Context
ML on Sensitive Data: Privacy versus Utility
ML on Sensitive Data: Privacy versus Utility (?)

our goal

1. Policy
2. New Technology

Push the pareto frontier with better technology.
Why federated learning?
Data is born at the edge

Billions of phones & IoT devices constantly generate data

Data enables better products and smarter models
Can data live at the edge?

Data processing is moving on device:
- Improved latency
- Works offline
- Better battery life
- Privacy advantages

E.g., on-device inference for mobile keyboards and cameras.
Can data live at the edge?

Data processing is moving on device:
● Improved latency
● Works offline
● Better battery life
● Privacy advantages

E.g., on-device inference for mobile keyboards and cameras.

What about analytics? What about learning?
2014: Three choices

- Don’t use data to improve products and services
- Log the data centrally *anyway*
- Invent a new solution

Choose your poison ☐
2014: Three choices

- Don’t use data to improve products and services
- Log the data centrally *anyway*
- Invent a new solution

Choose your poison 🕊
2019: Good reason to hope

- Don’t use data to improve products and services
- Log the data centrally *anyway*
- Federated learning and analytics
Federated Learning
Federated Learning

Initial (Untrained) Model.

Mobile Device

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

Initial (Untrained) Model.
2. Selected devices download the current model parameters.
Federated Learning

3. Users compute an update using local training data.
Federated Learning

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

Repeat until convergence.
Federated Learning

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

\[ \sum \]

Repeat until convergence.

- Data flowing back to server are model parameters, NOT raw input data.

- Composable with strong privacy guarantees, which we’ll describe in a bit.
The Final Model is Deployed For Inference

Deploy the best model to all devices (millions).
The Federated Averaging algorithm

Server

Until Converged:
1. Select a random subset of clients
2. In parallel, send current parameters $\theta_t$ to those clients

Selected Client $k$

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

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

Gboard: language modeling

- Predict the next word based on typed text so far
- Powers the predictions strip

When should you consider federated learning?
- On-device data is more relevant than server-side proxy data
- On-device data is privacy sensitive or large
- Labels can be inferred naturally from user interaction
Gboard: language modeling

Federated model compared to baseline

Other federated models in Gboard

**Emoji prediction**
- 7% more accurate emoji predictions
- prediction strip clicks +4% more
- 11% more users share emojis!

**Action prediction**
When is it useful to suggest a gif, sticker, or search query?
- 47% reduction in unhelpful suggestions
- increasing overall emoji, gif, and sticker shares

**Discovering new words**
Federated discovery of what words people are typing that Gboard doesn’t know.


Is federated computation just distributed computing?
Semi-cyclic data availability

Each device reflects one user's data.

So no one device is representative of the whole population.

Devices must idle, plugged-in, on wi-fi to participate.

Device availability correlates with both geo location and data distribution.

- Rounds complete faster when more devices available
- Device availability changes over the course of a day
FL andPrivacy
4. Server aggregates users' updates into a new model.

$\sum$

Repeat until convergence.
Might these updates contain privacy-sensitive data?

1. Ephemeral
2. Focused

Improve privacy & security by minimizing the "attack surface"
Might these updates contain privacy-sensitive data?

1. Ephemeral
2. Focused
3. Only in aggregate
Secure Aggregation

A practical protocol with
- Security guarantees
- Communication efficiency
- Dropout tolerance

Each contribution looks random on its own... but paired "antiparticles" cancel out when summed.

Might the final model memorize a user's data? (e.g., B. McMahan's credit card #)

1. Ephemeral
2. Focused
3. Only in aggregate
4. Differentially private
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.

Andy Greenberg
Wired 2016.06.13
**Differential Privacy**

$(\varepsilon, \delta)$-**Differential Privacy**: The distribution of the output $M(D)$ (a trained model) on database (training dataset) $D$ is nearly the same as $M(D')$ for all adjacent databases $D$ and $D'$. 
**(ε, δ)-Differential Privacy**: The distribution of the output $M(D)$ (a trained model) on database (training dataset) $D$ is **nearly the same** as $M(D')$ for all adjacent databases $D$ and $D'$

$$\forall S: \ Pr[M(D) \in S] \leq \exp(\epsilon) \cdot Pr[M(D') \in S] + \delta$$
(ε, δ)-Differential Privacy: The distribution of the output $M(D)$ (a trained model) on database (training dataset) $D$ is nearly the same as $M(D')$ for all adjacent databases $D$ and $D'$. 

**adjacent**: Sets $D$ and $D'$ differ only by presence/absence of one example

(ε, δ)-Differential Privacy: The distribution of the output $M(D)$ (a trained model) on database (training dataset) $D$ is nearly the same as $M(D')$ for all adjacent databases $D$ and $D'$.
**Differential Privacy**

(D, δ)-Differential Privacy: The distribution of the output $M(D)$ (a trained model) on database (training dataset) $D$ is nearly the same as $M(D')$ for all adjacent databases $D$ and $D'$

**Sensitivity:** How much $\text{Query}(D)$ and $\text{Query}(D')$ differ
Federated Learning w/ Differential Privacy
Federated Learning w/ Differential Privacy

Clip updates to limit a user’s contribution (bounds sensitivity).

