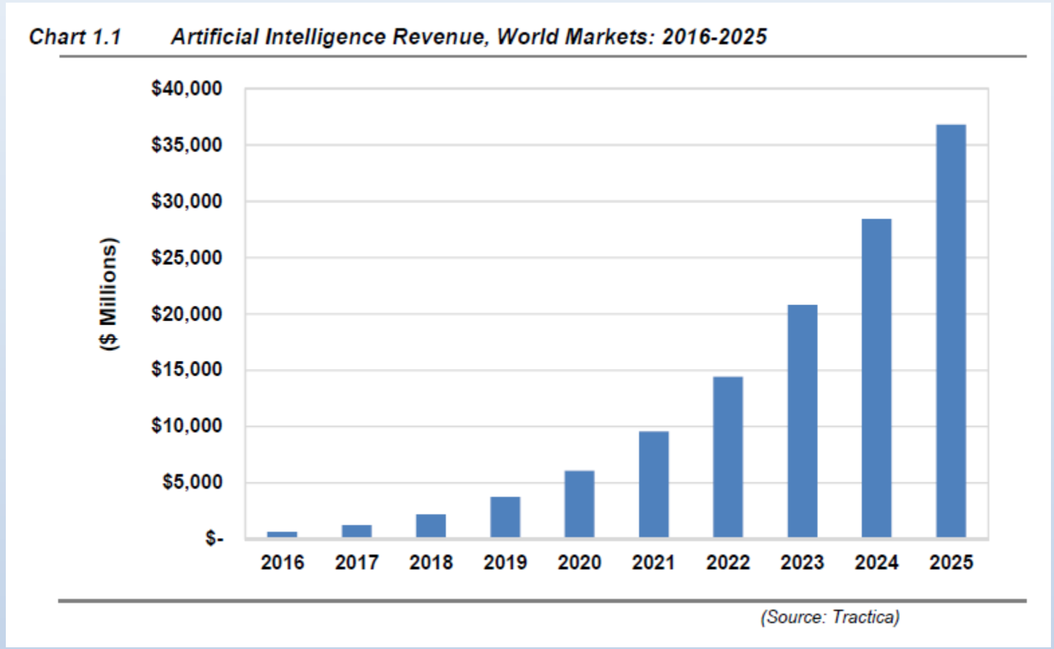
Federated Learning

Standing at the Intersection of ML, Security and System Design

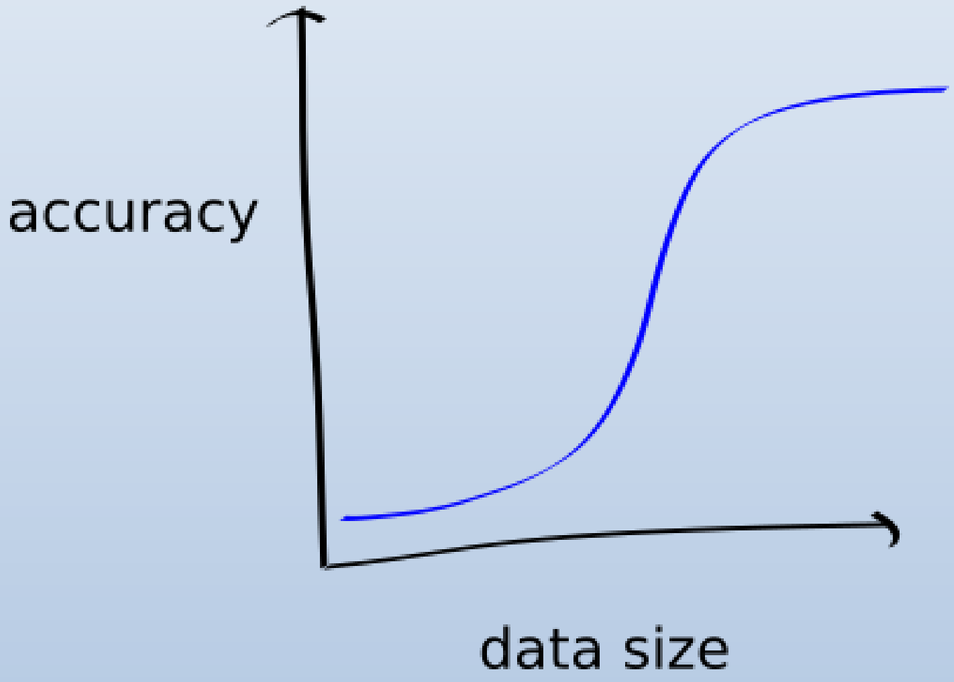
Sahand Khoshdel - Mehdi Zarei - Hamid Salami

01. Touching the Issue



How important is Artificial Intelligence today?

How much Data is needed to Train a **Good** Model?



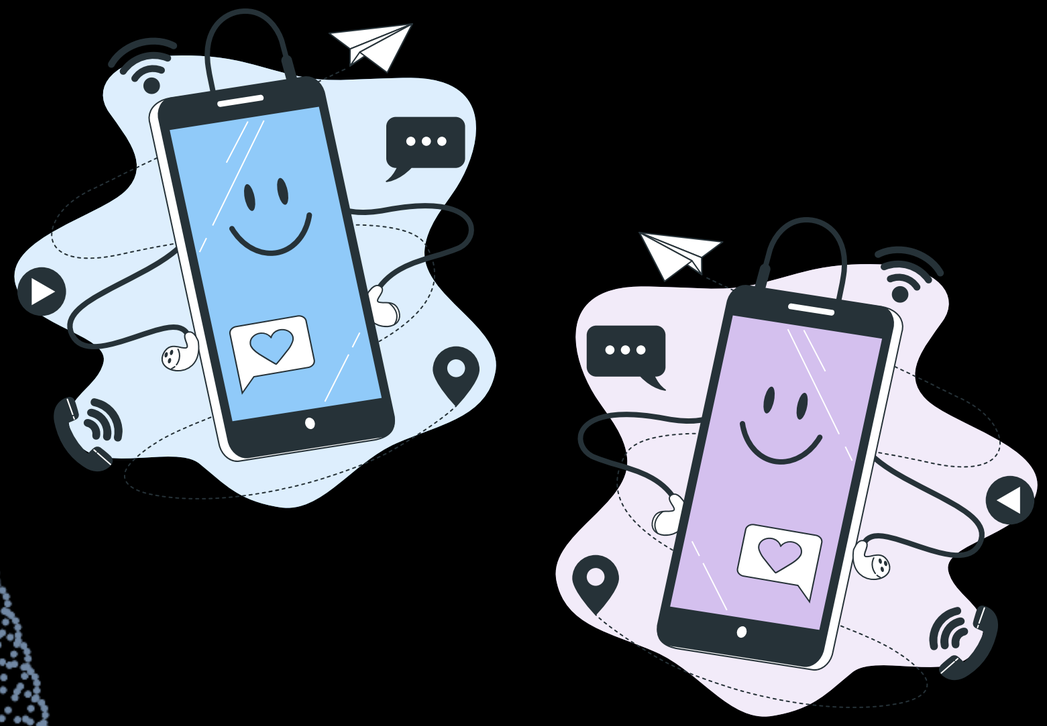
Downsides of System Centralization

- Differences between data types
- Data transfer cost
- Data security

Federated Learning !

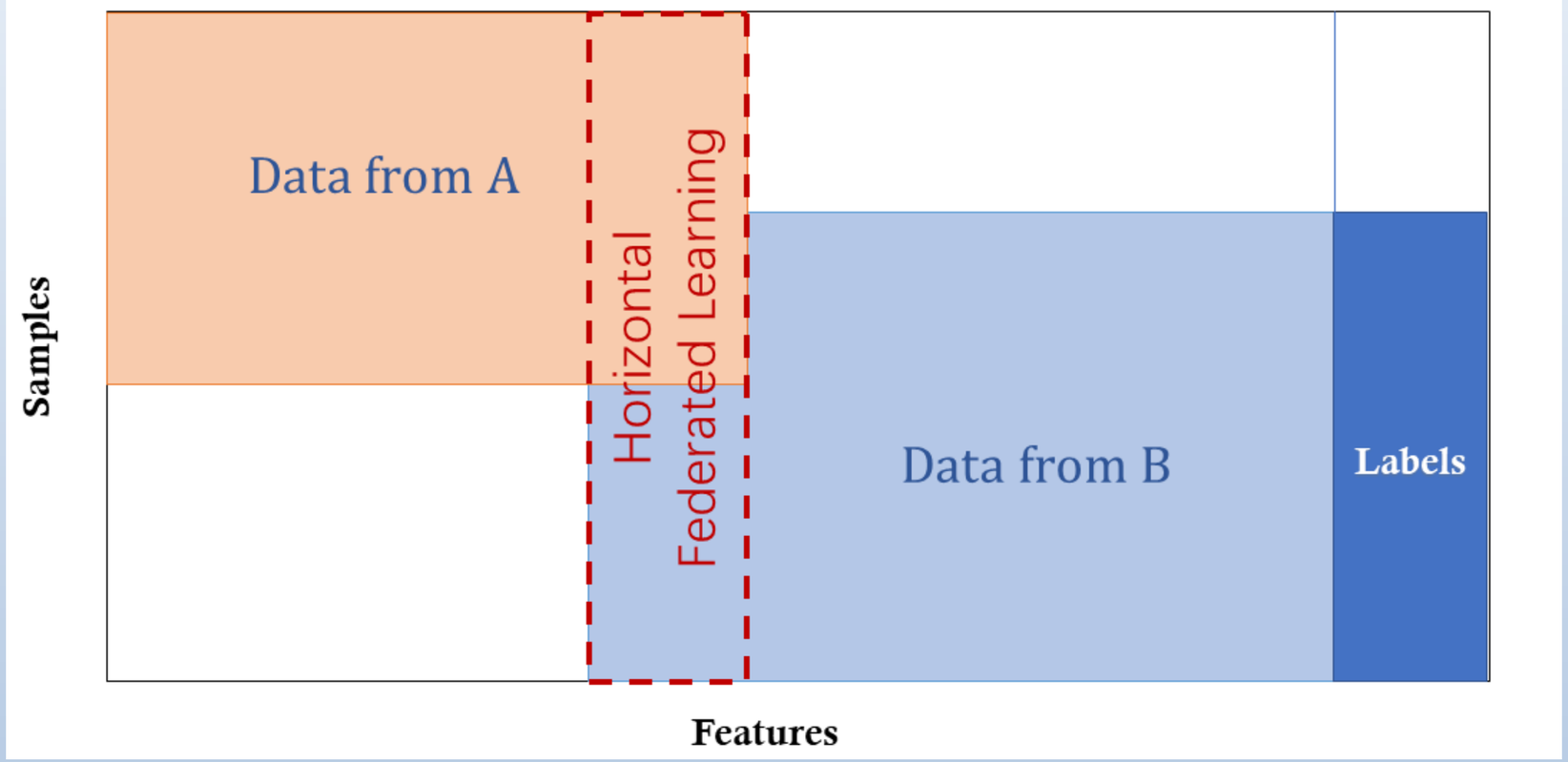
The main idea is to build machine learning models based on data sets that are distributed across multiple devices while preventing data leakage.

