ECEN 685 Machine Learning in CyberSecurity

Dr. Mahmoud Nabil
mnmahmoud@ncat.edu

North Carolina A & T State University

October 28, 2020

Talk Overview

1. Why Federated Learning?

Outline

Why Federated Learning?

Enables multiple actors to build a common machine learning systems without centralizing data and with privacy by default.

[1] https://www.slicktext.com/blog/2019/10/smartphone-addiction-statistics/

Why Federated Learning?

Enables multiple actors to build a common machine learning systems without centralizing data and with privacy by default.

- Mobile devices are personal computer
  - As of June 2019, 96% of Americans own a cellphone of some kind [1]
- Plethora of sensors
- Privacy issues.

[1] https://www.slicktext.com/blog/2019/10/smartphone-addiction-statistics/

Why Federated Learning?

Enables multiple actors to build a common machine learning systems without centralizing data and with privacy by default.

- Mobile devices are personal computer
  - As of June 2019, 96% of Americans own a cellphone of some kind [1]
- Plethora of sensors
- Privacy issues.

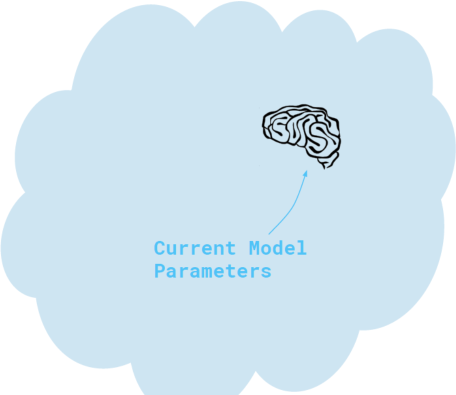
Challenges

- Deep Learning is non-convex
- millions of parameters
- complex structure

[1] https://www.slicktext.com/blog/2019/10/smartphone-addiction-statistics/

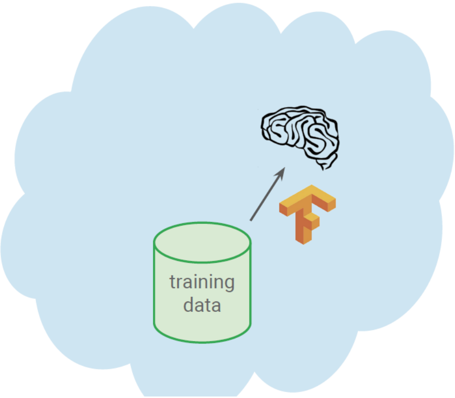
Current Machine Learning as a Service for Mobile Devices

The model lives in the cloud.

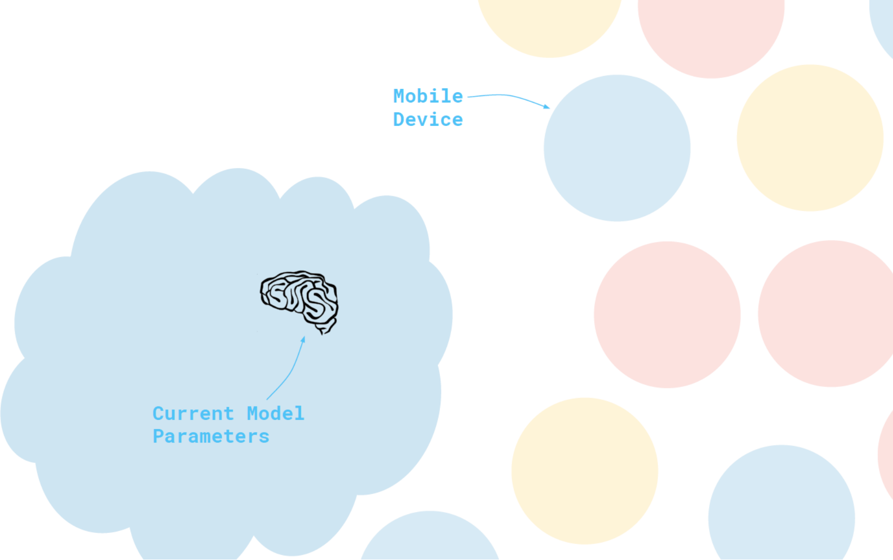


Current Machine Learning as a Service for Mobile Devices

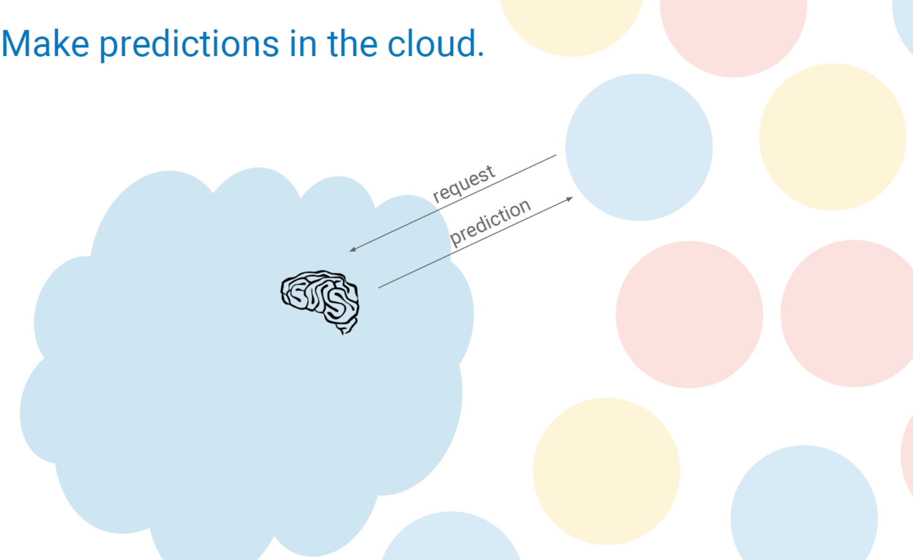
We train models in the cloud.



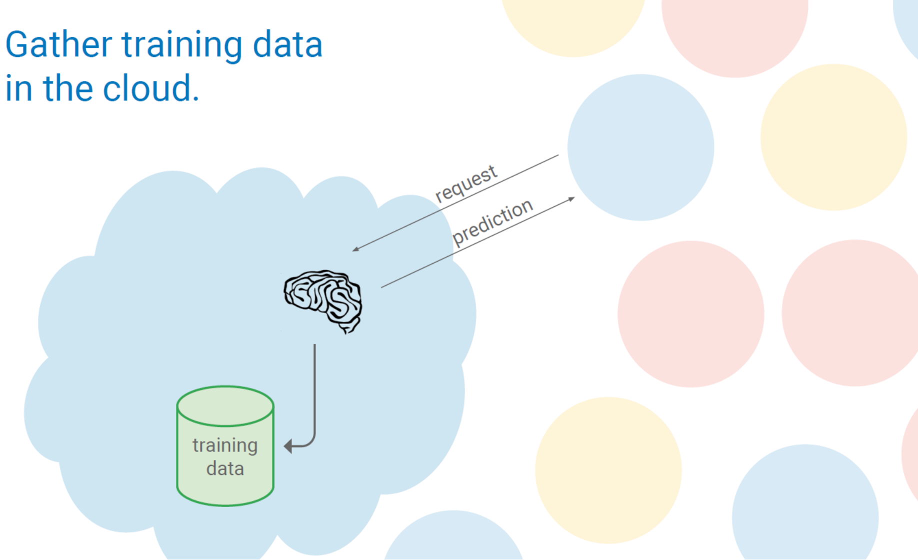
Current Machine Learning as a Service for Mobile Devices



Current Machine Learning as a Service for Mobile Devices

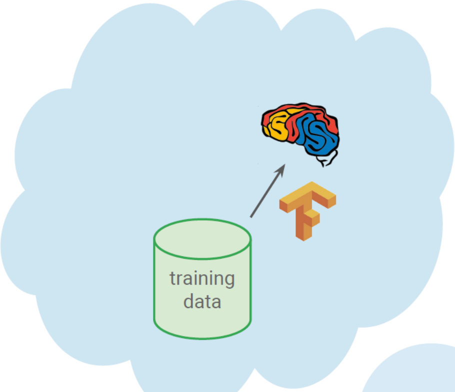


Current Machine Learning as a Service for Mobile Devices



Current Machine Learning as a Service for Mobile Devices

And make the models better.

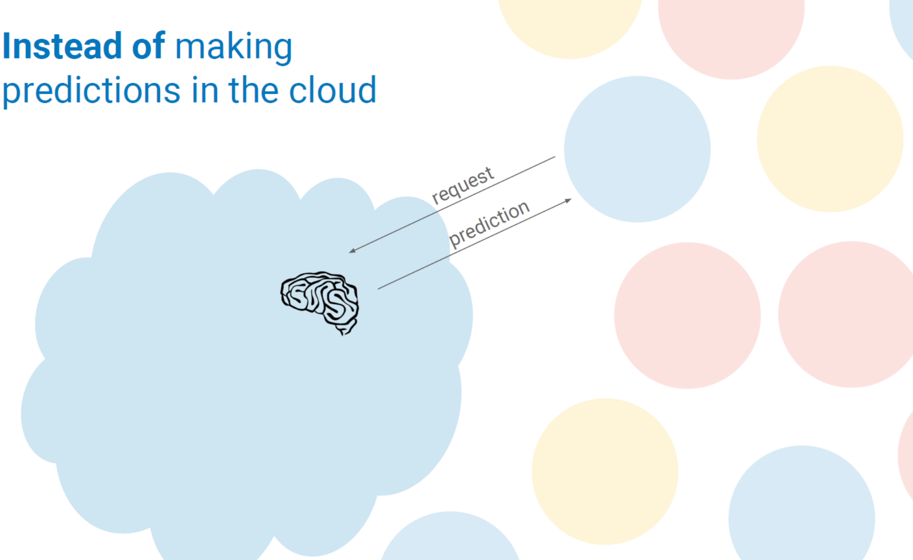


On-device inference

On-device inference is using a cloud-distributed model to make predictions directly on an edge device without a cloud round-trip

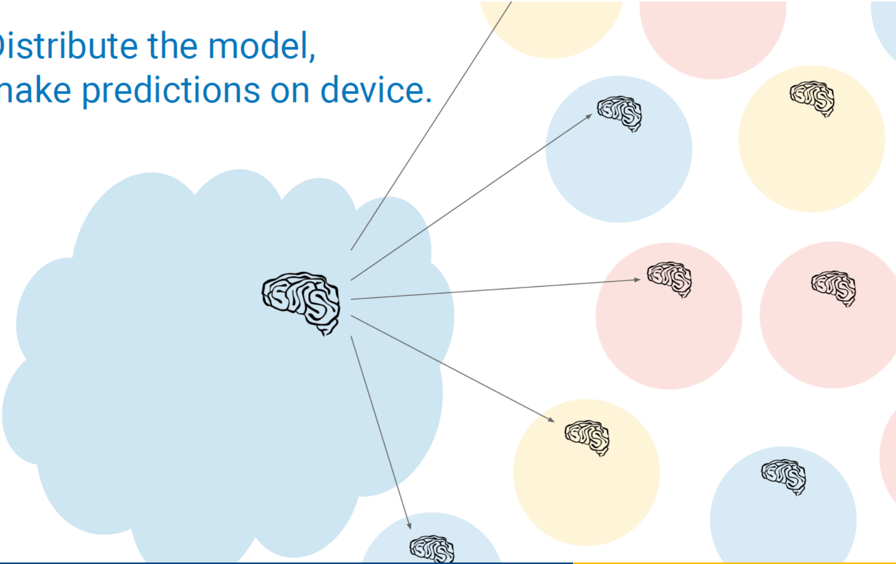
- ML models in the data center (e.g., Forecasting weather)
- ML models in the device (e.g., Keyboard suggestion)

On-device inference



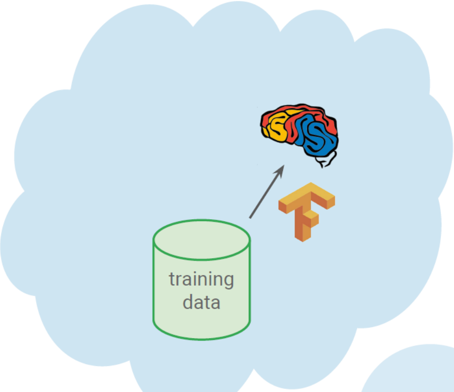
On-device inference

Distribute the model,
make predictions on device.