Federated Learning w/ Differential Privacy

Add Gaussian noise proportional to sensitivity

Clip updates to limit a user's contribution (bounds sensitivity).

Differentially-Private Federated Averaging

Server

 Until Converged:

1. Select each user independently with probability $q$, for say $E[C]=1000$ clients

2. In parallel, send current parameters $\theta_t$ to those clients

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

---

Selected Client $k$

1. Receive $\theta_t$ from server.

2. Run some number of minibatch SGD steps, producing $\theta'$

3. Return $\text{Clip}(\theta' - \theta_t)$ to server.
Challenges to private, decentralized learning/analytics
Example: Local Data Caches store Images
Example: Local Data Caches store Text
Challenges for the ML Modeler

You and your ML model are here...

Cloud

In FL you can’t directly inspect your data set.

... but the Training Data is here

Mobile Device
Challenges for the ML Modeler: Debugging

- “I’m observing metrics outside the expected range, why?”, or ...
- “My trained model is behaving pathologically, why?”
  - Inspect image data set, realize there’s a pixel range mismatch b/w examples and expected
    \[ x \in [0, 255] \text{ vs. } x \in [-1.0, 1.0] \]
  - Inspect image data set, realize bug in preprocessing (some images have intensity inverted)
  - Inspect text data set, realize bug in tokenizing (some tokens incorrectly concatenated)

['Ohthere', 'is', 'also', 'Gears', 'of', 'war', ',', 'other', 'character', 'in', 'the', 'halo', 'universe',...]
Challenges for the ML Modeler: Data Set Augmentation

- "I need to gather input samples (features), to pass to humans to apply labels"

\[
\begin{bmatrix}
\bar{x} \\
\bar{y}
\end{bmatrix}
\begin{bmatrix}
\bar{x} \\
\bar{y}
\end{bmatrix}
\begin{bmatrix}
\bar{x} \\
\bar{y}
\end{bmatrix}
\begin{bmatrix}
\bar{x} \\
\bar{y}
\end{bmatrix}
\begin{bmatrix}
\bar{x} \\
\bar{y}
\end{bmatrix}
\]

- "I have a biased dataset, I need to gather samples of underrepresented classes"

\[
\begin{bmatrix}
\bar{x} \\
\bar{y}
\end{bmatrix}
\begin{bmatrix}
\bar{x} \\
\bar{y}
\end{bmatrix}
\begin{bmatrix}
\bar{x} \\
\bar{y}
\end{bmatrix}
\begin{bmatrix}
\bar{x} \\
\bar{y}
\end{bmatrix}
\]
Challenges for the ML Modeler: Data Set Augmentation

- “I need to gather input samples (features), to pass to humans to apply labels”

- “I have a biased dataset, I need to gather samples of underrepresented classes”

How do you do these types of things when you can’t directly inspect the data?
Differentially Private, Federated Generate Models
Federated Generative Models

- Can we train via federation a model capable of synthesizing privatized, novel examples that match the distribution of the private, decentralized dataset?

- Privacy is paramount
  - A Federated Generative Model should not be able to memorize data unique to an individual

- Many options at our disposal:
  - Differentially Private, Federated GANs (for Image Applications)
  - Differentially Private, Federated Recurrent NNs (for Text Applications)
  - ...
Federated Generative Models

Take an image model debugging example...
Federated Generative Models

Add logic to gather samples in cases where metrics fall outside expectations
Federated Generative Models

Train a DP Federated GAN ...
Federated Generative Models

Train a DP Federated GAN and synthesize novel images (at the cloud) that match the characteristics of images in private dataset. Observe intensity inversion.
Federated Generative Models

Another example: debiasing

\[
\begin{bmatrix}
\bar{x} \\
\bar{y}
\end{bmatrix}
\]

cloud dataset
Federated Generative Models

Idea: first, train a discriminator that indicates if input is ‘like’ cloud dataset.
Federated Generative Models

Idea: first, train a discriminator that indicates if input is ‘like’ cloud dataset. Deploy to devices, and mark negatives.
Federated Generative Models

Train a DP Fed Generative Model on samples of the unrepresented data
Federated Generative Models

Use the DP Fed Generative Model to synthesize novel examples of unrepresented data...
Federated Generative Models

... and then developer uses synthesized examples to inform additional data collection, etc.
Federated Generative Models

Final example: beyond self-labeling limitations

On-device experience generates feature ‘x’, but not the label ‘y’
Federated Generative Models

Train a DP Fed Generative Model on examples of features in the private, decentralized data.
Federated Generative Models

