Refer Slide Time: 50:09)



Federated Learning: Challenges

Dealing with Non-I.I.D. data i.i.d (independent and identical distributed)

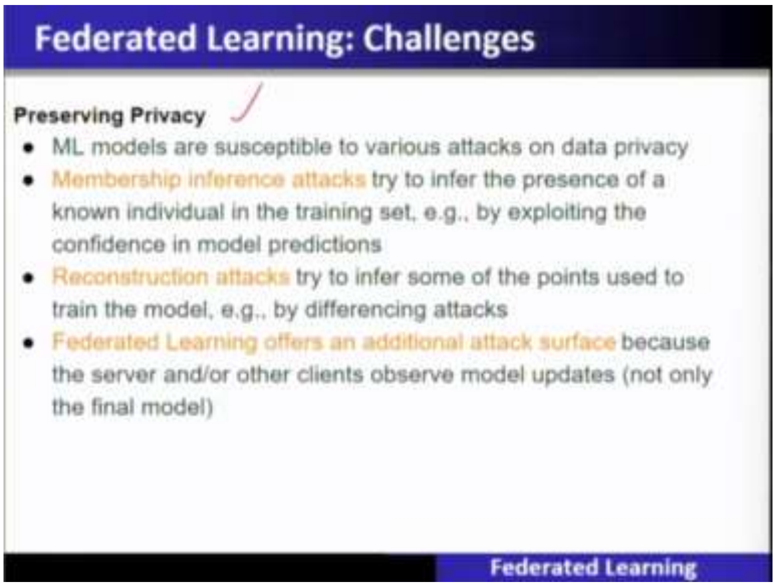
- Learning from non-i.i.d. data is difficult/slow because each IOT device needs the model to go in a particular direction
- If data distributions are very different, learning a single model which performs well for all IOT devices may require a very large number of parameters
- Another direction to deal with non-i.i.d. data is thus to lift the requirement that the learned model should be the same for all IOT devices ("one size fits all")
- Instead, we can allow each IOT k to learn a (potentially simpler) personalized model θk but design the objective so as to enforce some kind of collaboration
- When local datasets are non-i.i.d., FedAvg suffers from client drift
- To avoid this drift, one must use fewer local updates and/or smaller learning rates, which hurts convergence

Federated Learning

Now further challenges is to deal with the non I.I.D data, I.I.D data we have seen that is independent and identical distributed. So, learning from non I.I.D data is difficult and slow. Why? Because each IoT device needs the model to go in a particular direction. So, if the data distributions are very different, learning a single model which performs well for all the IoT devices may require a very large number of parameter in that case. So, this is another challenge to deal with the non I.I.D data.

Another direction is to deal with is to lift the requirement that the learn model should be the same for all the devices. So, in a state we can also allow IoT clients k, not all k to learn personalized model and so on. There are various other but it is a challenging to deal with non I.I.D data.

(Refer Slide Time: 51:08)



Federated Learning: Challenges

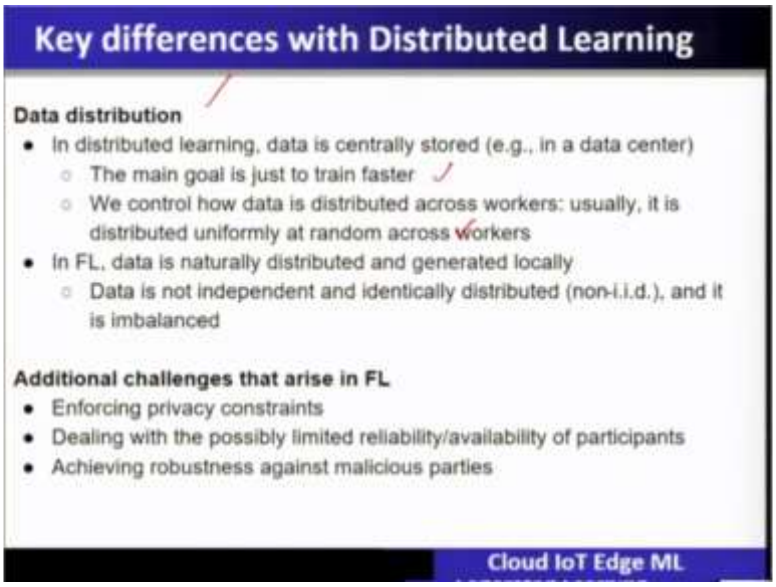
Preserving Privacy

- ML models are susceptible to various attacks on data privacy
- Membership inference attacks try to infer the presence of a known individual in the training set, e.g., by exploiting the confidence in model predictions
- Reconstruction attacks try to infer some of the points used to train the model, e.g., by differencing attacks
- Federated Learning offers an additional attack surface because the server and/or other clients observe model updates (not only the final model)

Federated Learning

So, privacy preserving, that is machine learning models are susceptible to various attacks on data privacy and the membership inference attacks, reconstruction and attacks and federated learning offers additional attacks surfaces.