On-device inference

But how do we continue to improve the model?

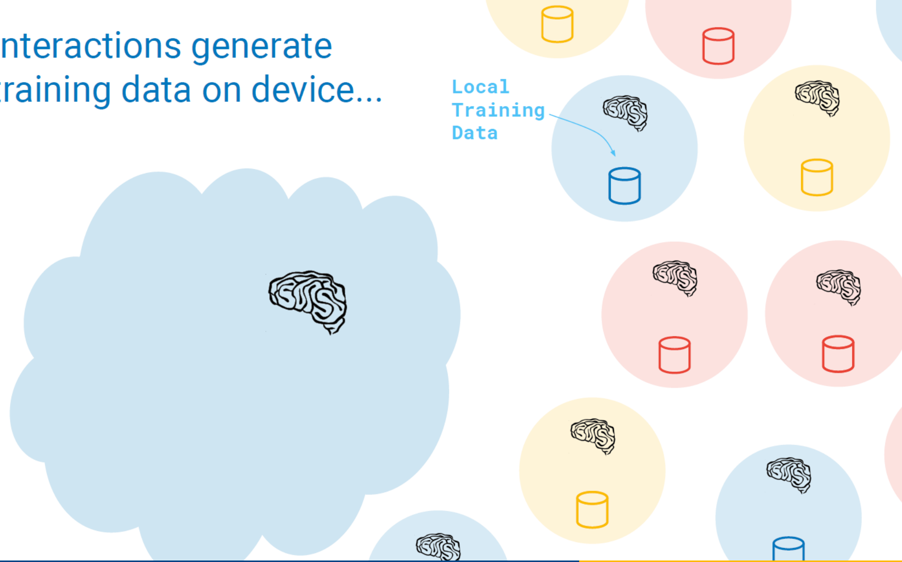


On-device inference

But how do we continue to improve the model?

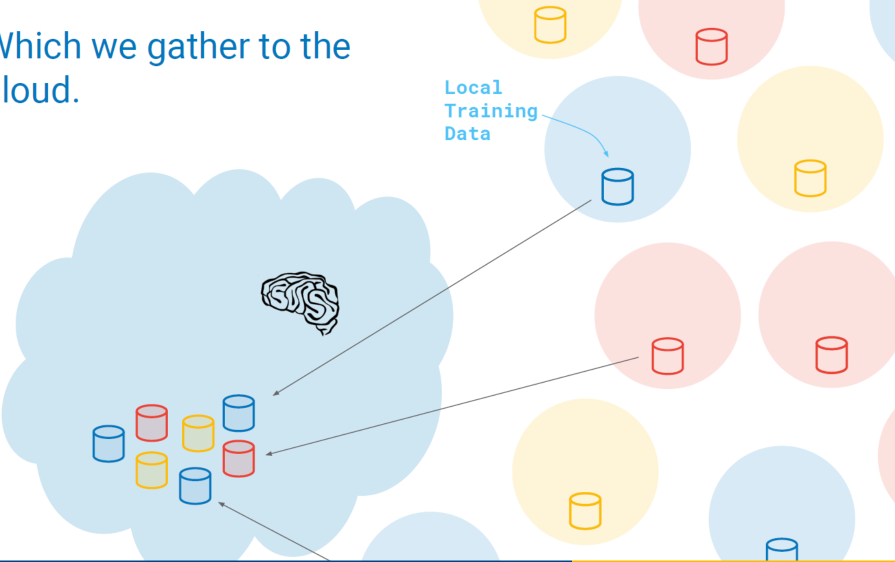
On-device inference

Interactions generate training data on device...



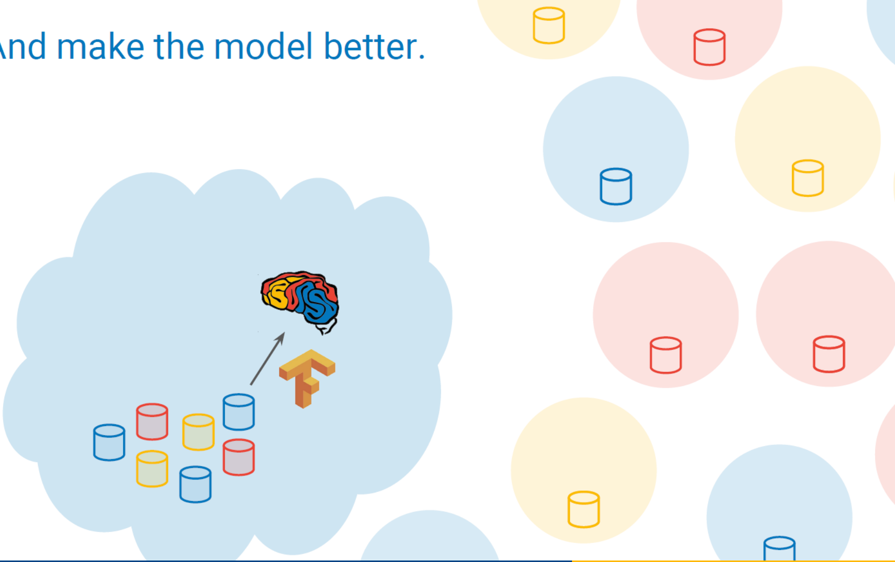
On-device inference

Which we gather to the cloud.

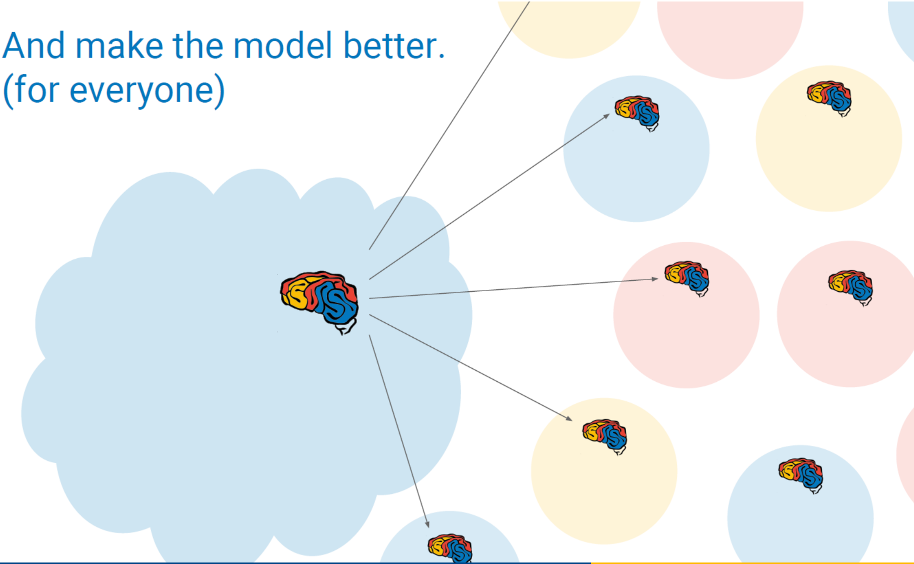


On-device inference

And make the model better.



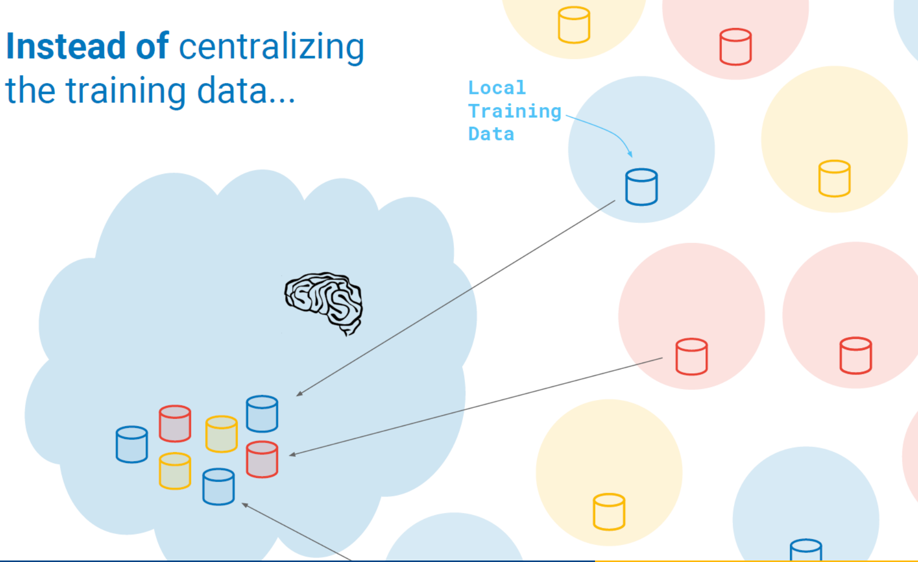
On-device inference



On-device inference

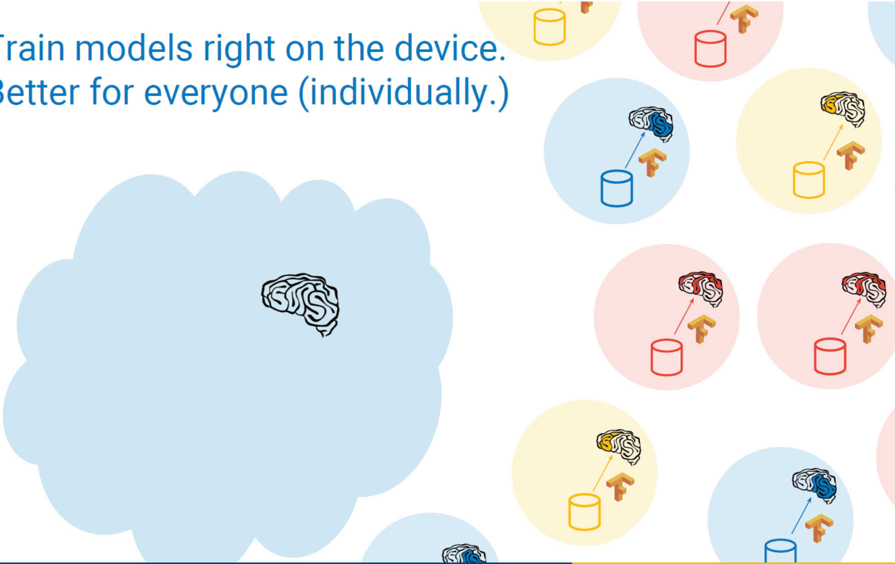
What about users privacy?