02. What do we share in common?



Horizontal Federated Learning

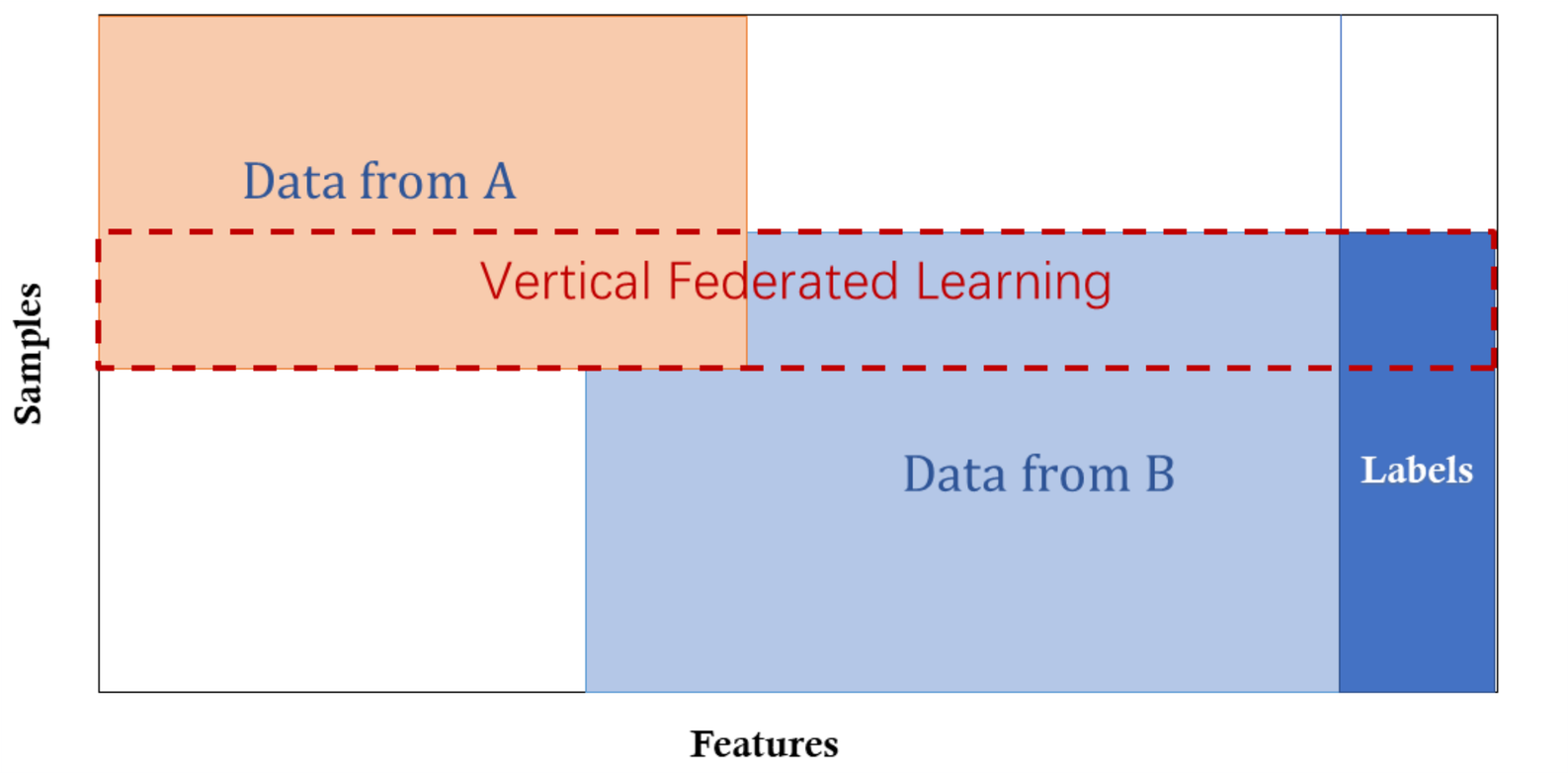
Horizontal federated learning, or sample-based federated learning, is introduced in the scenarios that data sets share the same feature space but different in samples



- Example: Two Regional banks Collaborating

Vertical federated learning

Vertical federated learning or feature-based federated learning is applicable to the cases that two data sets share the same sample ID space but differ in feature space.



- Example: Two Different Organizations in a Certain Area

03. Developing the Idea

Step 1. Define the goal.

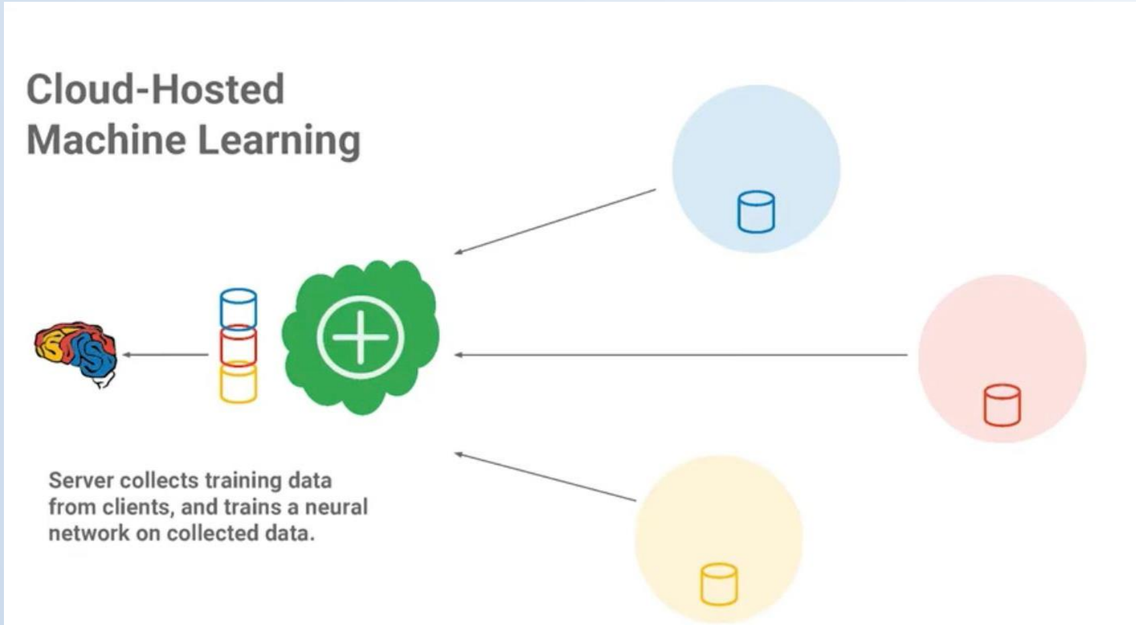
We want to train a global model with separate data associated to multiple users.

Considering the privacy constrains for each user's data, suggest a system to solve the issue.

Step 2. Start from the simple.

Gather Data in a Center → Feed Data to a model → Watch it Learn :)

Aggregation is directly performed on data, itself.



Step 3. Recognize the problems.

1. A lot of data people use is private. The server can observe any data it wants.
(No Confidentiality)

2. The server or an attacker can easily manipulate the data, degrading model's performance
(No Integrity)

Step 4. Improve the system

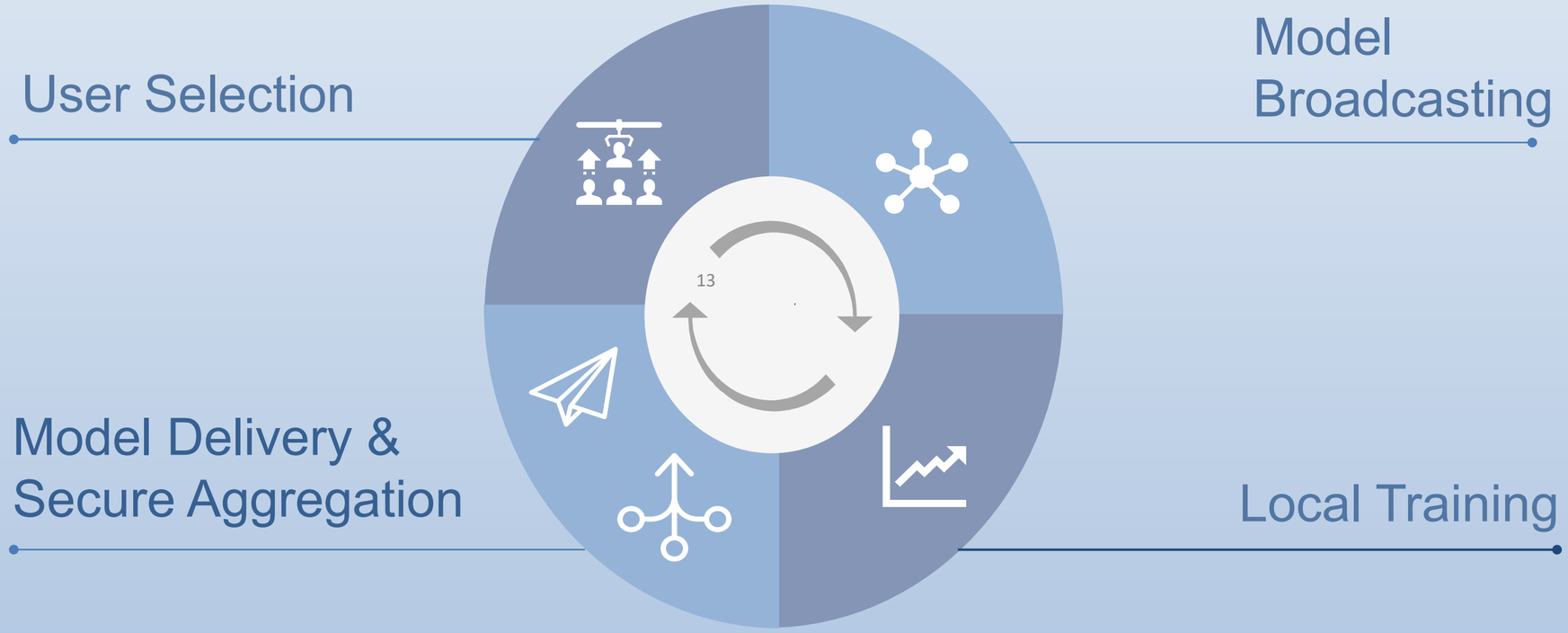
What if we send some information about the model instead of sending the data?

What can represent how a model is learning and evolving? **Gradient Flow! ∇**

Model Delivery ✓

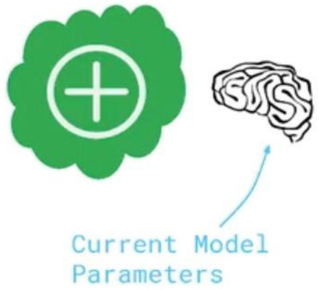
How is Federated Learning done?