Use the DP Fed Generative Model to synthesis novel examples of features the developer can use ...
Federated Generative Models

Use the DP Fed Generative Model to synthesis novel examples of features the developer can use (i.e., label)
Federated Generative Models

Now have a dataset with features and labels
(Differentially Private) Federated GAN Algorithm
Quick Review of GANs
GANs: Generative Adversarial Networks

Two distinct NNs...
- ‘G’ tries to emit values that emulate a distribution observed in ‘real’ data
- ‘D’ tries to decide whether a value is ‘real’ or ‘generated’

Discriminator: $D(x)$
- $x_{\text{generate}}$ $\sim P_{\text{real}}(x)$
- $x_{\text{real}}$ $\sim P_{\text{real}}(x)$

Generator: $G(z)$
- $z \sim \mathcal{N}(0,1)$
GANs: Generative Adversarial Networks

Step 1: Train the Discriminator

\[
\min D(G(z)) + 1 - D(x_{\text{real}})
\]

(In this step, \( G \) is frozen, we only update parameters in \( D \))

\[z \sim \mathcal{N}(0,1)\]
GANs: Generative Adversarial Networks

Step 2: Train the Generator

\[ \max D(G(z)) \]

(In this step, \(D\) is frozen, we only update parameters in \(G\))

Generator: \(G(z)\)

\[ z \sim \mathcal{N}(0,1) \]

Discriminator: \(D(x)\)

\[ x_{\text{generate}} \]

\[ p(x \sim P_{\text{real}}(x)) \]

\[ x_{\text{real}} \sim P_{\text{real}}(x) \]
GANs: Generative Adversarial Networks

- Iteratively train the two NNs

At convergence, you’ve got a NN (‘G’) which can generate novel instances that emulate the real world
  - E.g., generate novel images of human faces
FedAvg-GAN

Mobile Device

Local Training Data

Initial (Untrained) Gen. Model.

Initial (Untrained) Disc. Model.
FedAvg-GAN

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

Initial (Untrained) Disc. Model.

Mobile Device

Local Training Data
FedAvg-GAN

2. Selected devices download the current discriminator model parameters (+ fake data).
FedAvg-GAN

3. Users compute a discriminator update using local real training data (+ fake data)
FedAvg-GAN

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

\[ \sum \]

Repeat until convergence.
FedAvg-GAN

5. Server computes a generator update, using updated discriminator
4. Server aggregates users' updates into a new DP discriminator model.

Add Gaussian noise proportional to sensitivity.
DP-FedAvg-GAN

5. Server computes a generator update, using updated DP discriminator. Generator is also DP, via post-processing property.
Federated GAN
Example Problem:
Debugging Image Classification
Example Federated GAN Problem

On-device inference network classifies handwritten numbers and letters. It expects raw images (from the upstream data pipeline) where background is black and character is white.
Example Federated GAN Problem

After application update, the classification accuracy drops
Example Federated GAN Problem

Train a DP Federated GAN and synthesize novel images (at the cloud) that match the characteristics of images in private dataset. Do this both for subsets with high and low class accuracy.
Example Federated GAN Results

Population Description

EMNIST Dataset, 50% of Devices have their images ‘flipped’ (black<-> white)

Sub-Population Description

Devices where data classifies with ‘low’ accuracy

Example of Real Data on Devices in Sub-Population

GAN after 0 rds

GAN after 1000 rds

Devices where data classifies with ‘high’ accuracy
Example Federated GAN Results

Example of Real Data on Devices in Sub-Population

Now the modeler can discern this difference ...

... indicating that this is the problem
Conclusion
FL Research

FL Workshop in Seattle 6/17-18

2016 8 academic papers
2017 135
2018 256
2019 265 so far …

Multiple workshops and tutorials this year (CVPR, Google, IJCAI, NeurIPS, ...)

Google
to pursue research in ML/data science for decentralized data with privacy by default.
TensorFlow Federated (TFF)
An OSS framework for federated computation on decentralized data

tensorflow.org/federated
github.com/tensorflow/federated
TFF - What’s in the box

- **Federated Learning (FL)**
  - Implementations of federated training/evaluation
  - Can be applied to existing TF models/data

- **Federated Core (FC)**
  - Allows for expressing new federated algorithms
  - Local runtime for simulations
train_data = ... # uses tff.simulation.datasets.emnist.load_data()
model_fn = lambda: tff.learning.from_keras_model( ... )

train = tff.learning.build_federated_averaging_process(model_fn)

state = train.initialize()
for _ in range(5):
    state, metrics = train.next(state, train_data)
    print(metrics.loss)

eval = tff.learning.build_federated_evaluation(model_fn)
metrics = eval(state.model, test_data)
I WANT YOU to pursue research in ML/data science for decentralized data with privacy by default.

TensorFlow Federated (TFF)
An OSS framework for federated computation on decentralized data

tensorflow.org/federated
github.com/tensorflow/federated