On-device inference

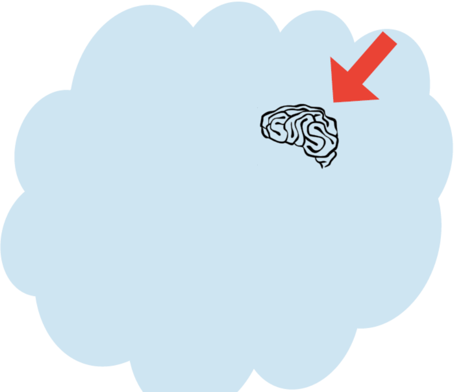


On-device inference

Train models right on the device.
Better for everyone (individually.)



On-device inference



But what about...

1. New User Experience
2. Benefitting from peers' data

Federated Computation and Learning

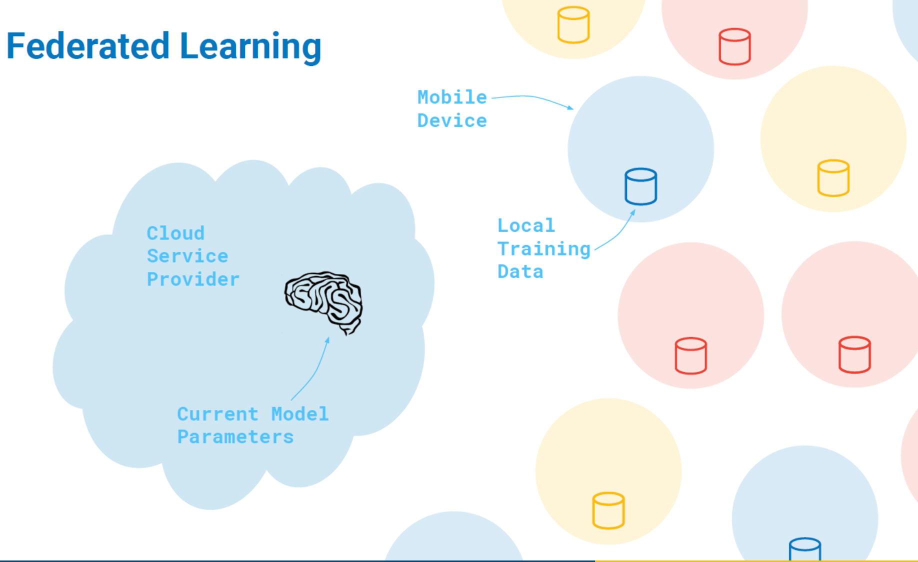
Federated learning

Where a server coordinates a fleet of participating devices to compute aggregated knowledge of devices private data.

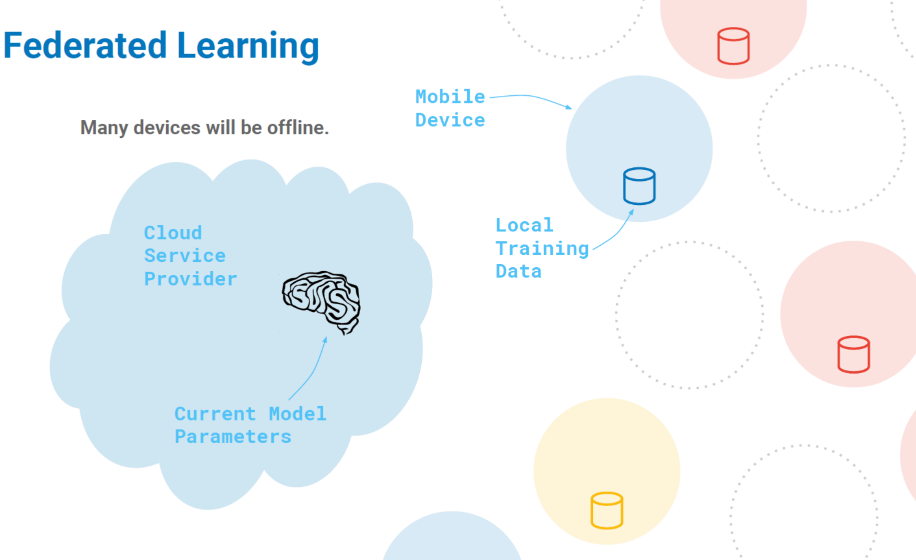
Benefits:

- Privacy
- Global Model
- On device inference (Communication Friendly)