The Art of Hidden Collaboration with the Motivation of Data Aggregation

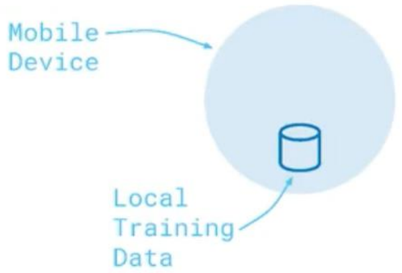


Federated Learning

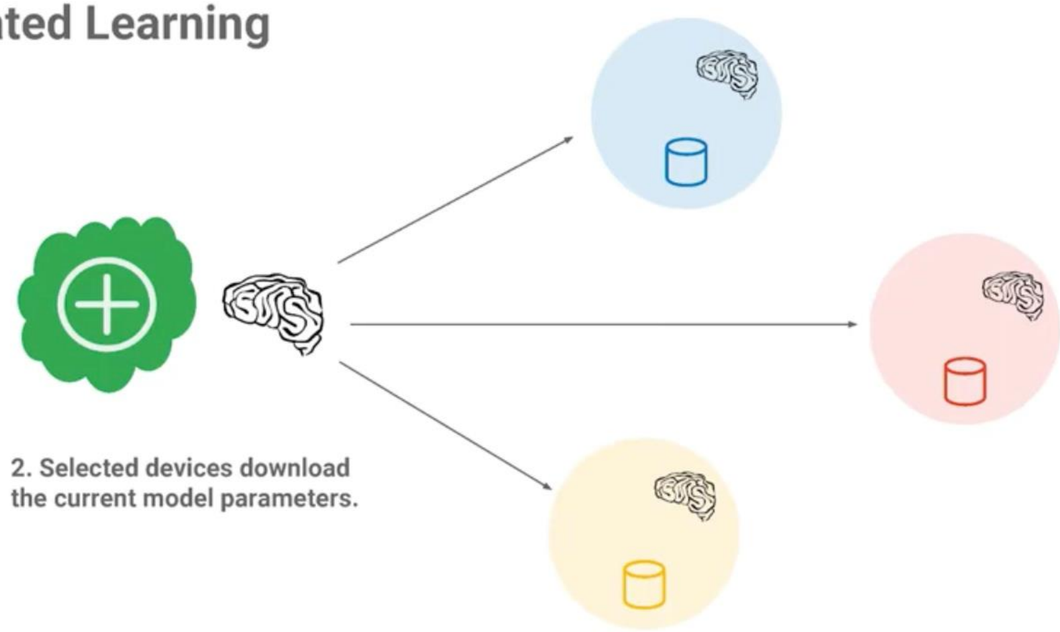
1. Server selects a sample of e.g. 1000 online devices.



Current Model Parameters

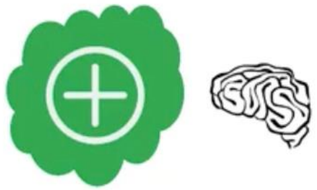


Federated Learning



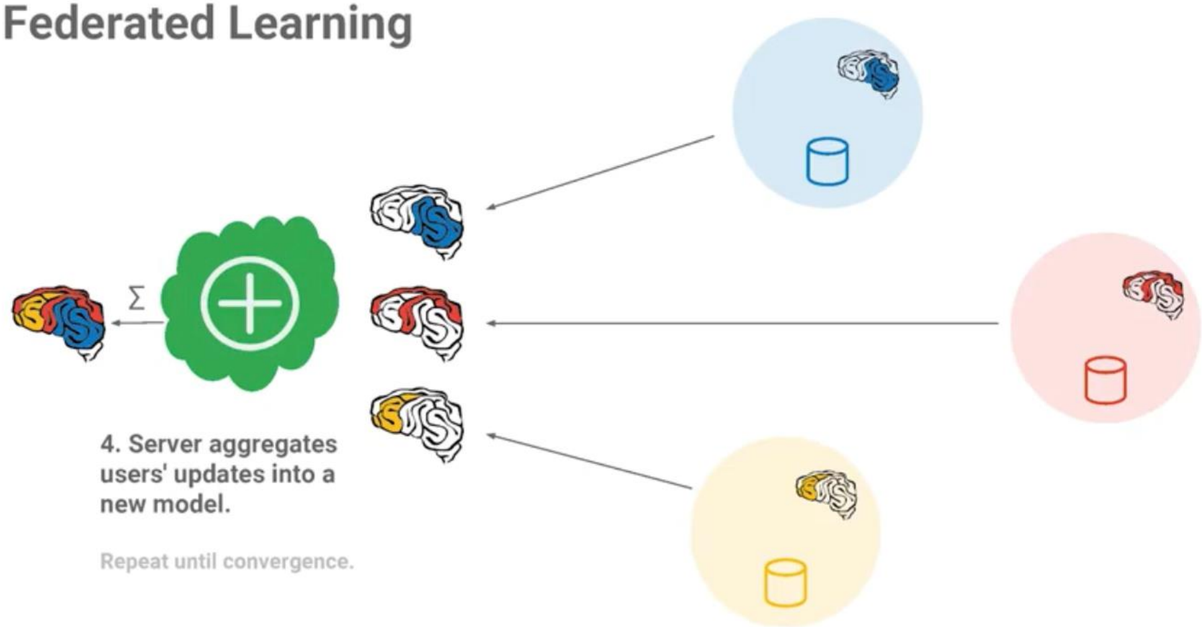
2. Selected devices download the current model parameters.

Federated Learning



3. Users run stochastic gradient descent on local training data

Federated Learning



4. Server aggregates users' updates into a new model.

Repeat until convergence.

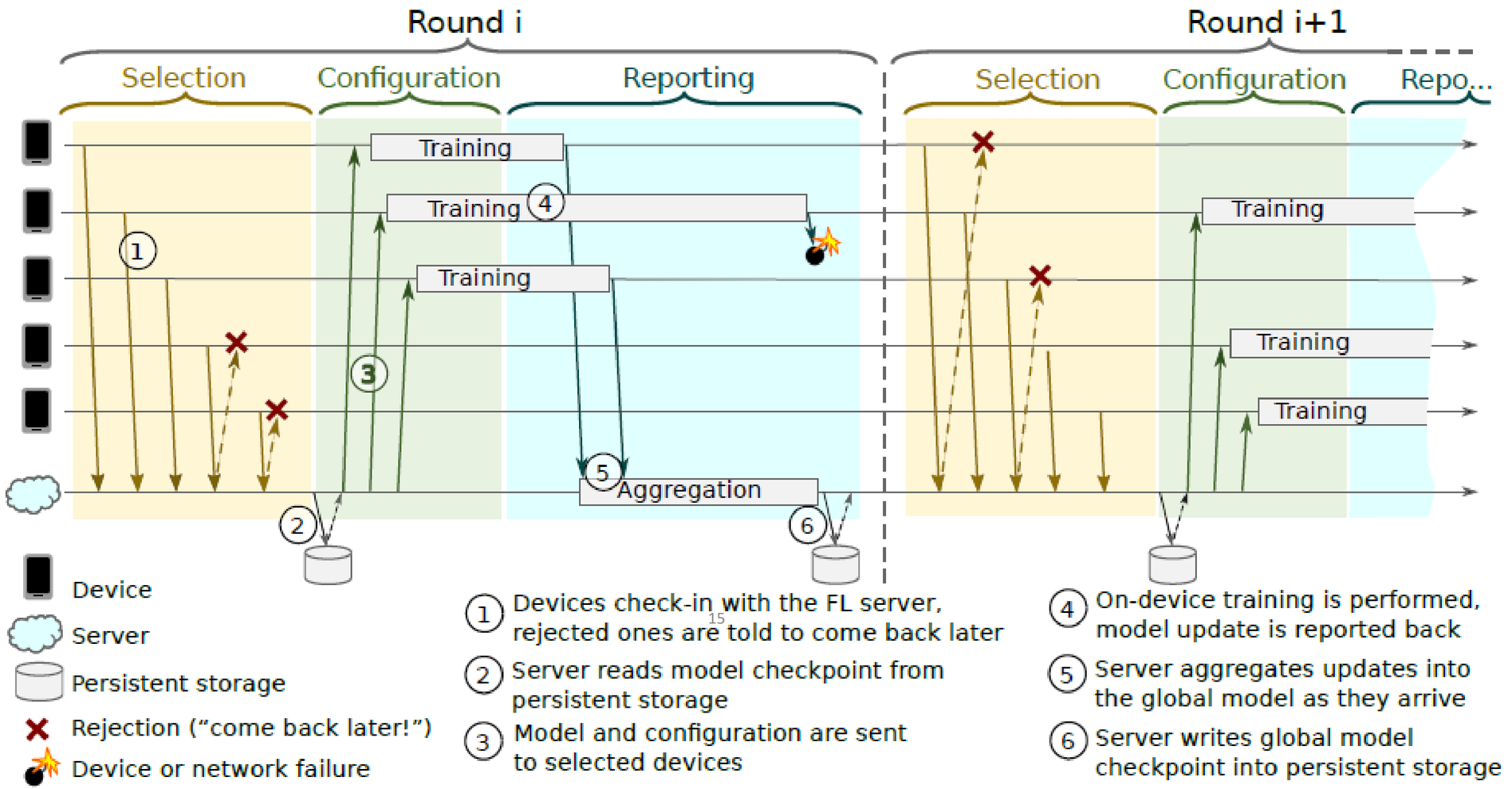
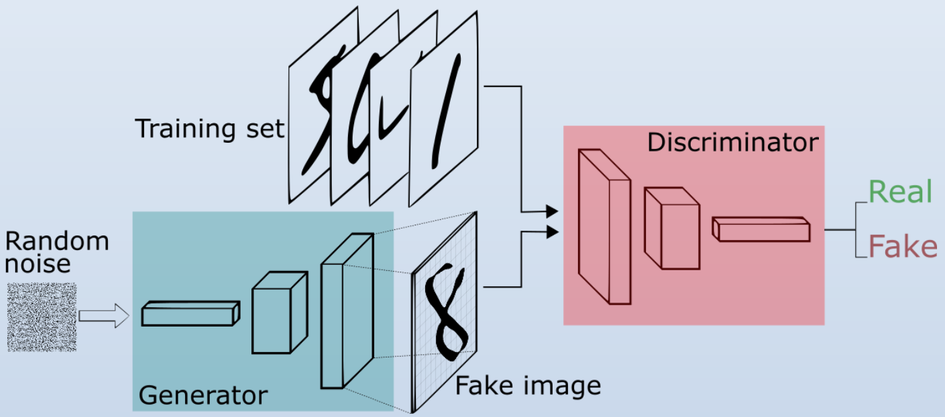
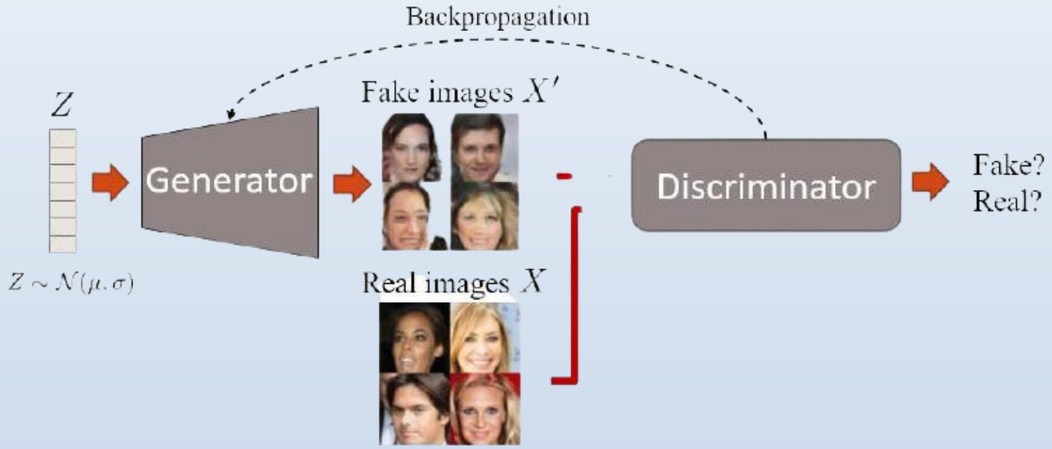


Figure 1: Federated Learning Protocol

Are the gradients secure enough?