(Refer Slide Time: 51:23)



Key differences with Distributed Learning

Data distribution

- In distributed learning, data is centrally stored (e.g., in a data center)
  - The main goal is just to train faster
  - We control how data is distributed across workers: usually, it is distributed uniformly at random across workers
- In FL, data is naturally distributed and generated locally
  - Data is not independent and identically distributed (non-i.i.d.), and it is imbalanced

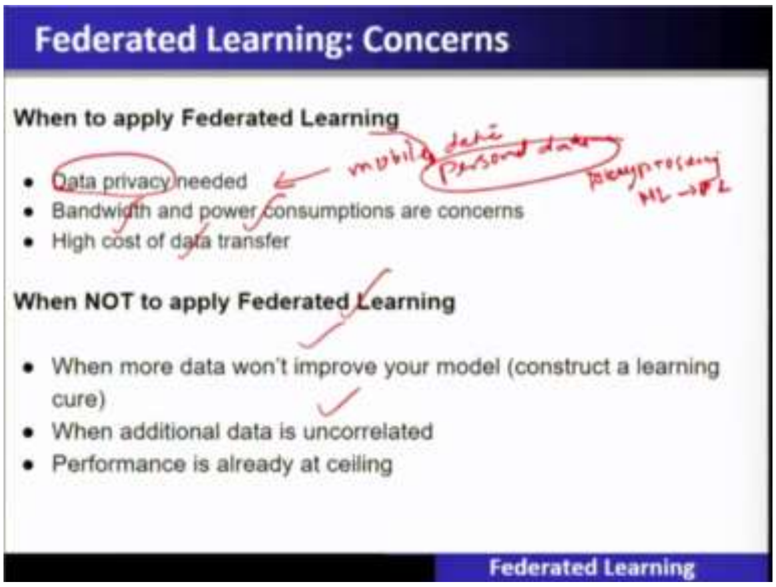
Additional challenges that arise in FL

- Enforcing privacy constraints
- Dealing with the possibly limited reliability/availability of participants
- Achieving robustness against malicious parties

Cloud IoT Edge ML

Data distribution. So, key difference with the distributor learning is the data distribution in a distributor learning data is centrally stored. That is in the data center. The main goal is to train faster and we control how the data is distributed across our. So, additional challenges in the federal learning is to enforce these privacy constraints dealing with possibly limited, reliable available participants and achieving robustness against the malicious parties.

(Refer Slide Time: 51:56)



Federated Learning: Concerns

When to apply Federated Learning

- Data privacy needed
- Bandwidth and power consumptions are concerns
- High cost of data transfer

When NOT to apply Federated Learning

- When more data won't improve your model (construct a learning cure)
- When additional data is uncorrelated
- Performance is already at ceiling

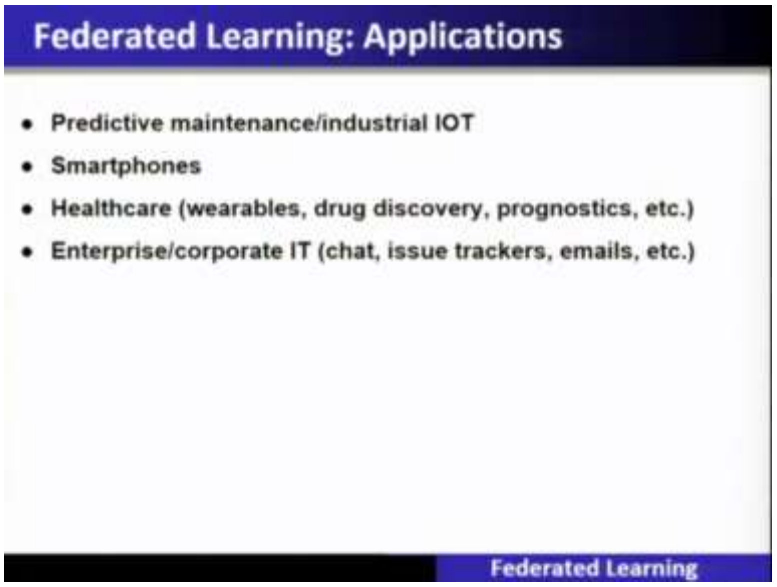
mobile data
Personal data
privacy preserving
ML → FL
Federated Learning