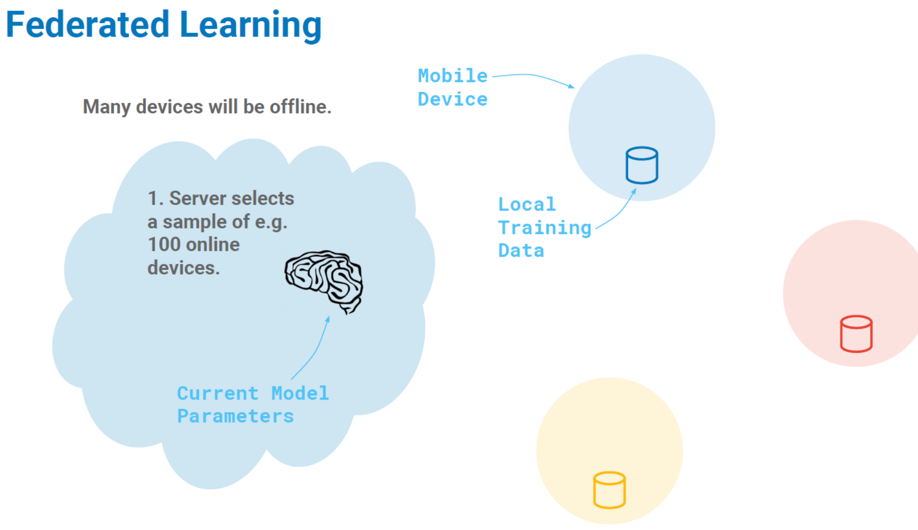
Federated Learning



Federated Learning

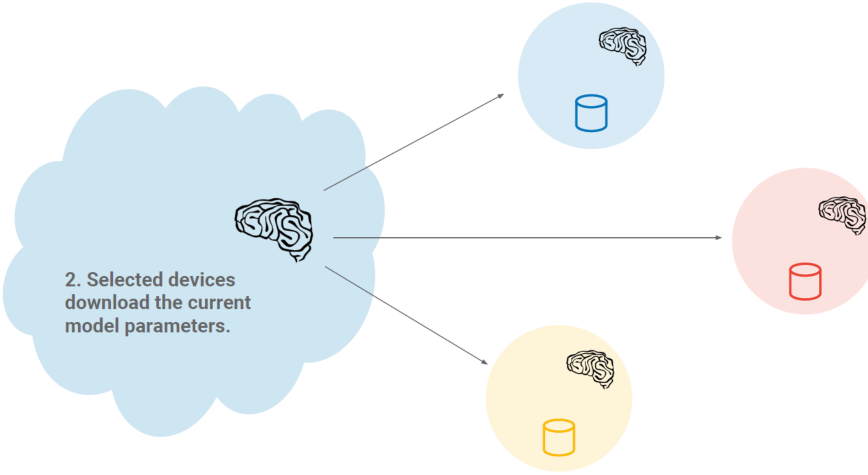


Federated Learning



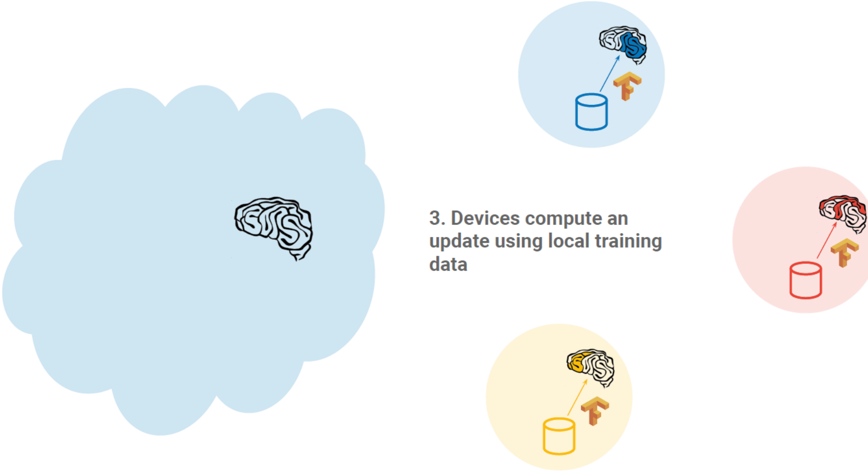
Federated Learning

Federated Learning

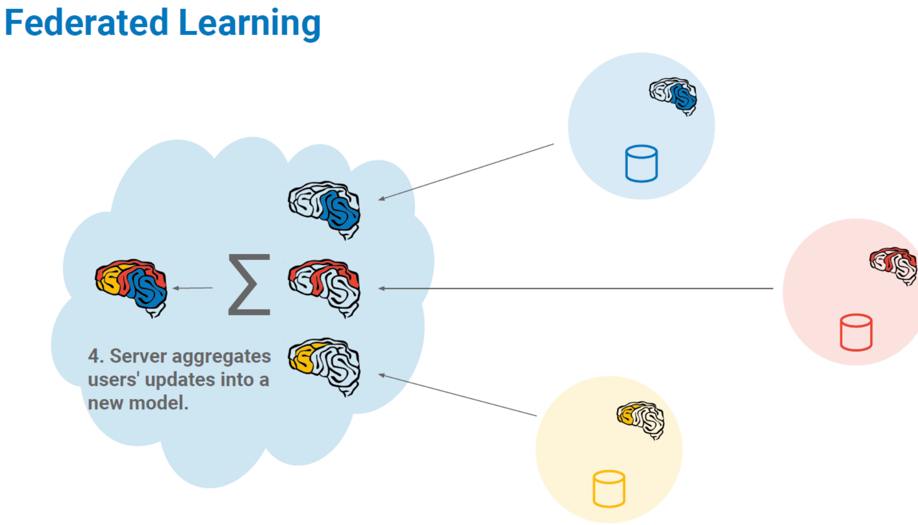


Federated Learning

Federated Learning

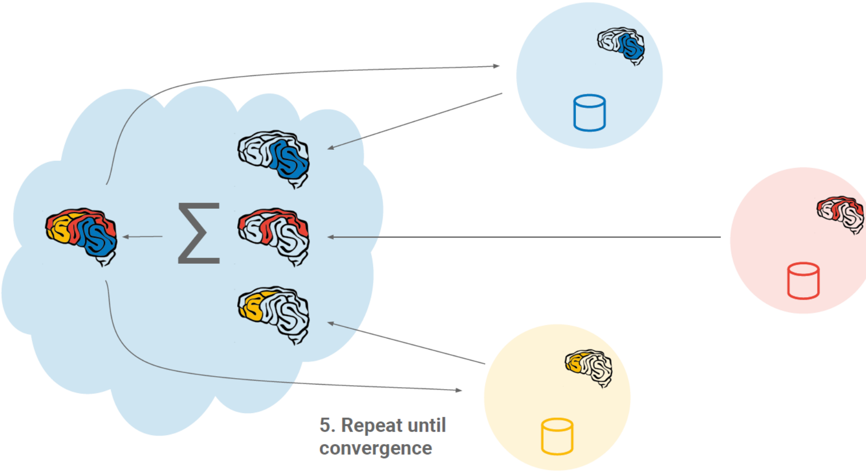


Federated Learning



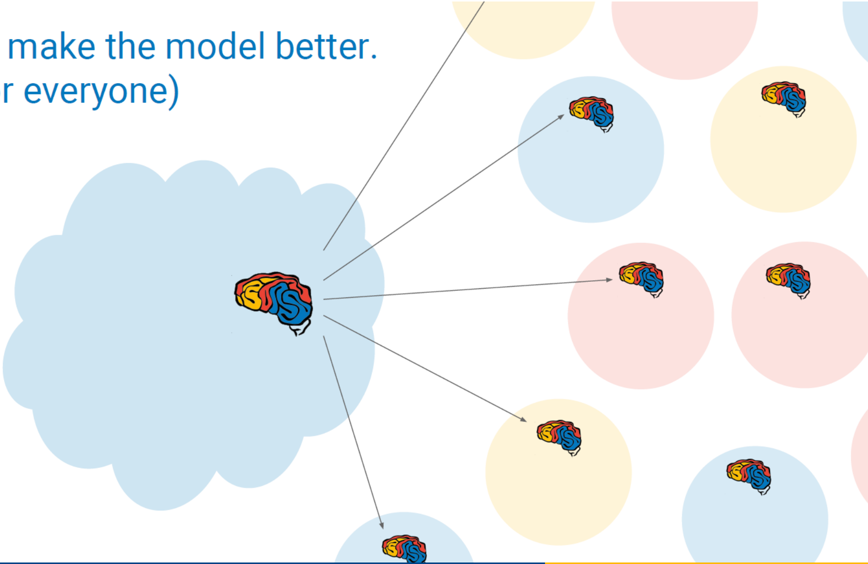
Federated Learning

Federated Learning



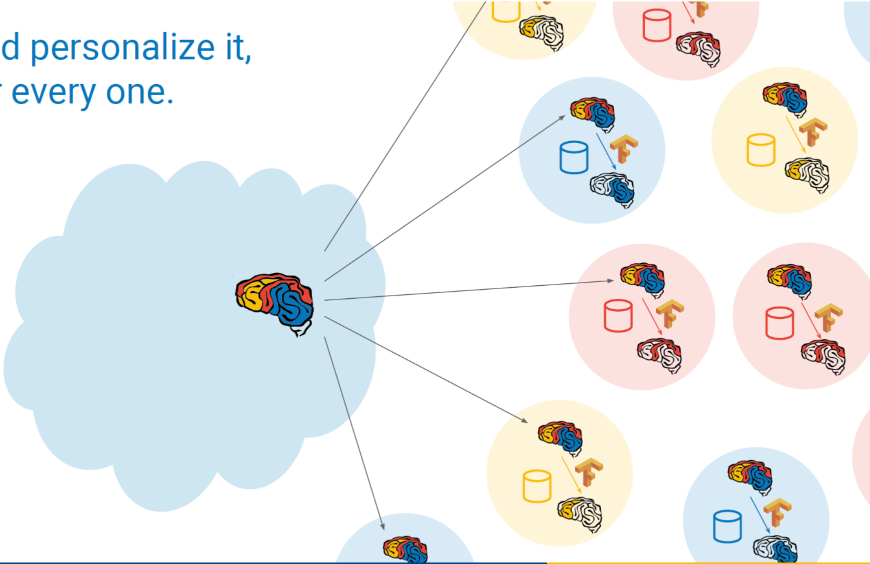
Federated Learning

To make the model better.
(for everyone)



Federated Learning

And personalize it,
for every one.



Characteristics/Challenges of Federated Learning

- **Massively Distributed**

Characteristics/Challenges of Federated Learning

- **Massively Distributed**
  - Training data is stored across a very large number of devices

Characteristics/Challenges of Federated Learning

- **Massively Distributed**
  - Training data is stored across a very large number of devices
- **Limited Communication**

Characteristics/Challenges of Federated Learning

- **Massively Distributed**
  - Training data is stored across a very large number of devices
- **Limited Communication**
  - Only a handful of rounds of unreliable communication with each devices

Characteristics/Challenges of Federated Learning

- **Massively Distributed**
  - Training data is stored across a very large number of devices
- **Limited Communication**
  - Only a handful of rounds of unreliable communication with each devices
- **Unbalanced Data**

Characteristics/Challenges of Federated Learning

- **Massively Distributed**
  - Training data is stored across a very large number of devices
- **Limited Communication**
  - Only a handful of rounds of unreliable communication with each devices
- **Unbalanced Data**
  - Some devices have few examples, some have orders of magnitude more

Characteristics/Challenges of Federated Learning

- **Massively Distributed**
  - Training data is stored across a very large number of devices
- **Limited Communication**
  - Only a handful of rounds of unreliable communication with each devices
- **Unbalanced Data**
  - Some devices have few examples, some have orders of magnitude more
- **Highly Non-IID Data**

Characteristics/Challenges of Federated Learning

- **Massively Distributed**
  - Training data is stored across a very large number of devices
- **Limited Communication**
  - Only a handful of rounds of unreliable communication with each devices
- **Unbalanced Data**
  - Some devices have few examples, some have orders of magnitude more
- **Highly Non-IID Data**
  - Data on each device reflects one individual's usage pattern

Characteristics/Challenges of Federated Learning

- **Massively Distributed**
  - Training data is stored across a very large number of devices
- **Limited Communication**
  - Only a handful of rounds of unreliable communication with each devices
- **Unbalanced Data**
  - Some devices have few examples, some have orders of magnitude more
- **Highly Non-IID Data**
  - Data on each device reflects one individual's usage pattern
- **Unreliable Compute Nodes**

Characteristics/Challenges of Federated Learning

- **Massively Distributed**
  - Training data is stored across a very large number of devices
- **Limited Communication**
  - Only a handful of rounds of unreliable communication with each devices
- **Unbalanced Data**
  - Some devices have few examples, some have orders of magnitude more
- **Highly Non-IID Data**
  - Data on each device reflects one individual's usage pattern
- **Unreliable Compute Nodes**
  - Devices go offline unexpectedly; expect faults and adversaries

Characteristics/Challenges of Federated Learning

- **Massively Distributed**
  - Training data is stored across a very large number of devices
- **Limited Communication**
  - Only a handful of rounds of unreliable communication with each devices
- **Unbalanced Data**
  - Some devices have few examples, some have orders of magnitude more
- **Highly Non-IID Data**
  - Data on each device reflects one individual's usage pattern
- **Unreliable Compute Nodes**
  - Devices go offline unexpectedly; expect faults and adversaries
- **Dynamic Data Availability**

Characteristics/Challenges of Federated Learning

- **Massively Distributed**
  - Training data is stored across a very large number of devices
- **Limited Communication**
  - Only a handful of rounds of unreliable communication with each devices
- **Unbalanced Data**
  - Some devices have few examples, some have orders of magnitude more
- **Highly Non-IID Data**
  - Data on each device reflects one individual's usage pattern
- **Unreliable Compute Nodes**
  - Devices go offline unexpectedly; expect faults and adversaries
- **Dynamic Data Availability**
  - The subset of data available is non-constant, e.g. time-of-day vs. country

Applications of Federating Learning

Federated learning will find a room to exist when:

- On-device data is more relevant than server-side proxy data
- On-device data is privacy sensitive or large

Examples of some application?

- Language modeling for mobile keyboards and voice recognition
- Medical diagnosis
- Mobile face recognition
- ...

The Federated Averaging Algorithm

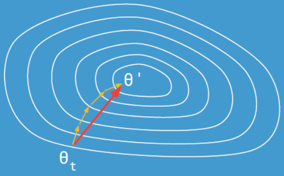
Server

Until Converged:

1. Select a random subset (e.g. 1000) of the (online) clients
2. In parallel, send current parameters θ_t to those clients

Selected Client k

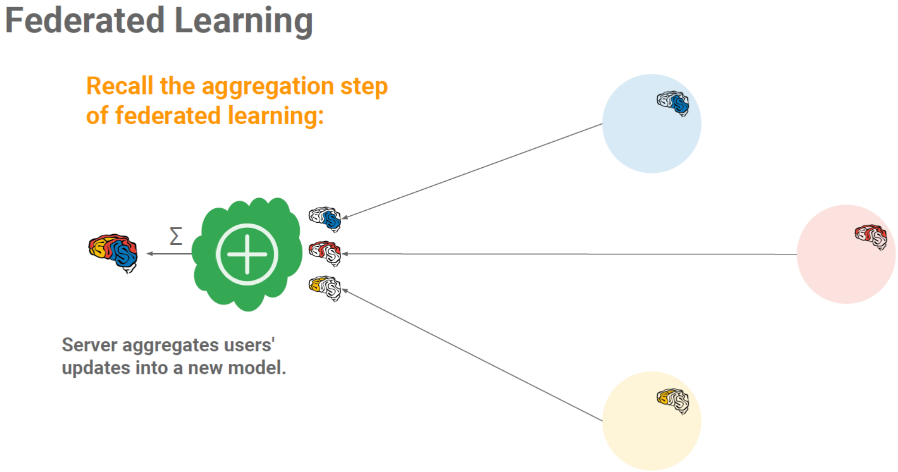
1. Receive θ_t from server.
2. Run some number of minibatch SGD steps, producing θ'
3. Return $\theta'-\theta_t$ to server.



3. $\theta_{t+1} = \theta_t$ + data-weighted average of client updates

H. B. McMahan, *et al.* **Communication-Efficient Learning of Deep Networks from Decentralized Data.** AISTATS 2017

Concerns in Federated Learning



Concerns in Federated Learning

Federated Learning

Might these updates contain privacy-sensitive data?

Concerns in Federated Learning

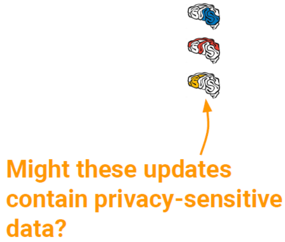
Federated Learning

1. Ephemeral

Might these updates contain privacy-sensitive data?

Concerns in Federated Learning

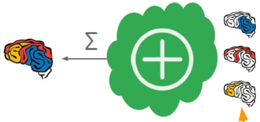
Federated Learning



1. Ephemeral
2. **Focused**

Concerns in Federated Learning

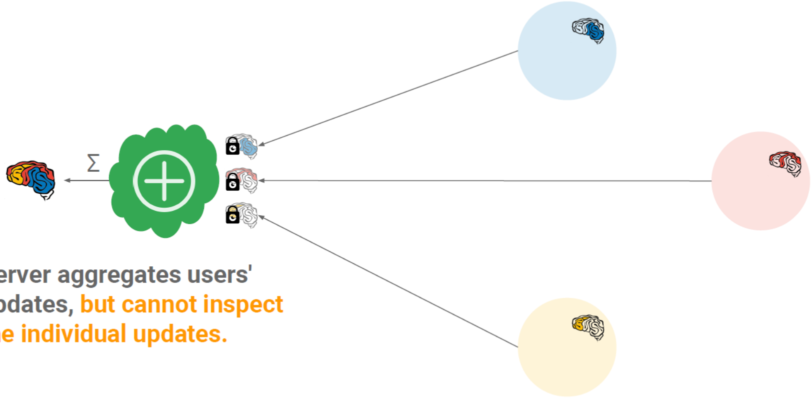
Federated Learning



1. Ephemeral
2. Focused
3. **Only in Aggregate**

Secure Aggregation

Wouldn't it be great if...



Server aggregates users' updates, but cannot inspect the individual updates.