GAN's learn to produce similar fake data from random noise in the latent space and the updates (gradient flows)





A possible attack idea is that a malicious server can train a GAN, according to the gradient flows received, and then broadcast the GAN model to the users.

If users find out the gradient of one another they can collaborate with fake generated data as well and try to trick the global model and eventually oneanother.

GAN attacks

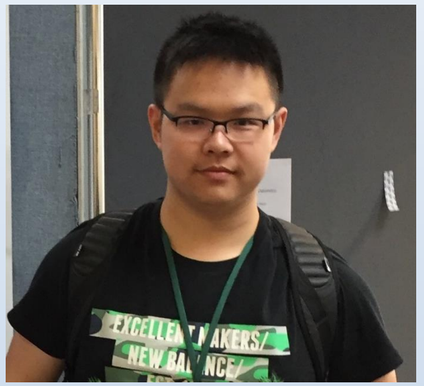
Ok. My training is ruined thanks to GANs 😒

but at least the exact data is still hidden right?

Well ... Actually, No 😬

Let me introduce our model. 😌

DLG (Deep Leakage from Gradient)



Ligeng Zhu, PHD student, MIT

L.Zhu, Z.Liu and S.Han published a paper on Dec. 2019, containing groundbreaking results.

They proved a dataset can be regenerated by only observing the gradients!

A closer look at DLG and data inference

At each round t, every node such as i, samples a minibatch $(x_{t,i}, y_{t,i})$, from it's own dataset and computes gradients. Assuming global parameter vector W_t at the beginning of round t, and local parameter vector $W_{t,i}$, and a differentiable ML model F:

Local training with global model

$$\nabla W_{t,1} = \frac{\partial L(F(x_{t,1}, W_t),\ y_{t,1})}{\partial W_t}$$

Person #1

...

$$\nabla W_{t,i} = \frac{\partial L(F(x_{t,i}, W_t),\ y_{t,i})}{\partial W_t}$$

Person #i

...

$$\nabla W_{t,N} = \frac{\partial L(F(x_{t,N}, W_t),\ y_{t,N})}{\partial W_t}$$

Person #N

Aggregating local gradients into a global gradient

$$\nabla W_t = \frac{1}{N}\sum_{j=1}^{N} \nabla W_{t,j}$$

Updating global model

$$W_{t+1} = W_t - \eta\, \nabla W_t$$

Problem:

Given local gradients $\nabla W_{t,k}$ from user k:
reveal the data associated to the user, $(x_{i,k}, y_{i,k})$

1. Randomly initialize dummy data pairs (x', y'):

2. Calculate the dummy gradient according to global model and dummy data pairs

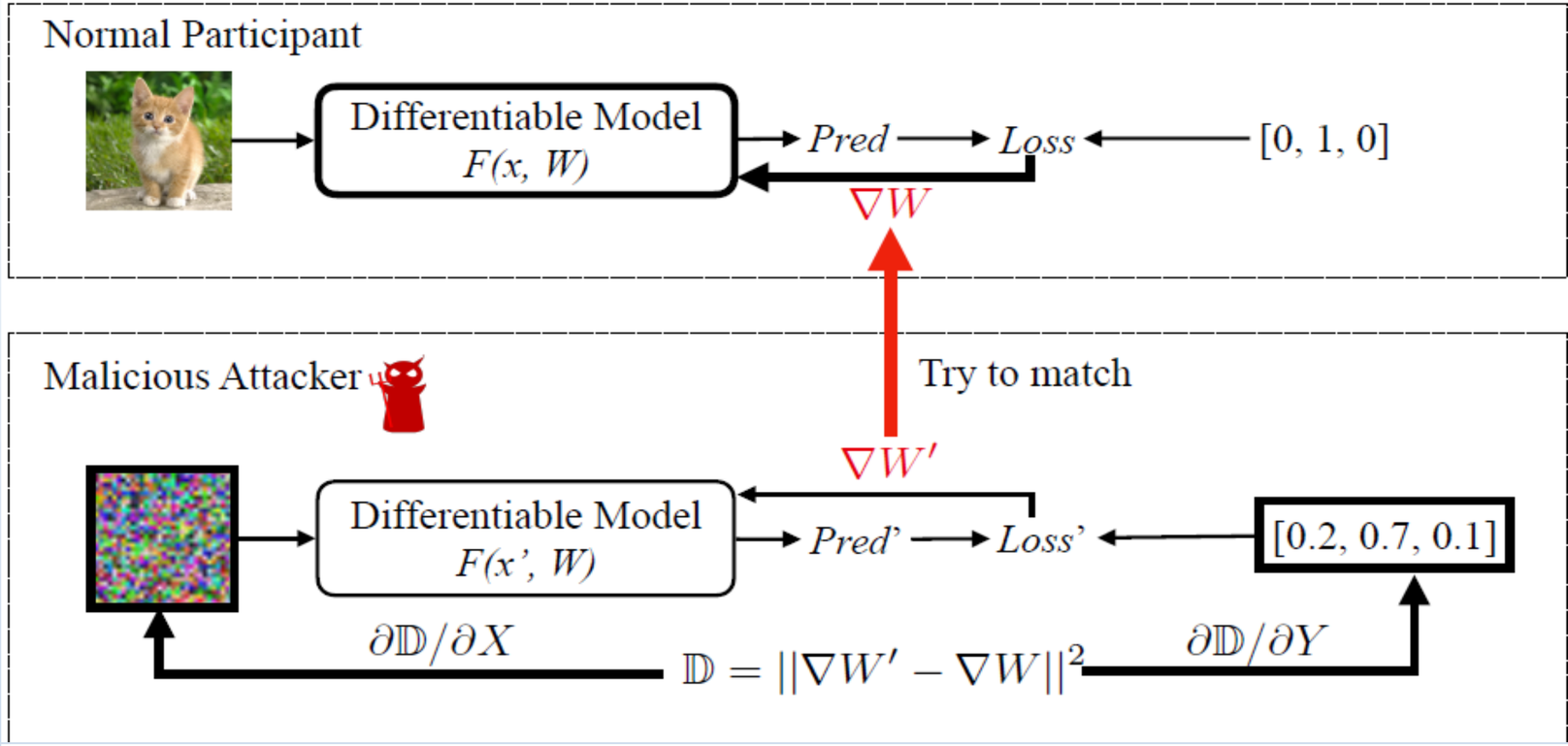
$$\nabla W' = \frac{\partial L(F(x', W), y')}{\partial W}$$

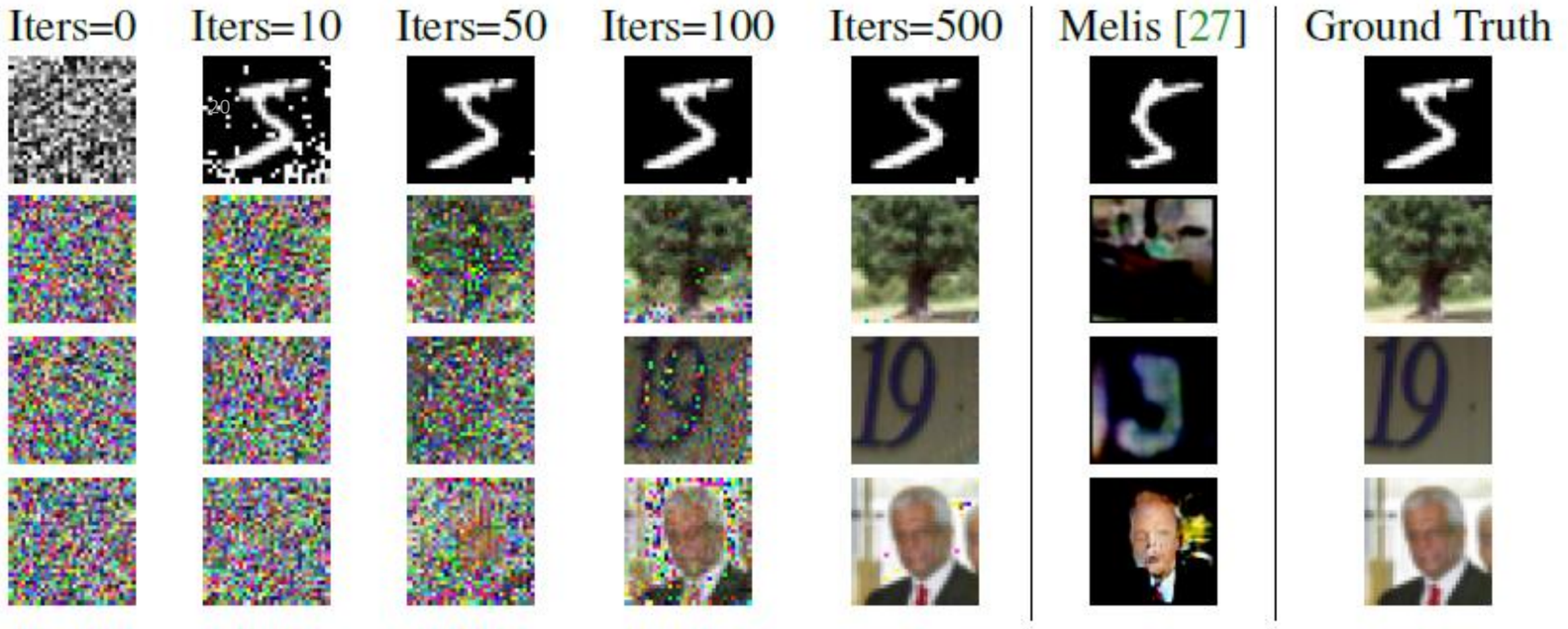
3. Match the dummy gradient and actual gradient (solve this optimization problem)

$$x'^*, y'^* = argmin_{x',y'} \|\nabla W' - \nabla W\|^2 = argmin_{x',y'} \left\| \frac{\partial L(F(x_{t,i}, W_t),\ y_{t,i})}{\partial W_t} - \nabla W \right\|^2$$

Gradient Matching

Complete Inference on Gradient Descent!





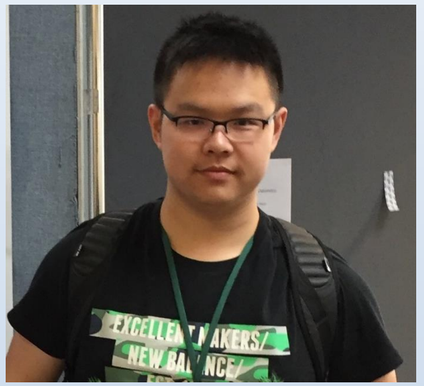
Wow, impressive work.

But can you infer data from batch gradients as well?

Yes. But a bit slower

One possible reason is batched data with a batch size of N has N! permutations and gradient directions vary according to the order which data is fed in to the model

For faster convergence we update a single dummy pair instead of a batch in each iteration



Matching bath gradients results

	BS=1	BS=2	BS=4	BS=8
ResNet-20	270	602	1173	2711

Table 1: The iterations required for restore batched data on CIFAR [21] dataset.