When to apply the t learning, whenever the data privacy is very essential. For example, the mobile phone data is a personal data. And if you want to use the data for training, then that is the only way that is privacy. Preserving machine learning is called the federated learning. Otherwise, it is not possible to pull the data from all the mobile phones to the, to the cloud and do the training. So, learning or a training is only way that is called federated learning to to perform in this kind of scenario of a mobile data, which contains the personal data.

Bandwidth and the power consumption or also of the concern. Then also federal learning is quite useful. Why? Because it does not require the huge bandwidth to send the data, but only a little bandwidth is required to send the train model. Power consumptions also are of concern, then federal learning is applied. High cost of data transfer. If it is there, then using federal learning you can reduce the cost of the data transfer because data transfer is not required while using the federated learning.

Let us see when you are not using the federated learning, what are the scenarios? When more data won't improve your moral construct a learning cure, then go for the centralized one when additional data is uncorrelated. So, that means it requires I.I.D data. If it is non IID, then also this particular federated learning will not be applicable. Performance is already at the ceiling.

(Refer Slide Time: 53:54)



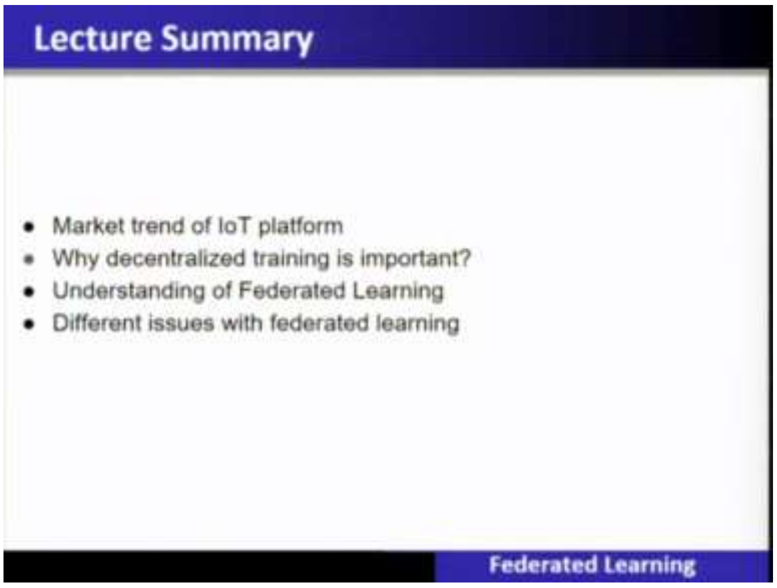
Federated Learning: Applications

- Predictive maintenance/industrial IOT
- Smartphones
- Healthcare (wearables, drug discovery, prognostics, etc.)
- Enterprise/corporate IT (chat, issue trackers, emails, etc.)

Federated Learning

There are various federated learning applications such as in the industrial IoT. Productive maintenance is a great opportunity mobile for phones, healthcare variables, drug discovery enterprise or corporate ITs.

(Refer Slide Time: 54:13)



Lecture Summary

- Market trend of IoT platform
- Why decentralized training is important?
- Understanding of Federated Learning
- Different issues with federated learning

Federated Learning

Let us summarize this particular lecture. We have seen the market trends for IoT use cases. So, IoT market is being predicted by different forecasting agencies to grow by 2025 several folds. Therefore, lot of opportunity is there for edge computing and privacy preserving machine learning to tap this, to join this wave of innovation.

So, why decentralized training is important that we have covered here in this lecture. We have also understood the federated learning concepts and we have also shown various issues involved in the federated learning, when to use, when not to use the federated learning and federated learning use cases for IoT and for mobile phone we have already seen. With this, let us conclude this lecture. Thank you very much. Thank you.