Secure Aggregation

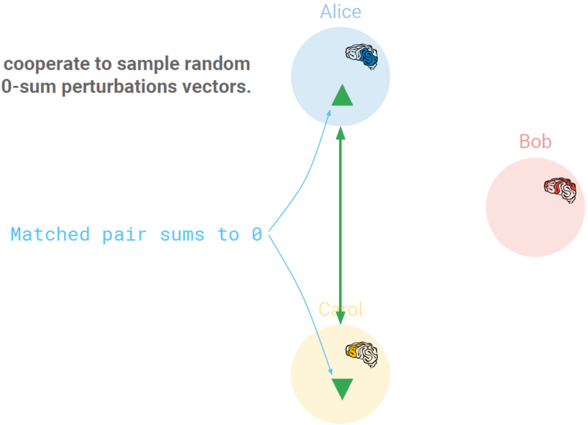
Secure Aggregation protocols aims to protect the privacy of the updates sent by the clients to the aggregator by letting the aggregator able only to calculate the aggregate update but not able to access the individual updates[2]

[2] https://storage.googleapis.com/pub-tools-public-publication-data/pdf/ae87385258d90b9e48377ed49d83d467b45d5776.pdf

Secure Aggregation

Random positive/negative pairs, *aka* antiparticles

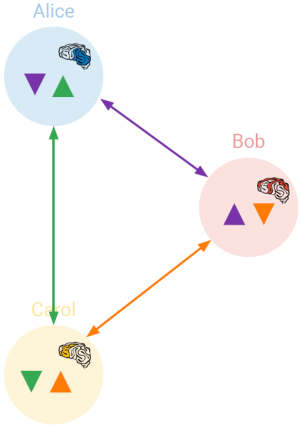
Devices cooperate to sample random pairs of 0-sum perturbations vectors.



Secure Aggregation

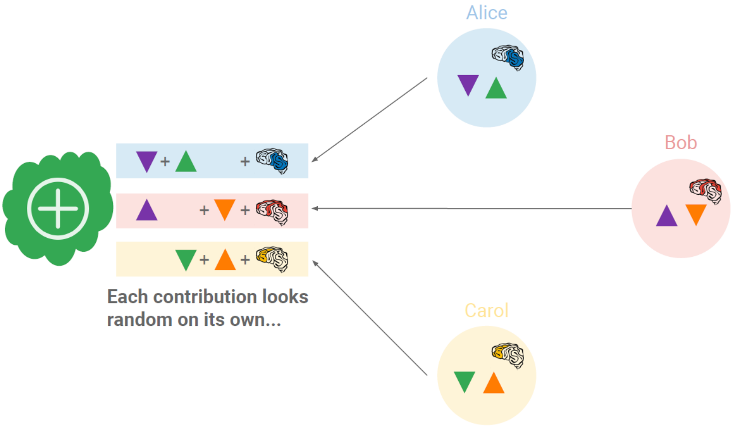
Random positive/negative pairs, *aka* antiparticles

Devices cooperate to sample random pairs of 0-sum perturbations vectors.



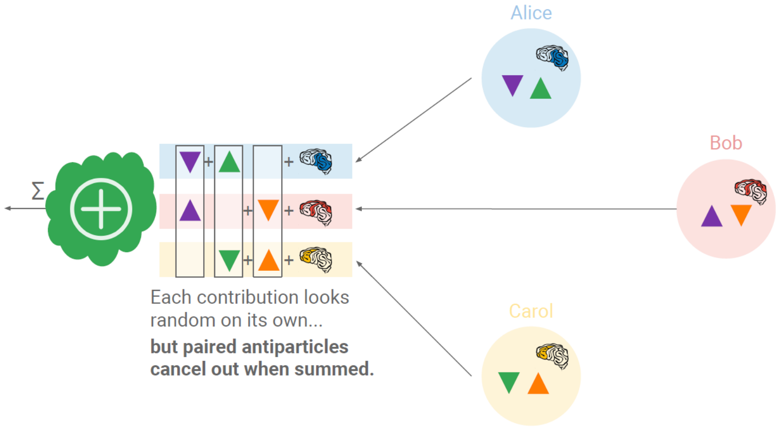
Secure Aggregation

Add antiparticles before sending to the server

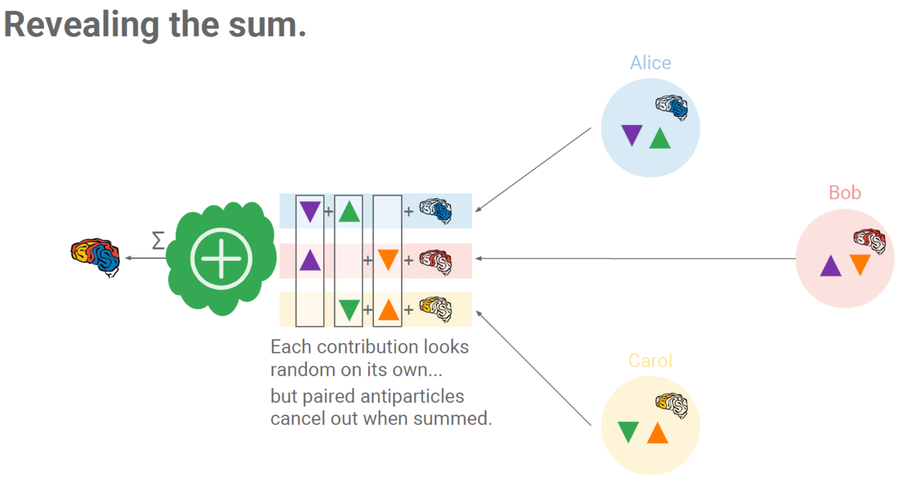


Secure Aggregation

The antiparticles cancel when summing contributions

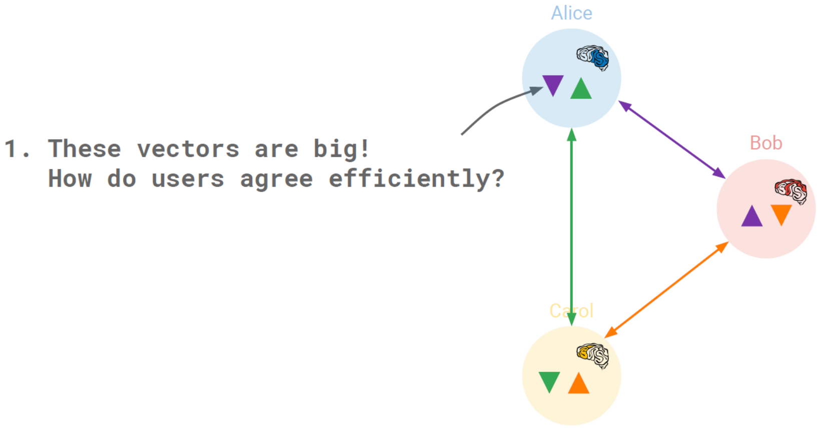


Secure Aggregation



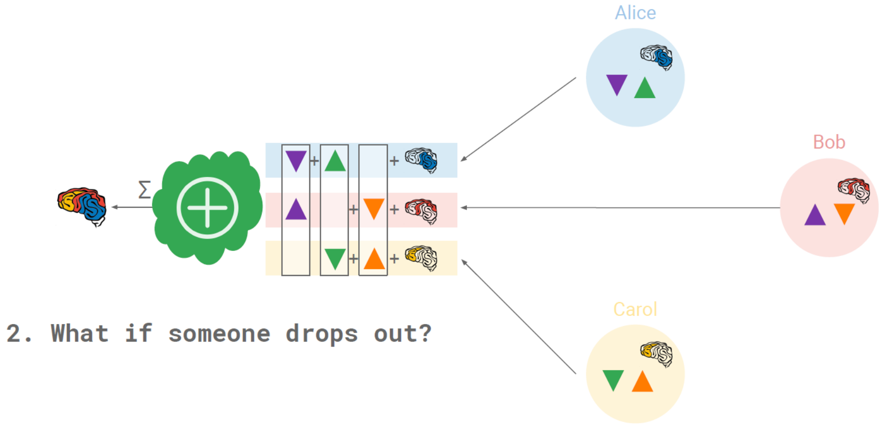
Problems in this approach

There are two main problems.



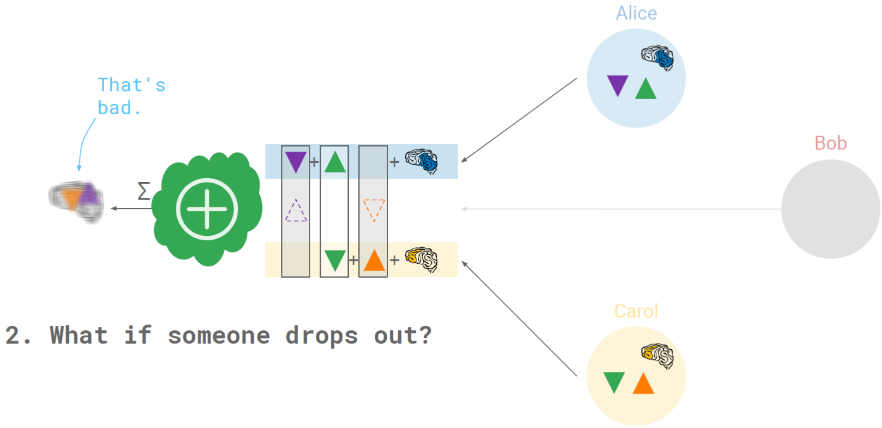
Problems in this approach

There are two main problems.



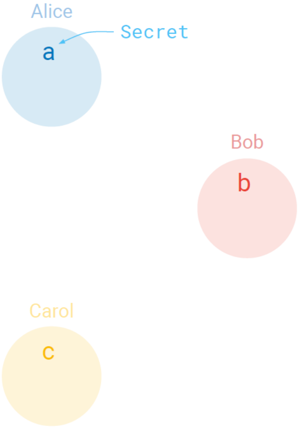
Problems in this approach

There are two main problems.



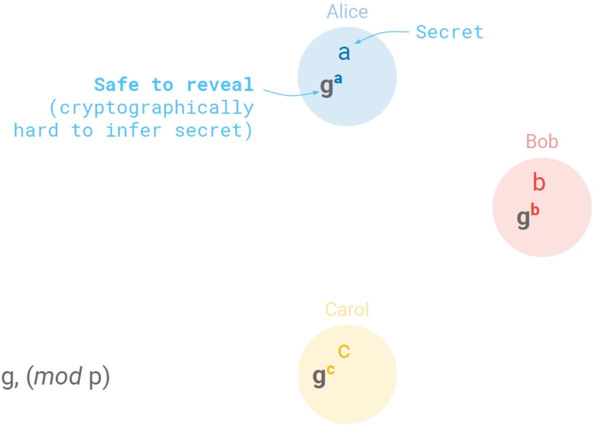
Secure Aggregation Protocol (Addressing First Problem)

Pairwise Diffie-Hellman Key Agreement



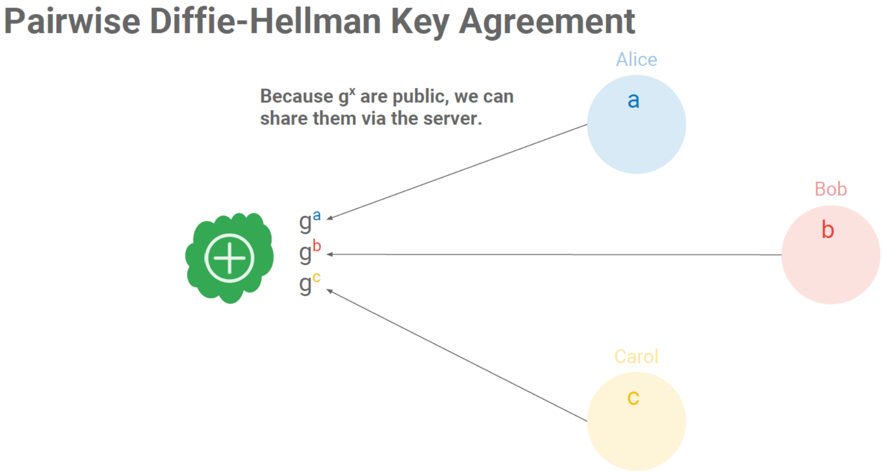
Secure Aggregation Protocol (Addressing First Problem)