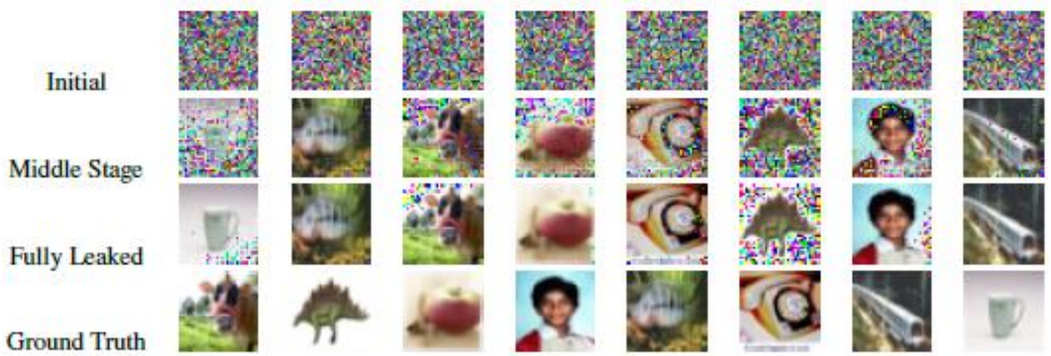
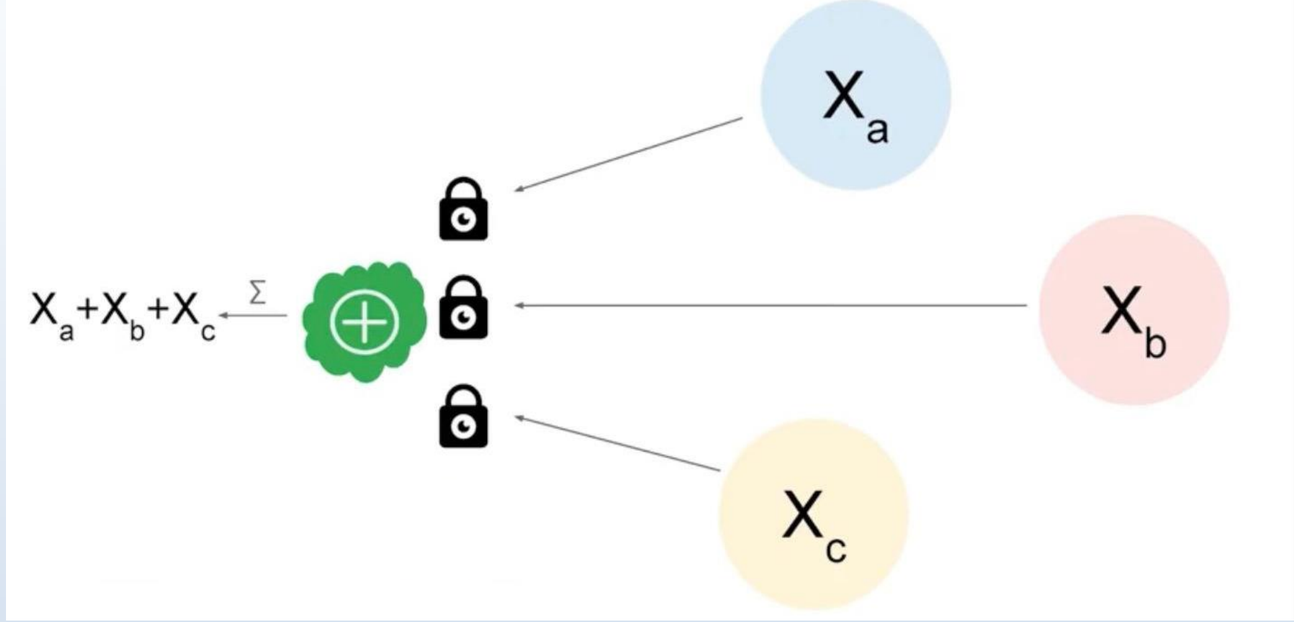


Figure 4: Results of deep leakage of batched data. Though the order may not be the same and there are more artifact pixels, DLG still produces images very close to the original ones.

Back to Federated Learning …



So the server should learn from the sum of the inputs
not from each individually

This is what's known as "Secure Aggregation"

Secure aggregation, is based on a security concept called "Secret Sharing"

04. How to share a secret?

Secret Sharing

The simple solution for sharing a secret is to distribute copies of it on a secure channel to authorized users.

- But what if no one has to have **full access** to the secret?

For Example:

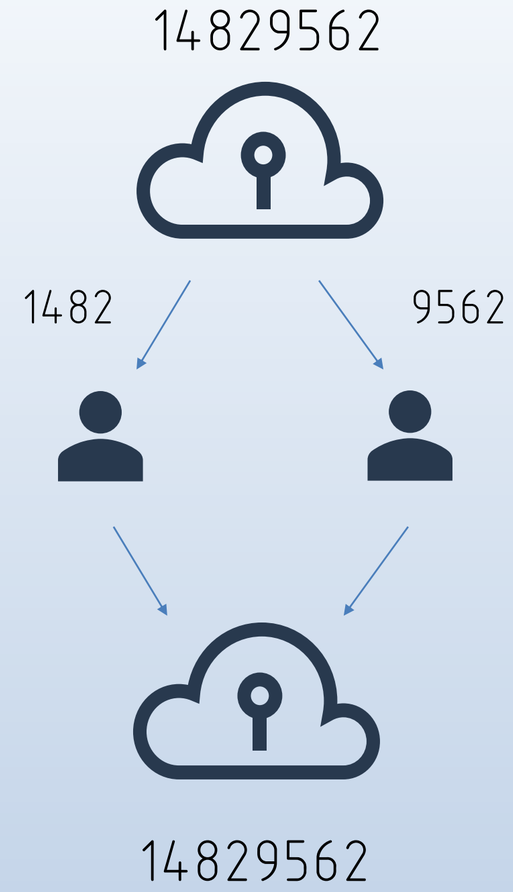
01 **Missile Launch Codes**

02 **Passwords for Large Transactions**

Solution #1- Just Split It !

Problem :

Having partial information **narrows down the search space** for a brute force attack to a smaller subspace.



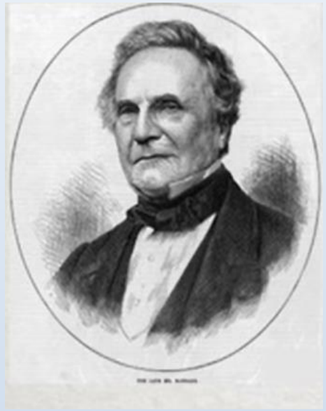
Perfect Secrecy:

The one who has a share of a secret shouldn't find it out easier than a person that has no share.

So whats the point of having a share?

Sharing should help decentralizing crucial information but also hiding (any part of it) from individuals.

Solution #2- Add a tablespoon of NOISE



S

$+$

N_A

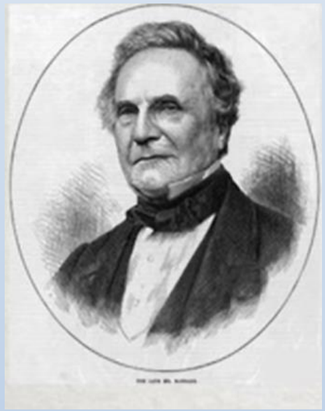
$=$

N_B

I have no clue of the secret! Unless I have N_a

B

$-$



A

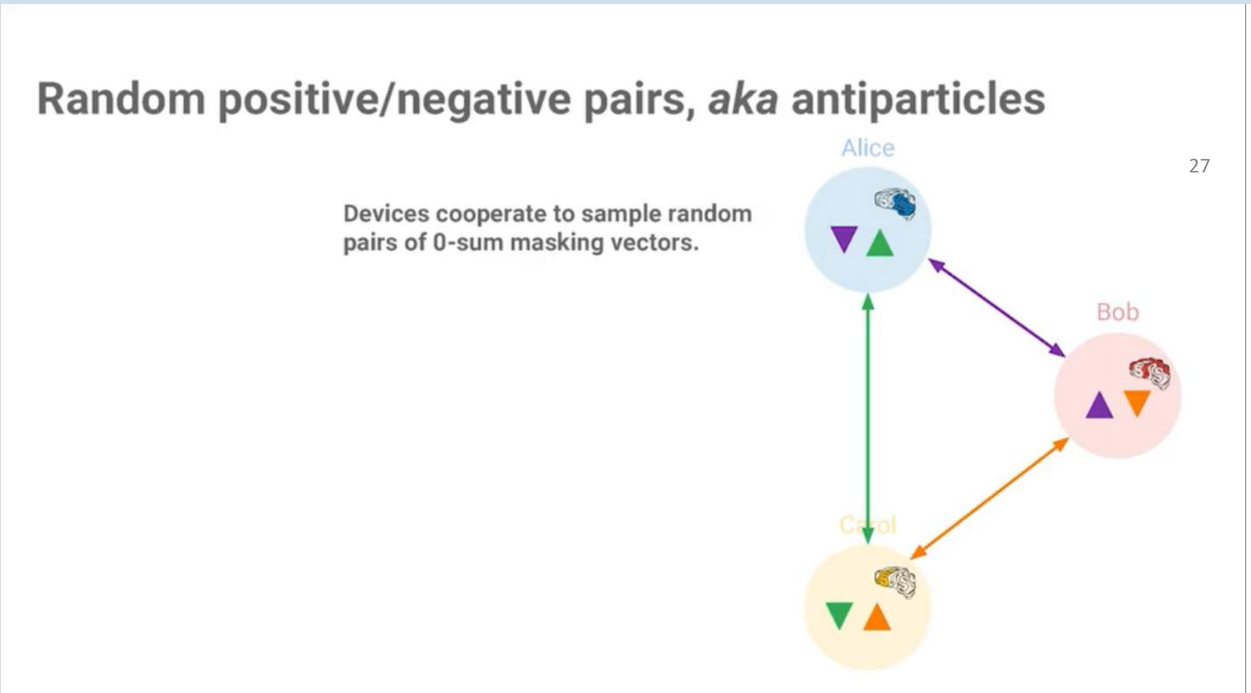
Only if parties don't collaborate individually!

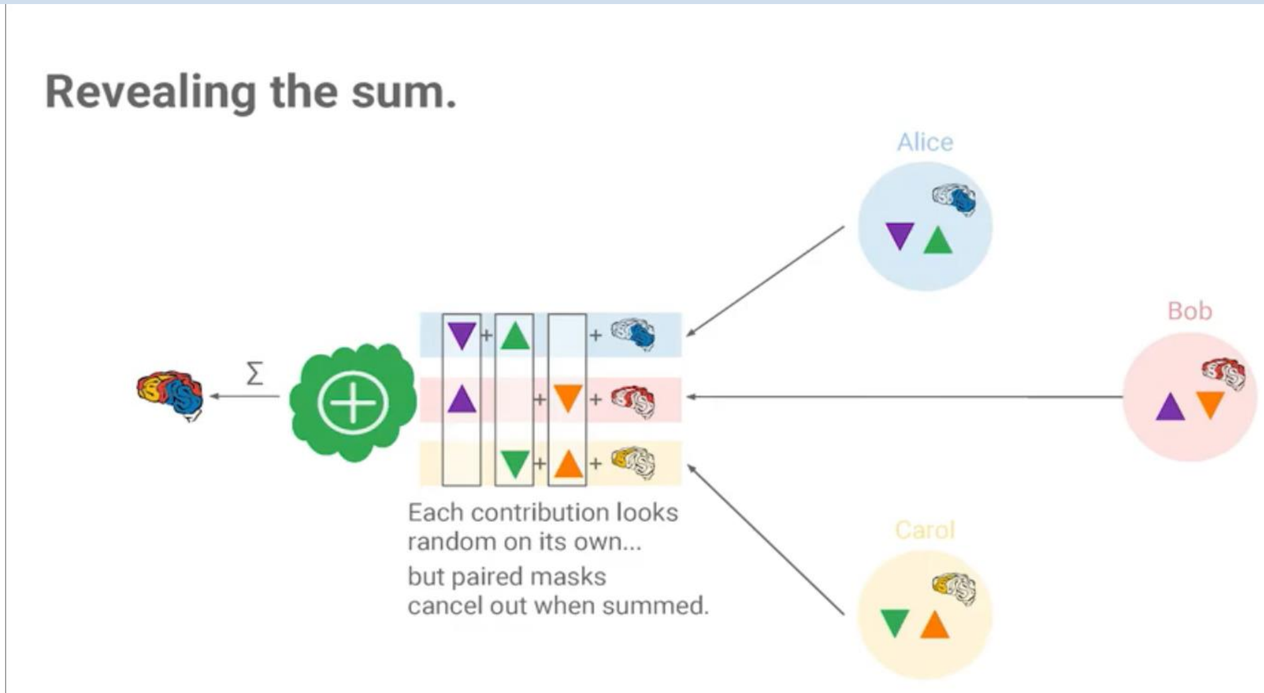
Confidentiality ✓
Decentralization ✓
Integrity ✗

