01

Distributed Machine Learning

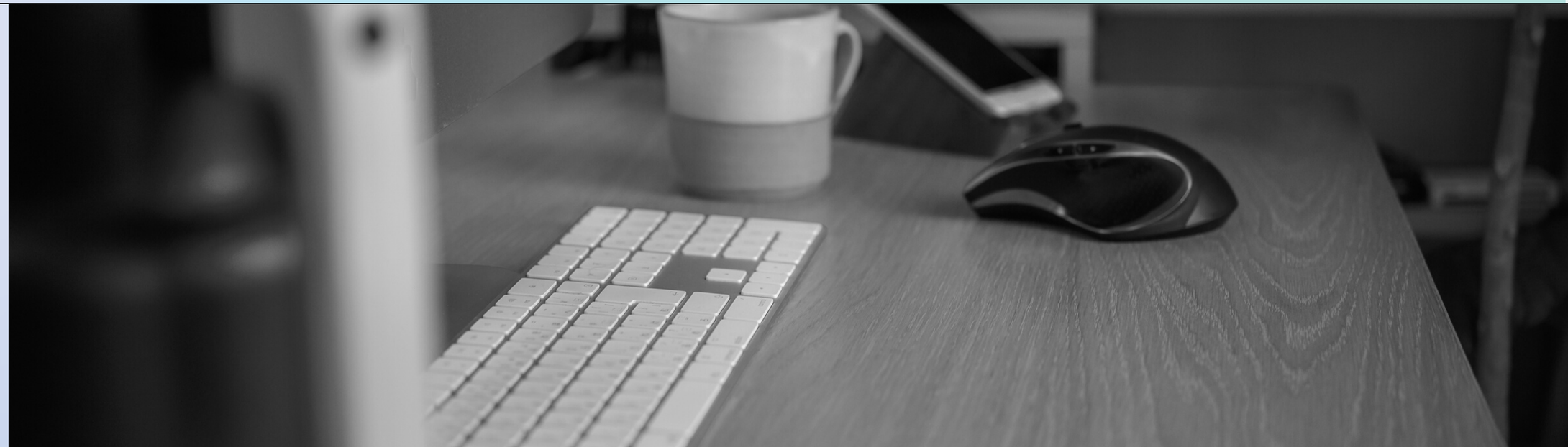
Federated Learning

A project by Apex.