Pairwise Diffie-Hellman Key Agreement

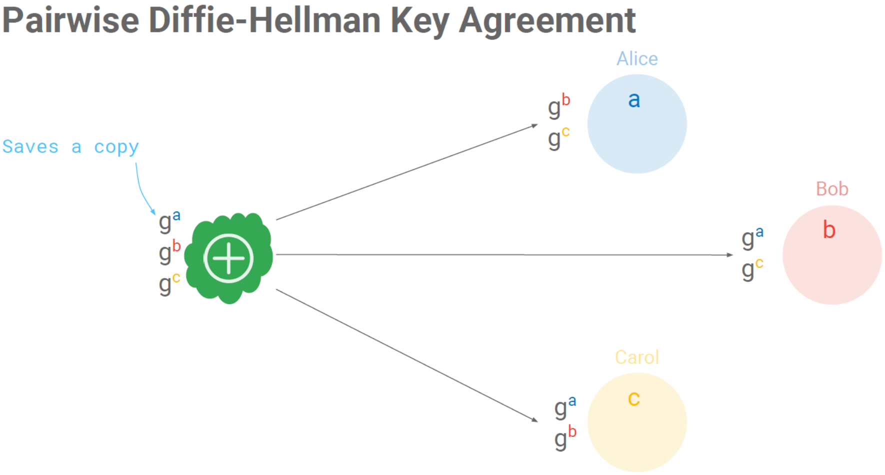


Public parameters: g, (*mod* p)

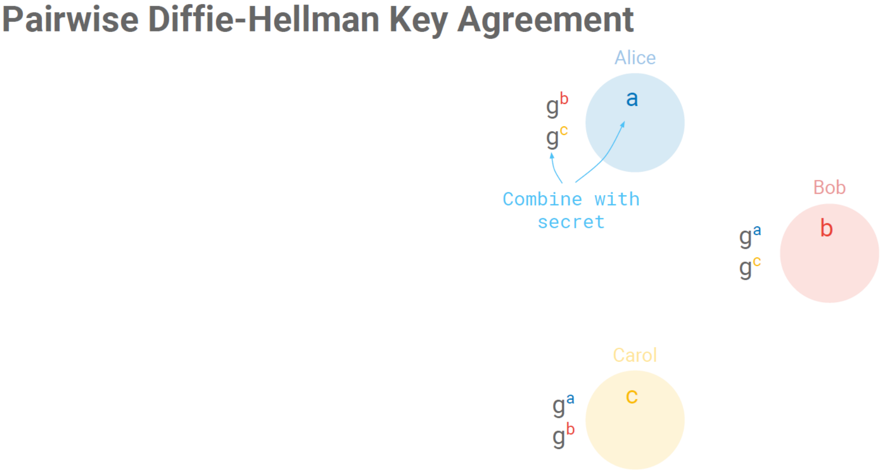
Secure Aggregation Protocol (Addressing First Problem)



Secure Aggregation Protocol (Addressing First Problem)

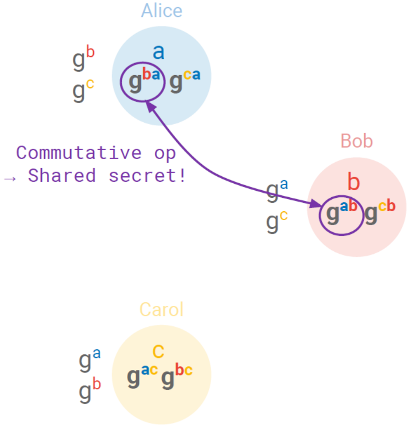


Secure Aggregation Protocol (Addressing First Problem)



Secure Aggregation Protocol (Addressing First Problem)

Pairwise Diffie-Hellman Key Agreement



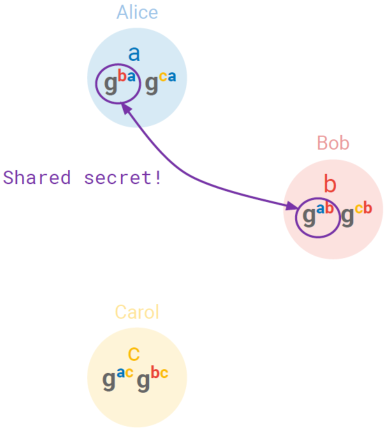
Secure Aggregation Protocol (Addressing First Problem)

Pairwise Diffie-Hellman Key Agreement + PRNG Expansion

Secrets are scalars, but....

Use each secret to seed a **pseudorandom number generator**, generate paired antiparticle vectors.

$\mathrm{PRNG}(g^{ba}) \rightarrow \vec{\blacktriangledown} = -\vec{\blacktriangle}$



Secure Aggregation Protocol (Addressing Second Problem)

How to enable online users to recover the secrets of any user that may go offline?

Secure Aggregation Protocol (Addressing Second Problem)

How to enable online users to recover the secrets of any user that may go offline?

Using k-out-of-n Threshold Secret Sharing

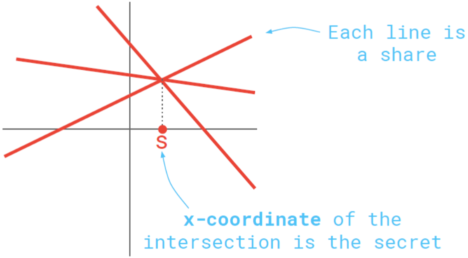
k-out-of-n Threshold Secret Sharing

k-out-of-*n* Threshold Secret Sharing

Goal: Break a secret into *n* pieces, called shares.

- **<k shares:** learn nothing
- **≥k shares**: recover *s* perfectly.

2-out-of-3 secret sharing:

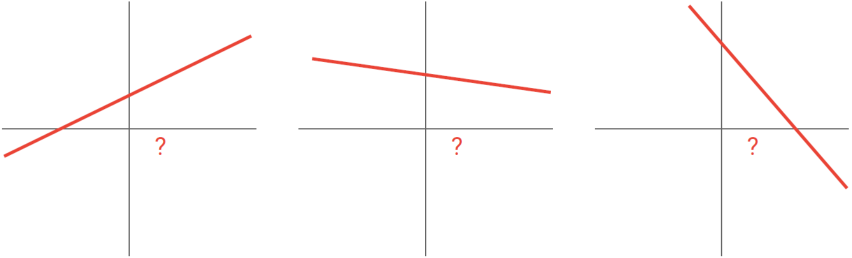


k-out-of-n Threshold Secret Sharing

k-out-of-*n* Threshold Secret Sharing

Goal: Break a secret into *n* pieces, called shares.

- **<k shares:** learn nothing
- **≥k shares**: recover *s* perfectly

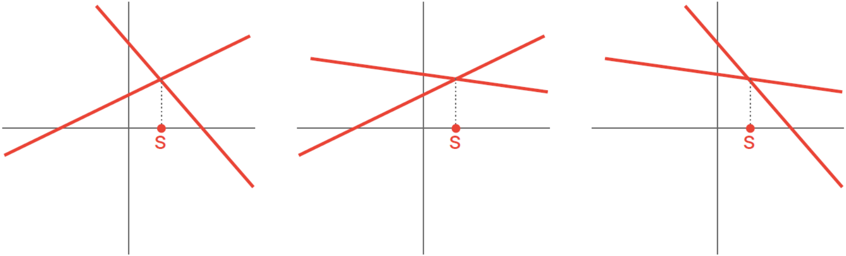


k-out-of-n Threshold Secret Sharing

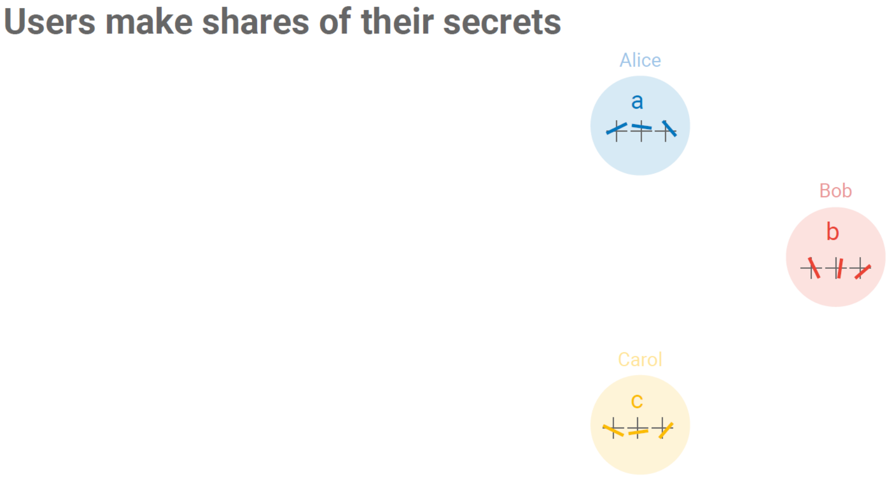
k-out-of-*n* Threshold Secret Sharing

Goal: Break a secret into *n* pieces, called shares.

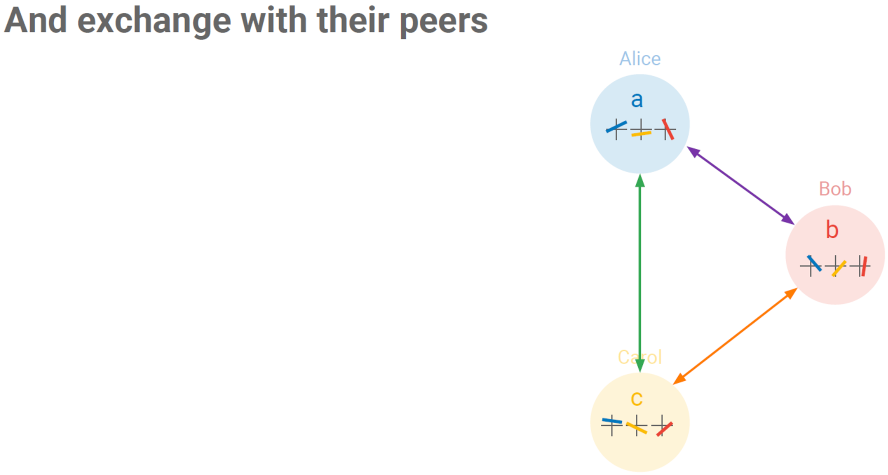
- **<k shares:** learn nothing
- **≥k shares**: recover *s* perfectly



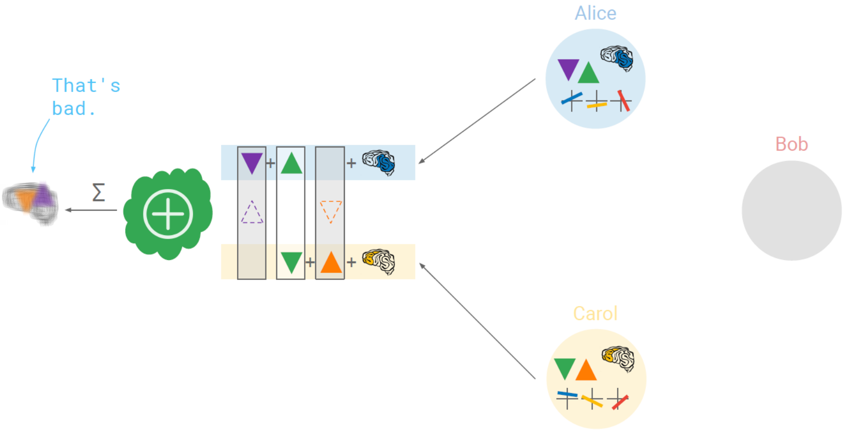
Secure Aggregation Protocol (Addressing Second Problem)