Applying Solution #2- Masking Vectors

Generate N random noise vectors (masking vectors) for N users in the system

Assign a pair to each masking vector by negating it. Give this pair to another user





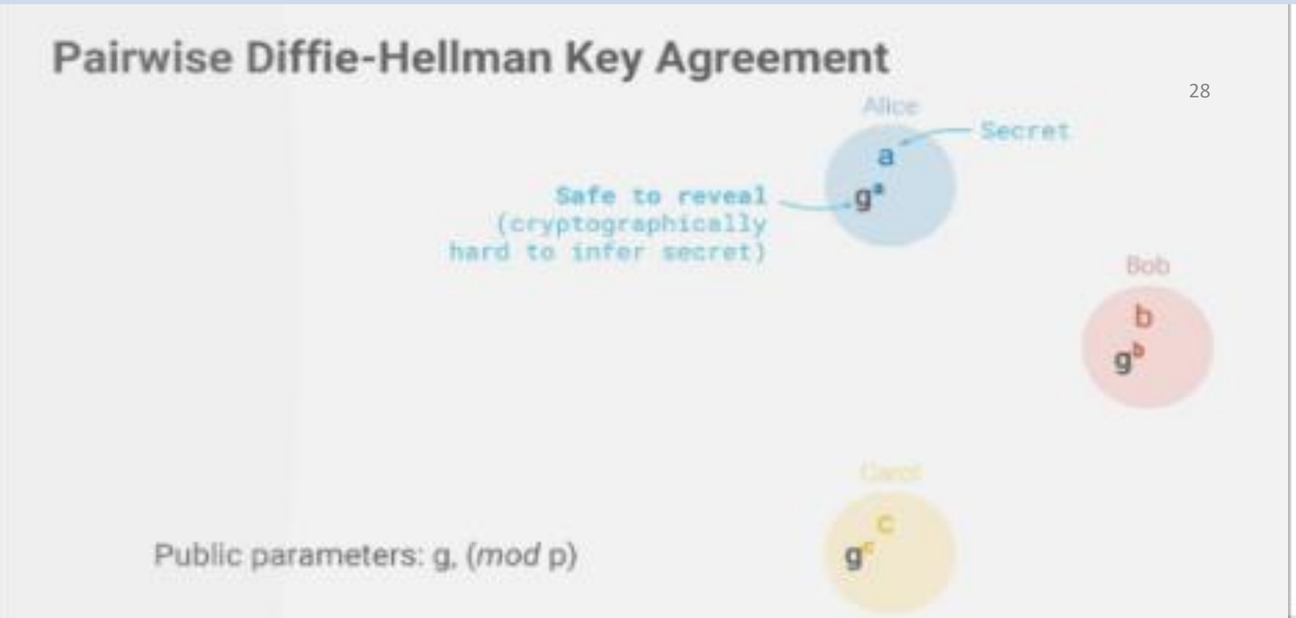
Problem: Masking vectors are large!

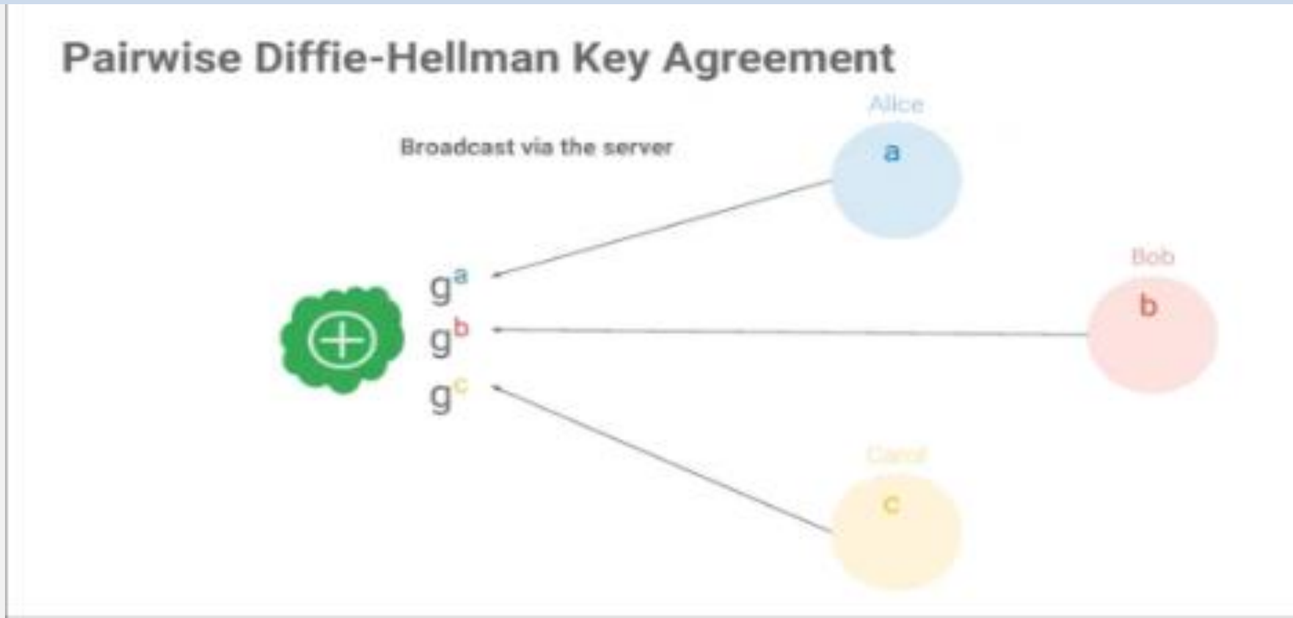
How to agree on them efficiently?

01 Deep Learning → Large parameter space → Long gradient vector → Slow training

Long masking vector ↗

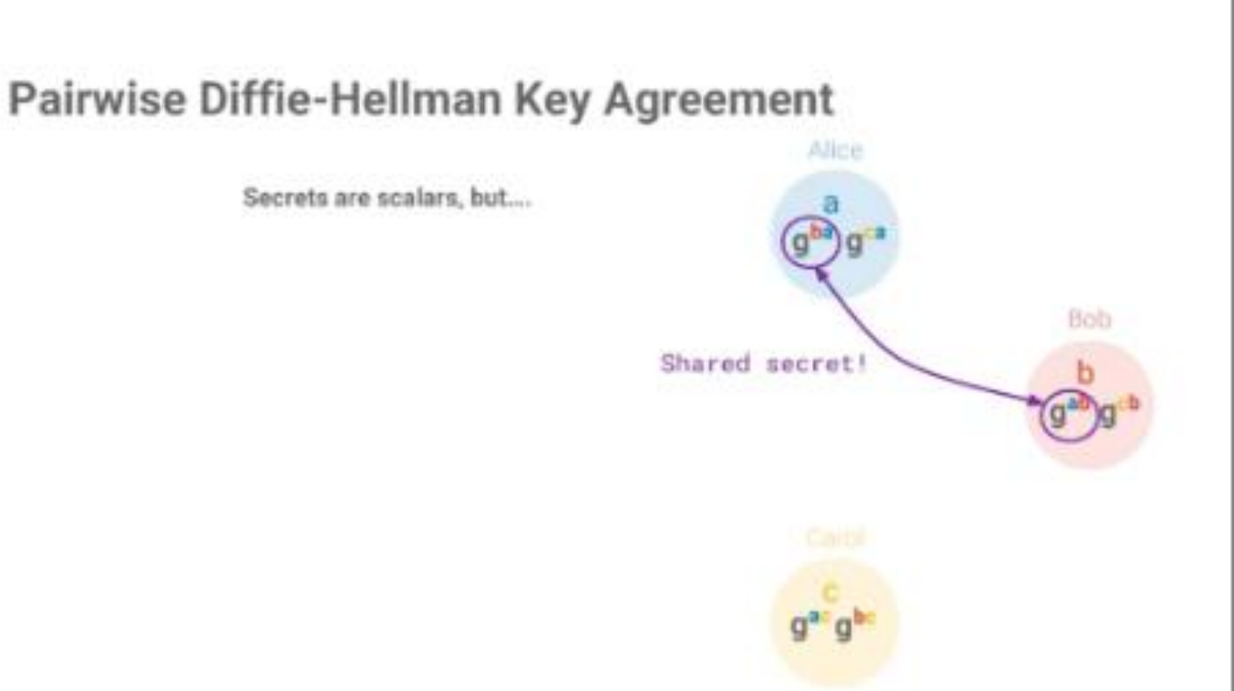
Solution: Pairwise Diffie Hellman





OK, so we have agreed on Keys but how to generate these equal long masks?

We use the Diffie-Hellman share as the input to a PRNG
(Pseudo random number generator)



Efficiency ✓

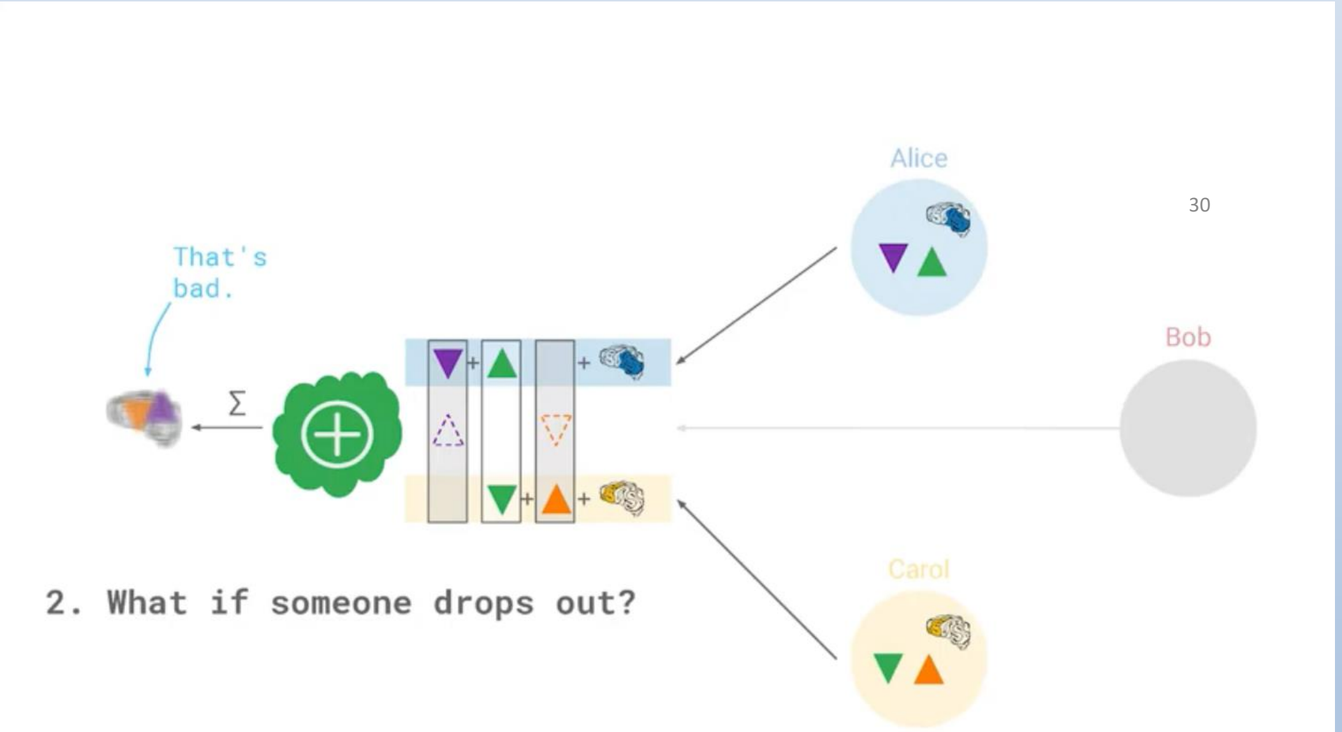
Eliminate need for peer-to-peer communication ✓