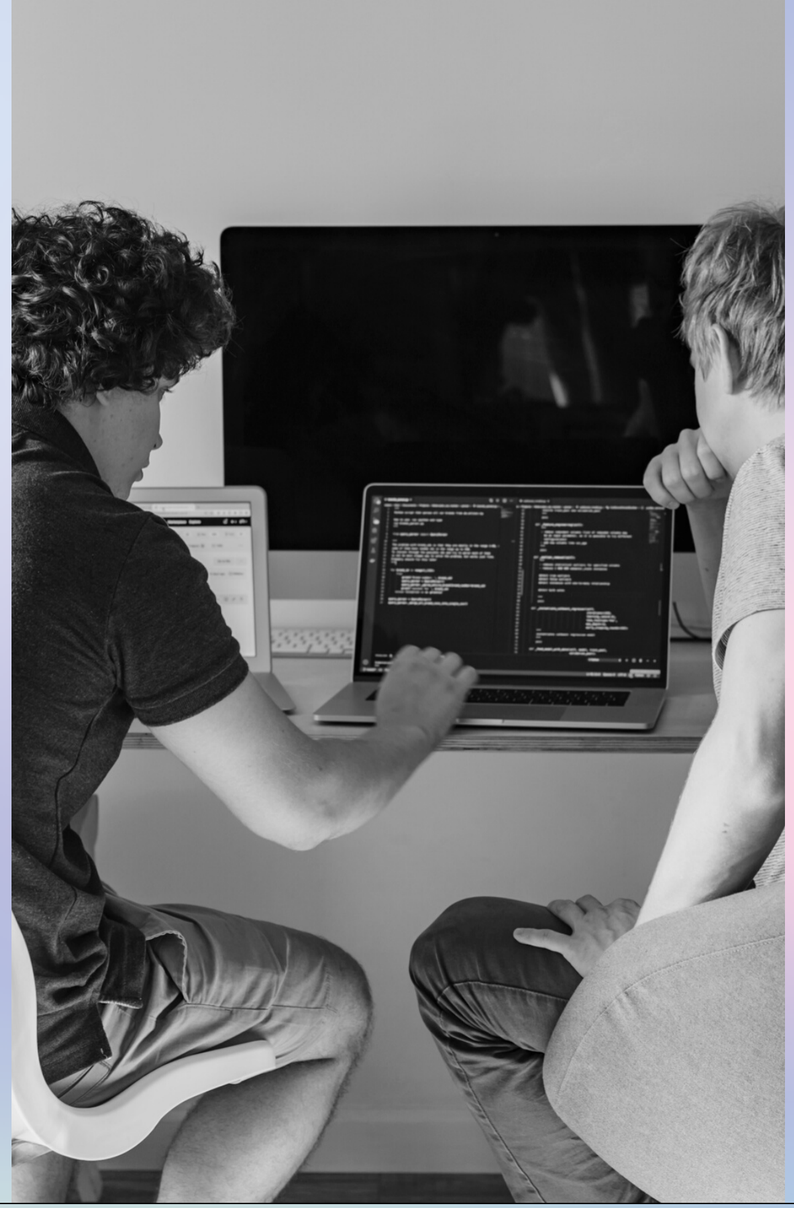
02

Outline

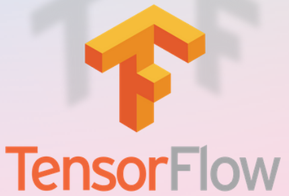
Federated Learning is a novel solution proposed to solve this challenge of maintaining data integrity and security while still training the models.



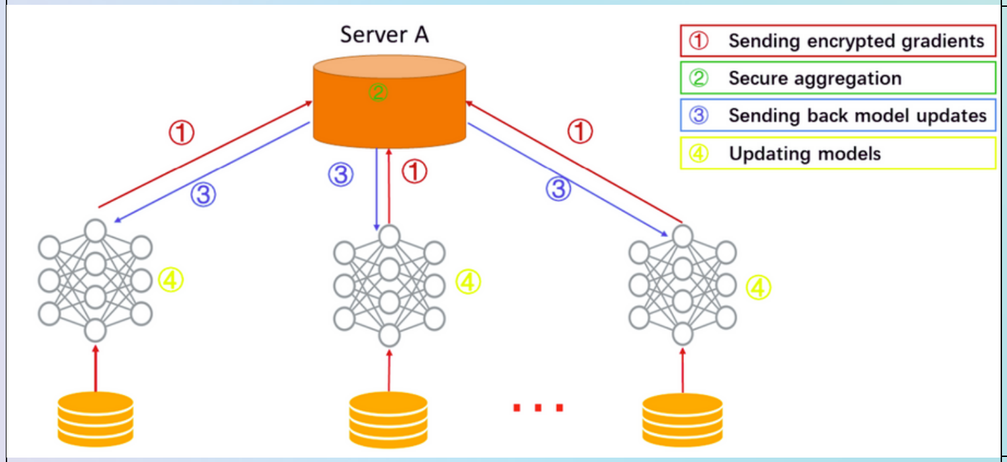
03



Tech Stack



Overview



01. Collect nodes to compute the model on.

Select a small subset of client devices which will download the trained model.

02. Start training on the client devices.

This subset trains the model on data either generated by the client or provided to the client.

03. The model updates are sent to the server

The server uses an averaging algorithm to account for the changes sent by the client.

05

Approach

Instead of creating a distributed network and we will use a single model which will be distributed to all the clients.

Step 1: Get the dataset.

The dataset used is MNIST.

Step 2: Generate the model.

We will create initial model in Python.

Step 3: Prepare for distribution

Using tensorflowjs_converter ,we will convert model.h5 to model.json for tfjs.

Step 4: Acquire client nodes.

We will simulate clients using multiple CPUs with each CPU acting as a client.

Step 5: Distribution

A subset of data(skipping i.i.d) and the server model is given to each client on which it trains and saves weights in model.json.

Step 6: Collection

Using tensorflowjs_converterconvert model.json to h5 files. We will average all models and update the server model which can again be distributed to clients.

06

Accomplishments

We are able to accomplish >85% accuracy based on single training session with minimal operating priority on node devices.

→

07

Further Improvements

One of the major pitfalls of this implementation is that the data is not i.i.d because we randomly sampled our data-set. This might lead to fallacies in our trained model.

This is a real challenge when it comes to Federated Learning. Here we could have easily manipulated data to fit our needs but we skipped that step keeping in mind the real world implications of the same.

Federated Learning also introduces another risk of a model poisoning into the system. An injector could potentially train a model to be equally accurate but towards a set of a single feature and send the updates to server.

“Those who can imagine anything, can create the impossible.”

- Alan Turing

Let's be courageous in our imagination.
Don't be afraid to create!

09

Meet our Team

Feel free to ask any questions now:

Aaryan Kapur

E18CSE004

Abhinav Robinson

E18CSE006

Aniket Chowdhury

E18CSE015

Ashok Kumar

E18CSE029

Jai Chhabra

E18CSE071