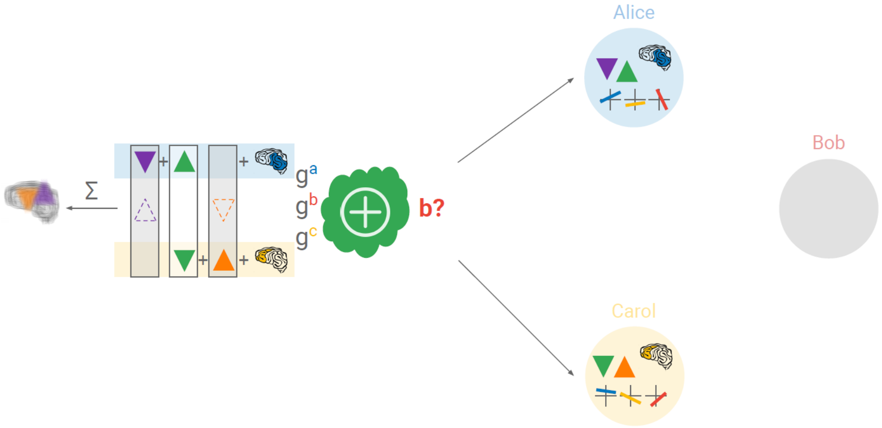
Secure Aggregation Protocol (Addressing Second Problem)



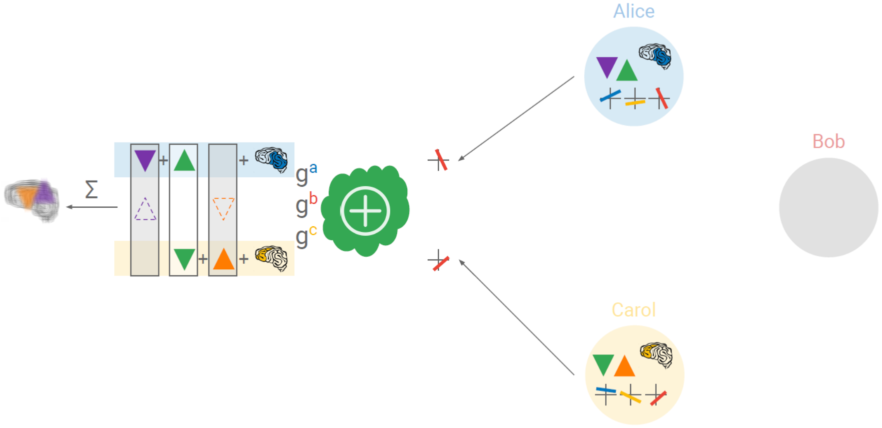
Secure Aggregation Protocol (Addressing Second Problem)



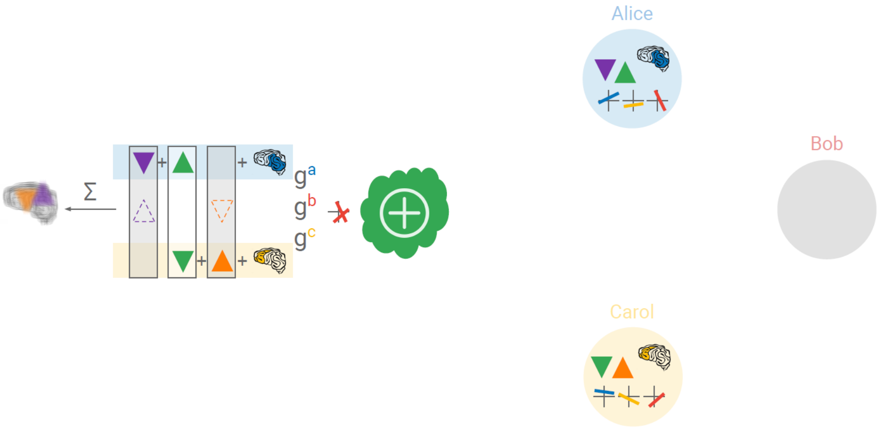
Secure Aggregation Protocol (Addressing Second Problem)



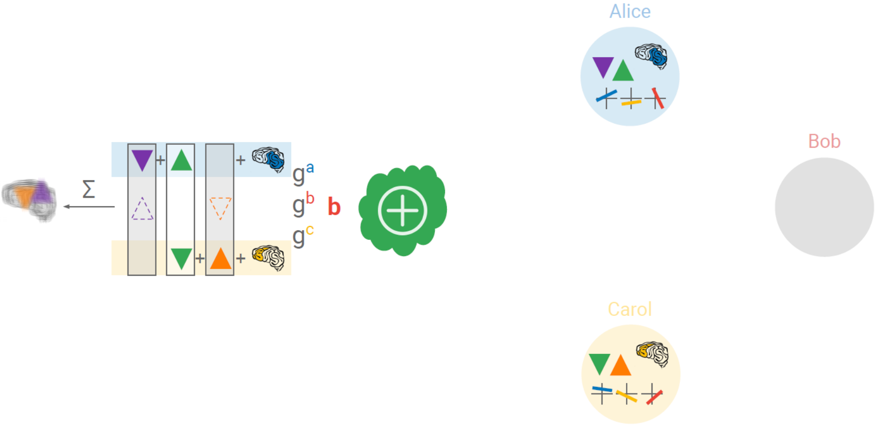
Secure Aggregation Protocol (Addressing Second Problem)



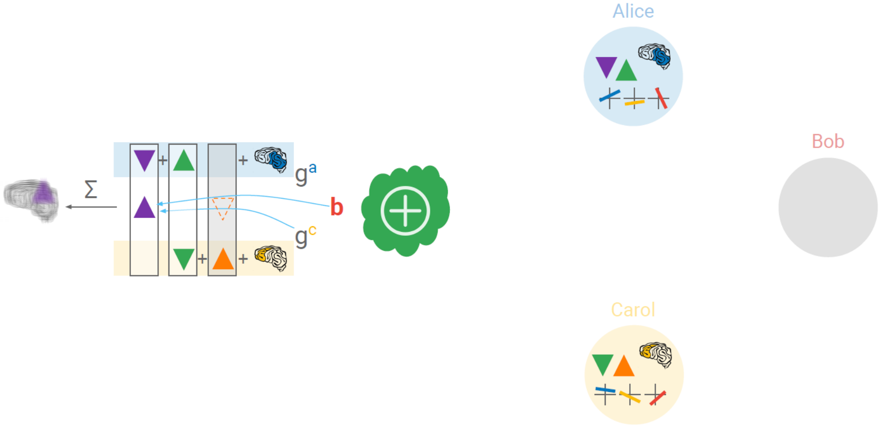
Secure Aggregation Protocol (Addressing Second Problem)



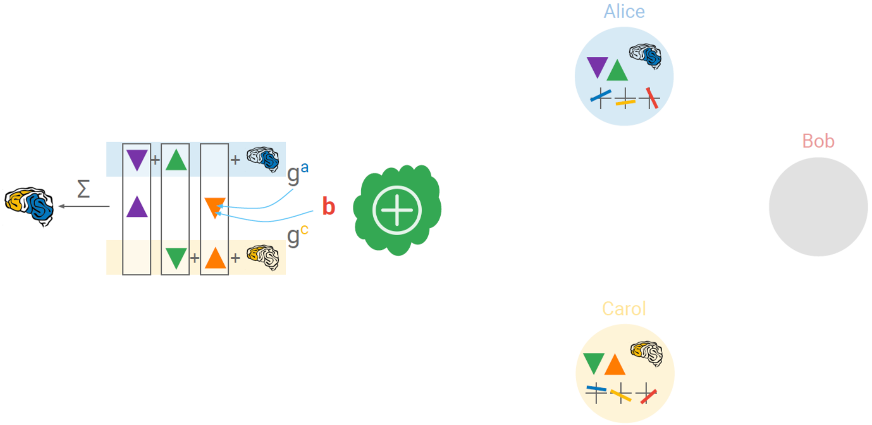
Secure Aggregation Protocol (Addressing Second Problem)



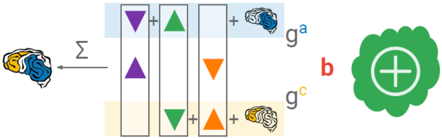
Secure Aggregation Protocol (Addressing Second Problem)



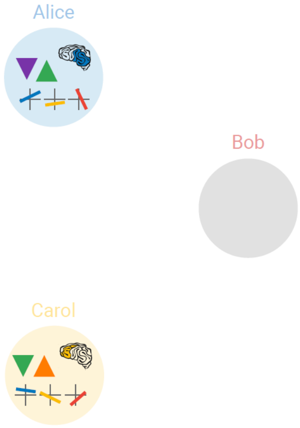
Secure Aggregation Protocol (Addressing Second Problem)



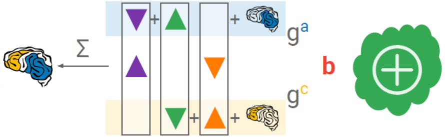
Secure Aggregation Protocol (Addressing Second Problem)

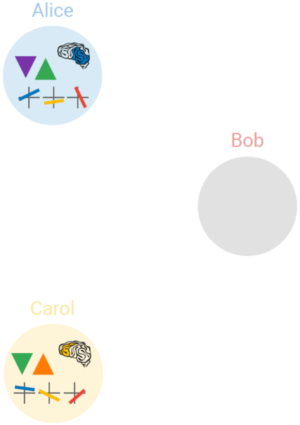


Enough honest users + a high enough threshold
⇒ dishonest users cannot reconstruct the secret.



Secure Aggregation Protocol (Addressing Second Problem)

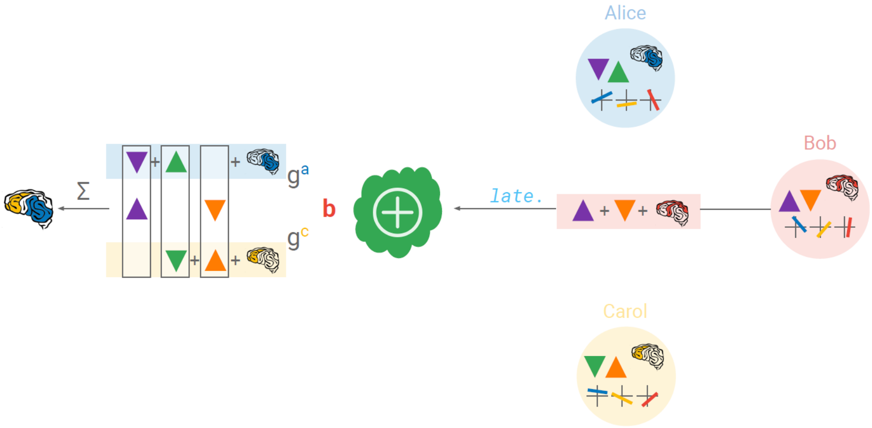




Enough honest users + a high enough threshold
⇒ dishonest users cannot reconstruct the secret.