Easier recovery ✓

Problem: Dropout

02 Mobile Users → Channel loss, Interference, … → User Dropout

Pure output can’t be recovered due to presence of noise



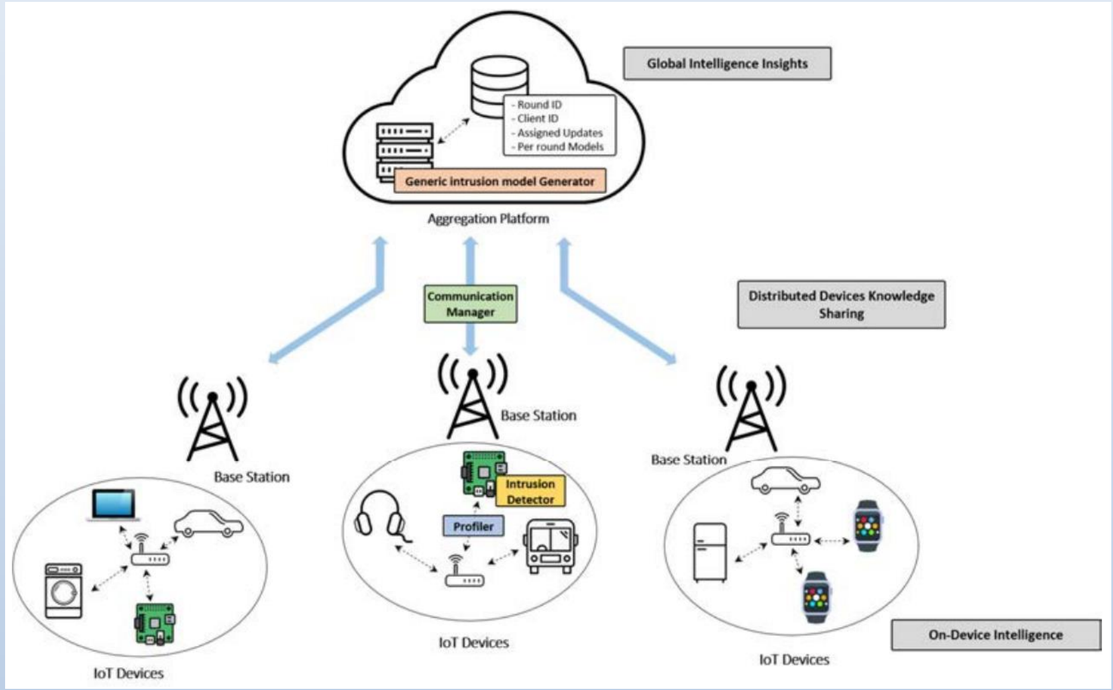
A large system like this shouldn’t rely on a single point of failure.

K out of N users should be able to recover the secret

05. Applications & Challenges

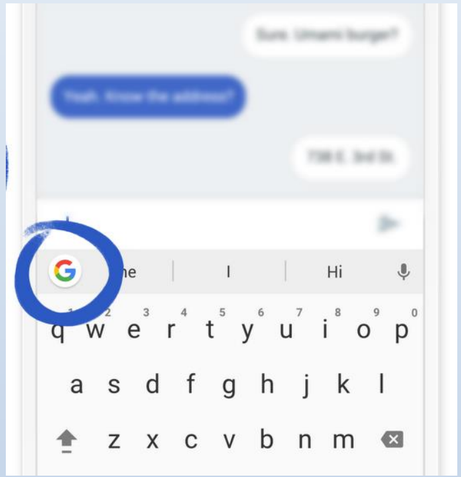
Application of federated learning to IoT system in smart city

1. Data application scenarios under IoT
2. Blockchain federated learning (Block FL)
3. Federated learning for edge computing

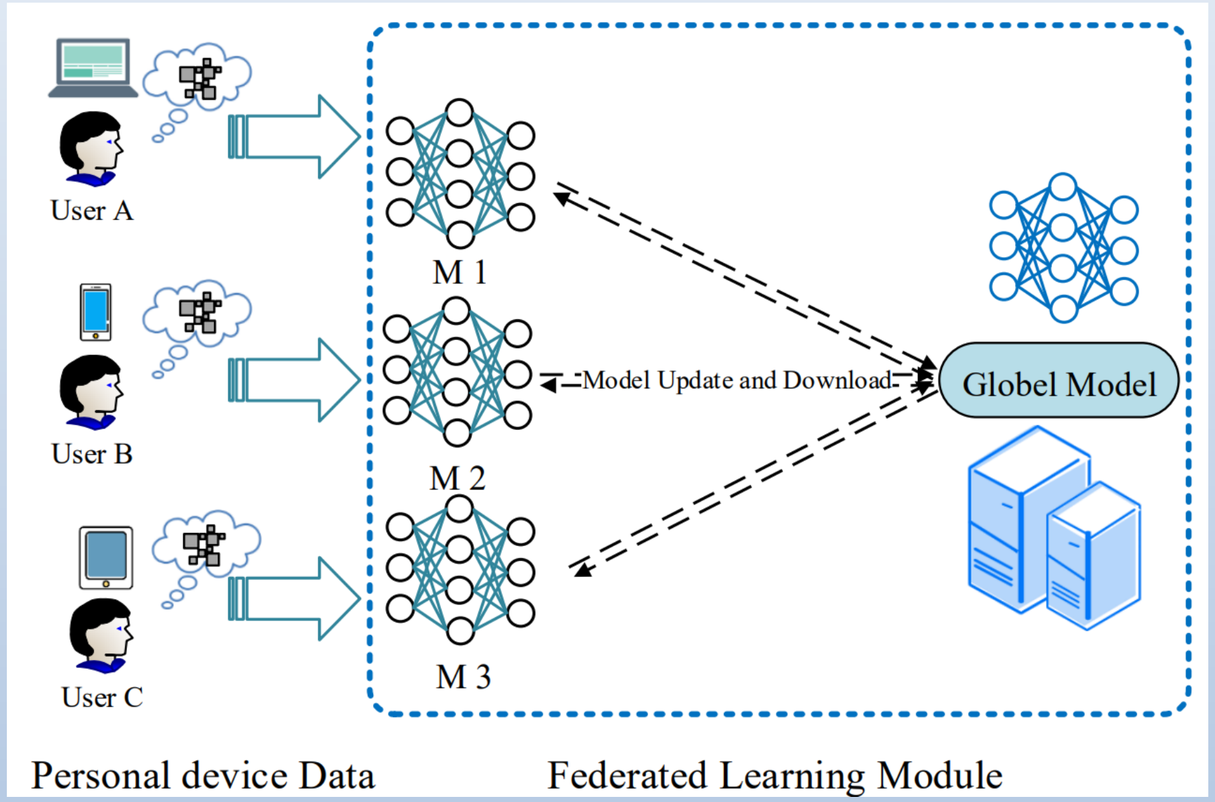


Learning over smart phones

1. Smart phones keyboard (i.e. Gboard)
   1) Tap typing
   2) Gesture typing
   3) Auto-correction
   4) Word prediction
   5) Emoji prediction
   6) GIF prediction



2. Smart phones voice assistant (i.e. Siri)
   1) Speech to text
   2) Noise cancelation
   3) And more…

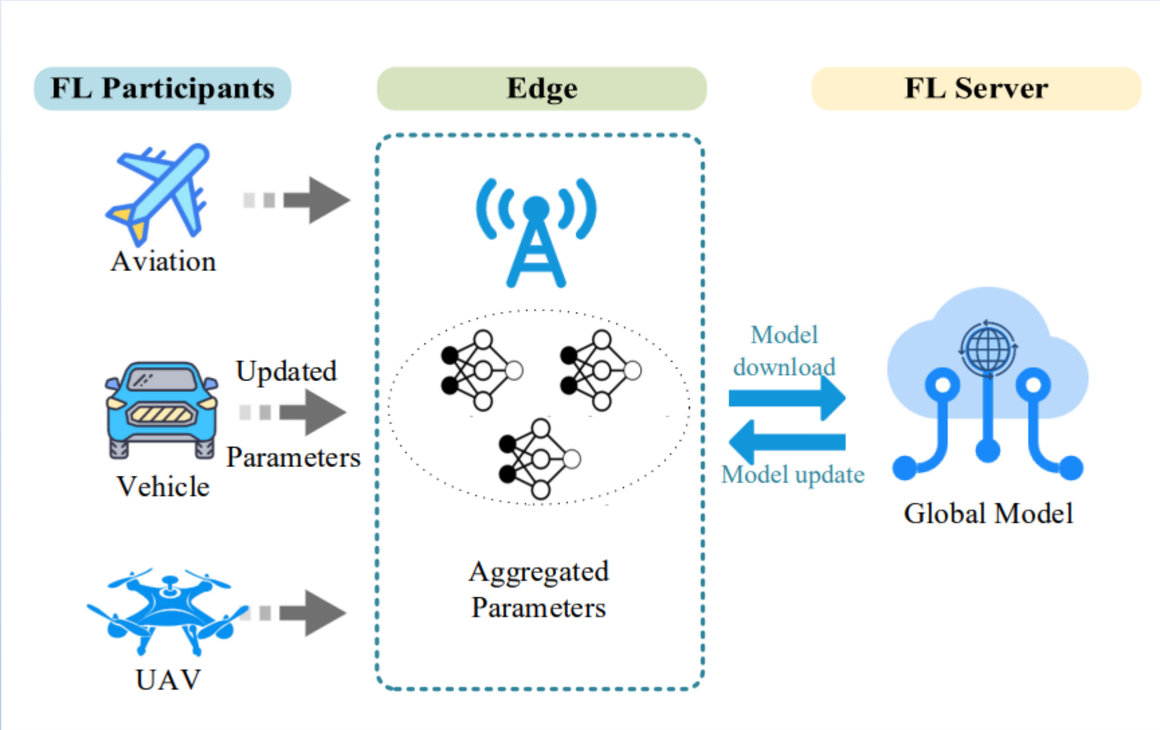


Federated learning to intelligent transportation system in smart cities

1. Combination of federated learning and vehicle system
   1) Vehicle communication
   2) Electric vehicle
   3) Autonomous vehicle