However....

Secure Aggregation Protocol (Addressing Second Problem)

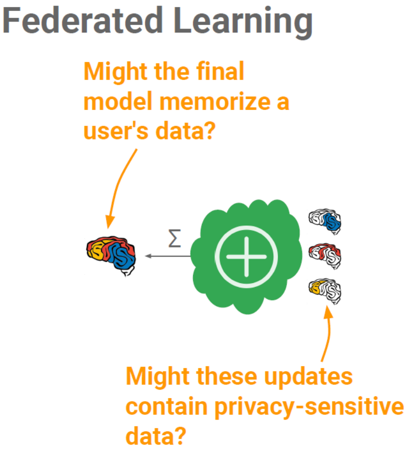


Secure Aggregation Protocol (Addressing Second Problem)

Summary for Secure Aggregation

- Diffie Hellman: Used for efficient keys (secrets) sharing among participants
- k-out-of-n Threshold secret sharing: Used to make the algorithm resilience to the drop out of any participant

Differential Privacy



1. Ephemeral
2. Focused
3. Only in Aggregate
4. **Differential Privacy**

Differential Privacy

Differential privacy is the statistical science of trying to learn as much as possible about a group while learning as little as possible about any individual in it.

Differential Privacy

Differential privacy is the statistical science of trying to learn as much as possible about a group while learning as little as possible about any individual in it.

Differential privacy is achieved by simply adding a gaussian noise to the data or the output of the function we are protecting.

- Local Differential Privacy

Differential Privacy

Differential privacy is the statistical science of trying to learn as much as possible about a group while learning as little as possible about any individual in it.

Differential privacy is achieved by simply adding a gaussian noise to the data or the output of the function we are protecting.

- Local Differential Privacy
  - $f(x_1, \ldots, x_n) = \sum_i x_i$

Differential Privacy

Differential privacy is the statistical science of trying to learn as much as possible about a group while learning as little as possible about any individual in it.

Differential privacy is achieved by simply adding a gaussian noise to the data or the output of the function we are protecting.

- Local Differential Privacy
  - $f(x_1, \ldots, x_n) = \sum_i x_i$
  - $f(x_1 + \mathcal{N}_1, \ldots, x_n + \mathcal{N}_n) = \sum_i (x_i + \mathcal{N}_i)$

Differential Privacy

Differential privacy is the statistical science of trying to learn as much as possible about a group while learning as little as possible about any individual in it.

Differential privacy is achieved by simply adding a gaussian noise to the data or the output of the function we are protecting.

- Local Differential Privacy
  - $f(x_1, \ldots, x_n) = \sum_i x_i$
  - $f(x_1 + \mathcal{N}_1, \ldots, x_n + \mathcal{N}_n) = \sum_i (x_i + \mathcal{N}_i)$
- Global Differential Privacy

Differential Privacy

Differential privacy is the statistical science of trying to learn as much as possible about a group while learning as little as possible about any individual in it.

Differential privacy is achieved by simply adding a gaussian noise to the data or the output of the function we are protecting.

- Local Differential Privacy
  - $f(x_1, \ldots, x_n) = \sum_i x_i$
  - $f(x_1 + \mathcal{N}_1, \ldots, x_n + \mathcal{N}_n) = \sum_i (x_i + \mathcal{N}_i)$
- Global Differential Privacy
  - $f(x_1, \ldots, x_n) = \sum_i x_i$

Differential Privacy

Differential privacy is the statistical science of trying to learn as much as possible about a group while learning as little as possible about any individual in it.

Differential privacy is achieved by simply adding a gaussian noise to the data or the output of the function we are protecting.

- Local Differential Privacy
  - $f(x_1, \ldots, x_n) = \sum_i x_i$
  - $f(x_1 + \mathcal{N}_1, \ldots, x_n + \mathcal{N}_n) = \sum_i (x_i + \mathcal{N}_i)$
- Global Differential Privacy
  - $f(x_1, \ldots, x_n) = \sum_i x_i$
  - $f(x_1, \ldots, x_n) = \sum_i x_i + \mathcal{N}$

Differential Privacy

Differential privacy is the statistical science of trying to learn as much as possible about a group while learning as little as possible about any individual in it.

Differential privacy is achieved by simply adding a gaussian noise to the data or the output of the function we are protecting.

- Local Differential Privacy
  - $f(x_1, \ldots, x_n) = \sum_i x_i$
  - $f(x_1 + \mathcal{N}_1, \ldots, x_n + \mathcal{N}_n) = \sum_i (x_i + \mathcal{N}_i)$
- Global Differential Privacy
  - $f(x_1, \ldots, x_n) = \sum_i x_i$
  - $f(x_1, \ldots, x_n) = \sum_i x_i + \mathcal{N}$

Differential Privacy

Differential privacy is the statistical science of trying to learn as much as possible about a group while learning as little as possible about any individual in it.

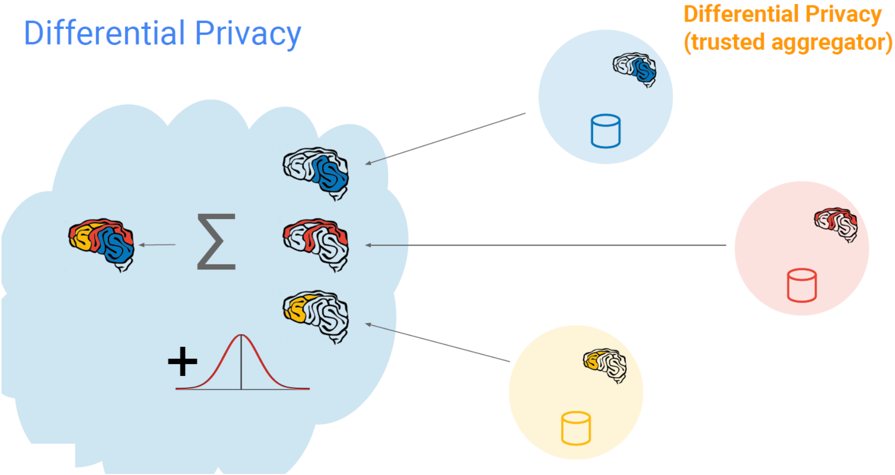
Differential privacy is achieved by simply adding a gaussian noise to the data or the output of the function we are protecting.

- Local Differential Privacy
  - $f(x_1, \ldots, x_n) = \sum_i x_i$
  - $f(x_1 + \mathcal{N}_1, \ldots, x_n + \mathcal{N}_n) = \sum_i (x_i + \mathcal{N}_i)$
- Global Differential Privacy
  - $f(x_1, \ldots, x_n) = \sum_i x_i$
  - $f(x_1, \ldots, x_n) = \sum_i x_i + \mathcal{N}$

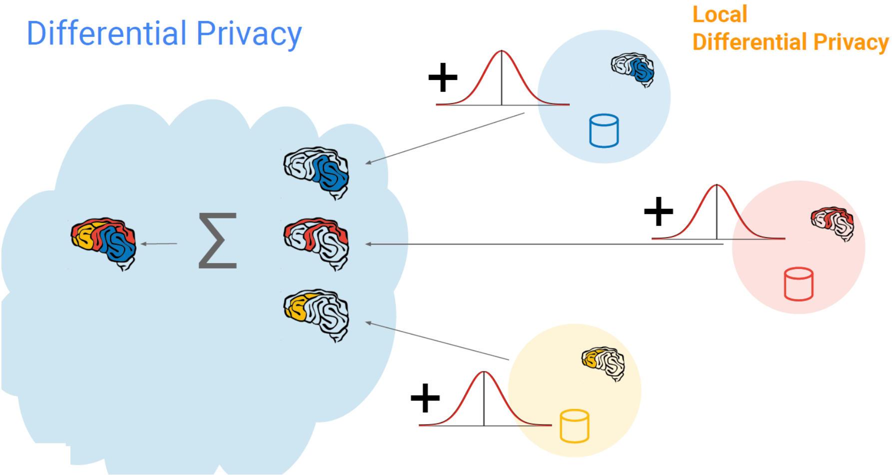
Note

Adding noise should be done with caution. We consider function Sensitivity.

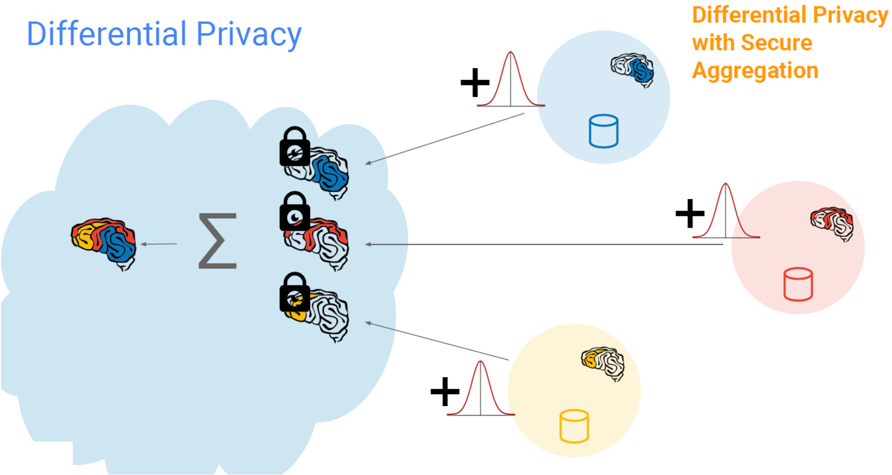
Differential Privacy



Differential Privacy



Differential Privacy



Differentially-Private Federated Averaging[3]

Server

Until Converged:

1. Select a random subset (e.g. C=100) of the (online) clients

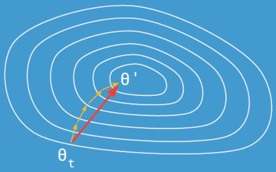
2. In parallel, send current parameters $\boldsymbol{\theta}_t$ to those clients

Selected Client k

1. Receive θ_t from server.

2. Run some number of minibatch SGD steps, producing θ'

3. Return $\theta' - \theta_t$ to server.



3. $\theta_{t+1} = \theta_t$ + data-weighted average of client updates

[3] McMahan, Ramage, Talwar, Zhang. Learning Differentially Private Recurrent Language Models.

Differentially-Private Federated Averaging[3]

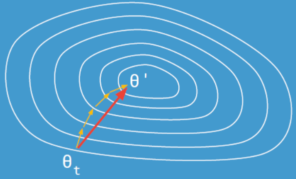
Server

Until Converged:

1. Select each user **independently** with **probability q**, for say E[C]=1000 clients
2. In parallel, send current parameters $\boldsymbol{\theta_t}$ to those clients

Selected Client k

1. Receive θ_t from server.
2. Run some number of minibatch SGD steps, producing θ'
3. Return **Clip(θ'-$\boldsymbol{\theta_t}$)** to server.



3. $\theta_{t+1} = \theta_t$ + **bounded sensitivity** data-weighted average of client updates + Gaussian noise **N(0, I$\boldsymbol{\sigma^2}$)**

[3] McMahan, Ramage, Talwar, Zhang. Learning Differentially Private Recurrent Language Models.

References

- Jakub Konecny, Federated Learning Privacy-Preserving Collaborative Machine Learning without Centralized Training Data
- H. B. McMahan, et al. Communication-Efficient Learning of Deep Networks from Decentralized Data. AISTATS 2017
- K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, K. Seth. Practical Secure Aggregation for Privacy-Preserving Machine Learning, CCS 2017.

Questions