2. Combination of FL & aviation system
   1) Aircraft
   2) Unmanned aerial vehicle(UAV)

3. Challenges and problems
   1) Communication and calculation costs
   2) Privacy protection
   3) Energy issues

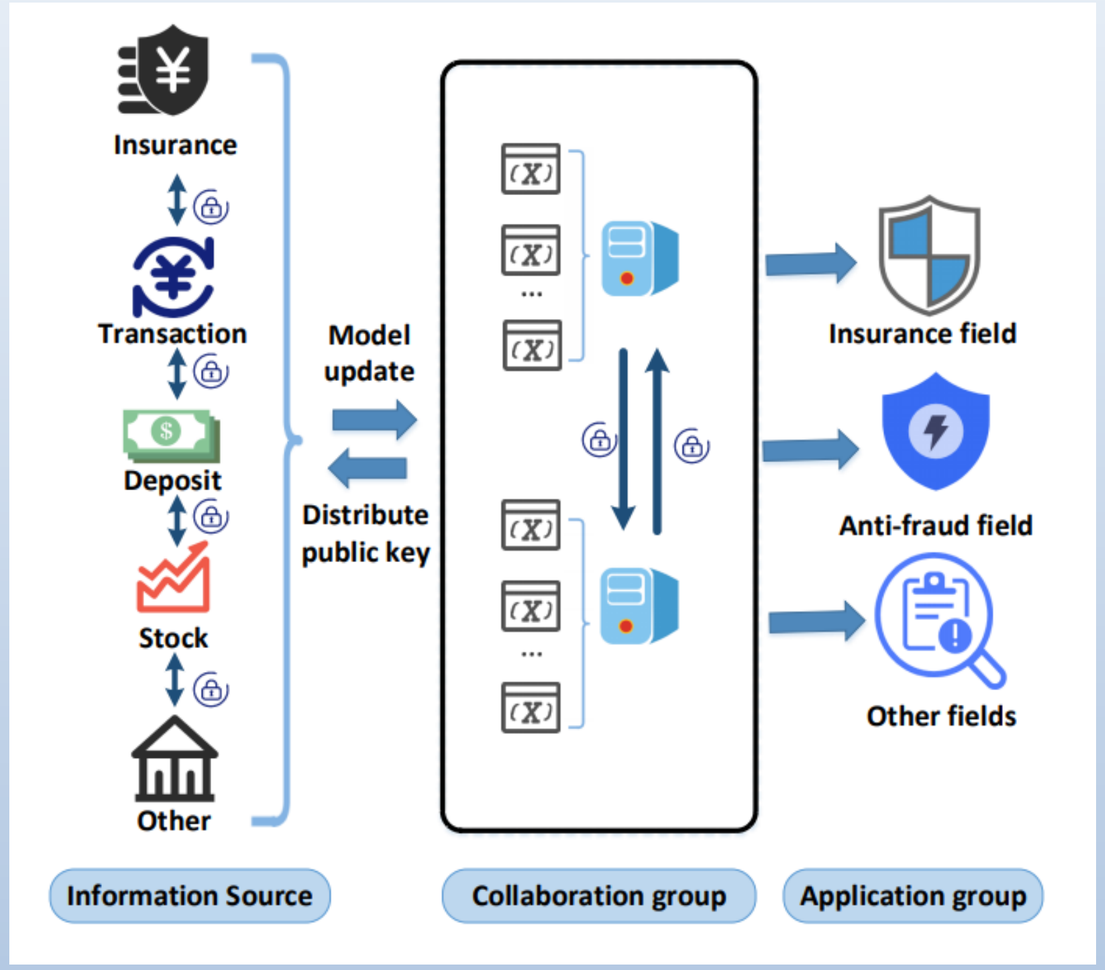


Federated learning in the financial field

1. Federated learning in the field of financial fraud
   1) Development of ML
   2) Traditional FL
   3) Bilateral privacy-protected FL

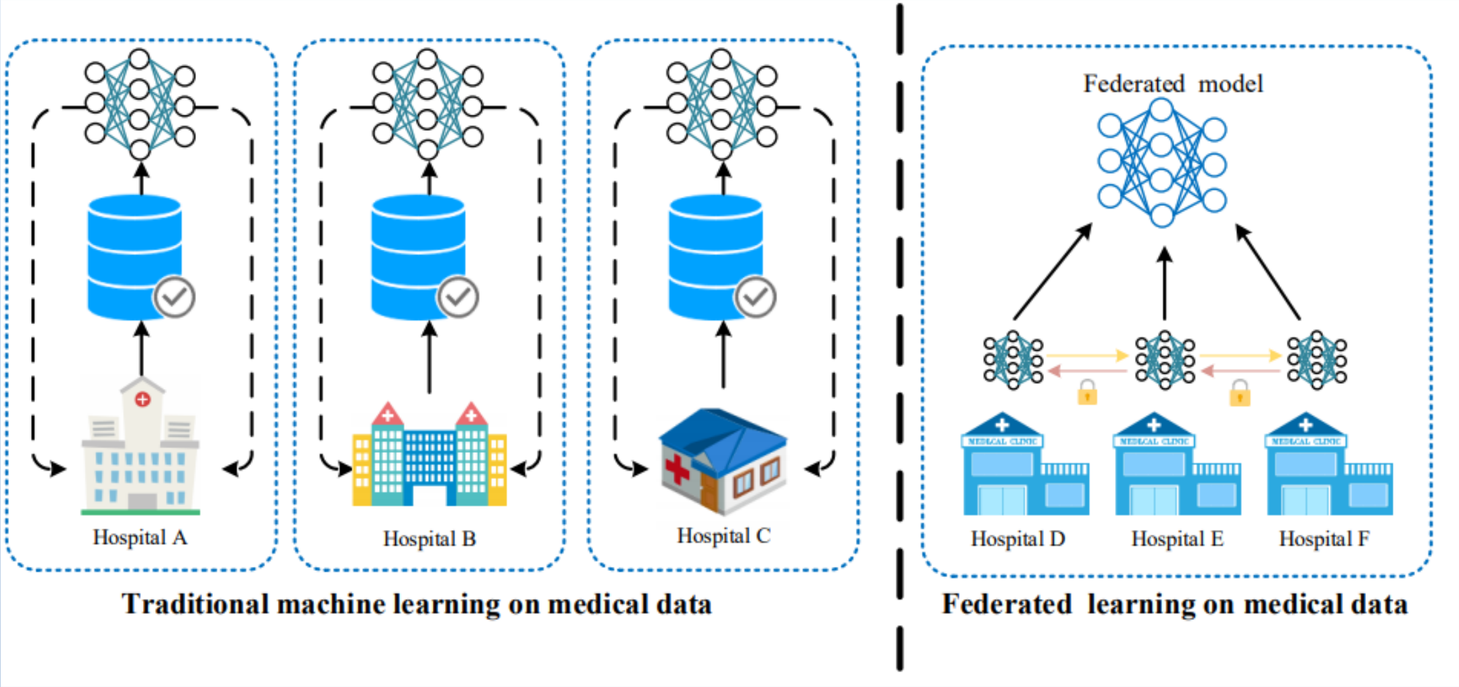
2. Federated learning in the field of insurance
   1) Isolated data islands

3. Challenges and problems
   1) Statistical challenges
   2) Incentive mechanism



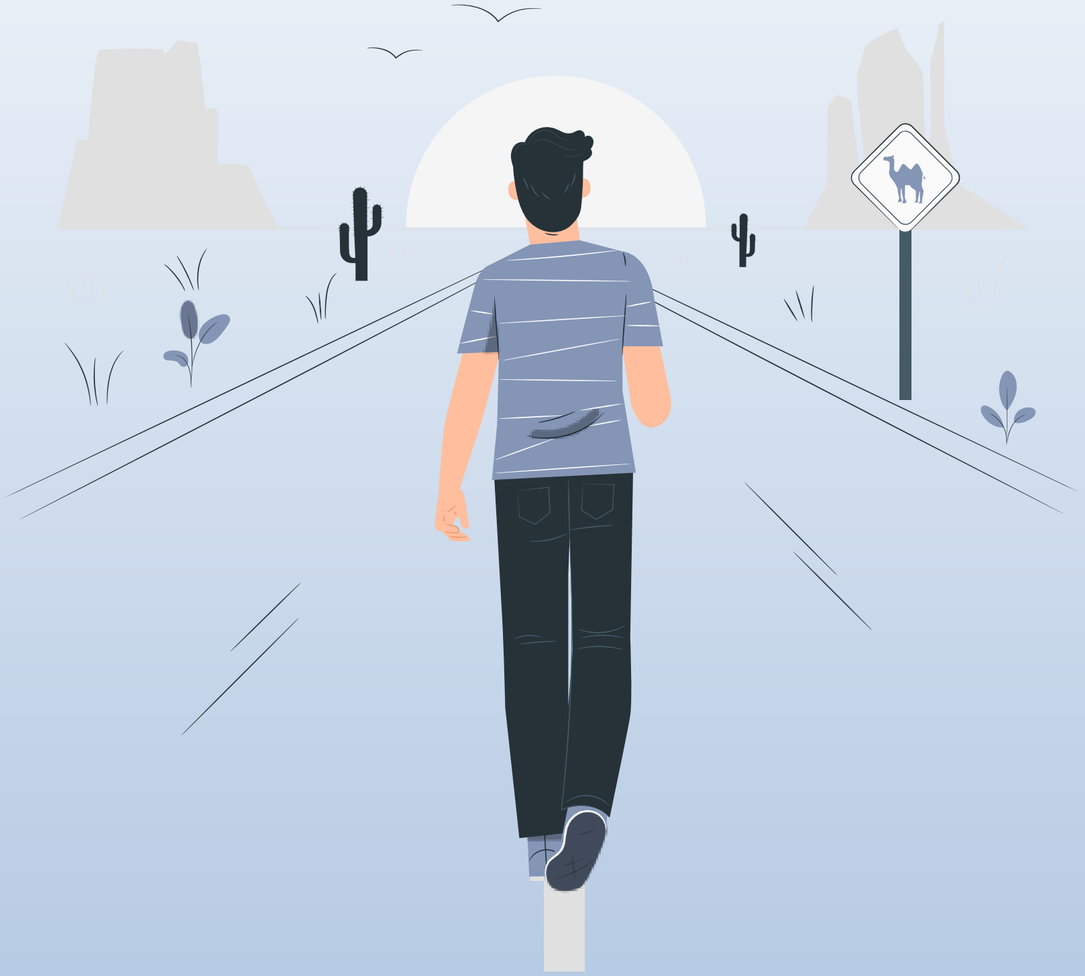
Application of federated learning to the medical field

1. Medical Privacy
2. Drug Development
3. Disease Prediction



The future development and direction of federated learning

- Defense against Attacks
- Algorithm Efficiency
- Technology Application



Thanks for your attention!

Papers:

1. Federated Learning; Concepts and Applications, Q.Yang, Feb 2019
2. Towards Federated Learning At Scale, K.Bonawitz & Google AI Team
3. Practical Secure Aggregation on FL from user-held data, K. Bonawitz & Google AI Team

Credits:

Design and Icons:

1. stroyset.com
2. icons8.com/

Technical part inspired by Google ACM Conference on Secure Aggregation - 2017