Communication Efficient Learning of Deep Networks from Decentralized data

H. Brendan McMahan Eider Moore Daniel Ramage Seth Hampson Blaise Aguera y Arcas

Team 07: CyberDyneSystems

Kapil Kanna

Adithyaa Seyyone

Yuvan Dhurghesh

Akshaay

Problem Statement

Mobiles, Tablets and Laptops are the closest and the most frequently carried edge devices. Modern mobile devices house sensors and GPUs.
They have huge amounts of data and compute, models trained on which can be used for intelligent applications.

The problem here is that the data in these devices are sensitive hence, privacy and security is a paramount concern.
Traditional centralized approach doesn't guarantee the safety of the data sent to the server. This is where Federated Learning steps in.

Why this is an edge problem?

In the decentralized approach proposed by the paper, instead of bringing the data to the model, the model to brought closer to the data source. The server sends a copy of the global model to the edge and the edge trains the model with the local data.

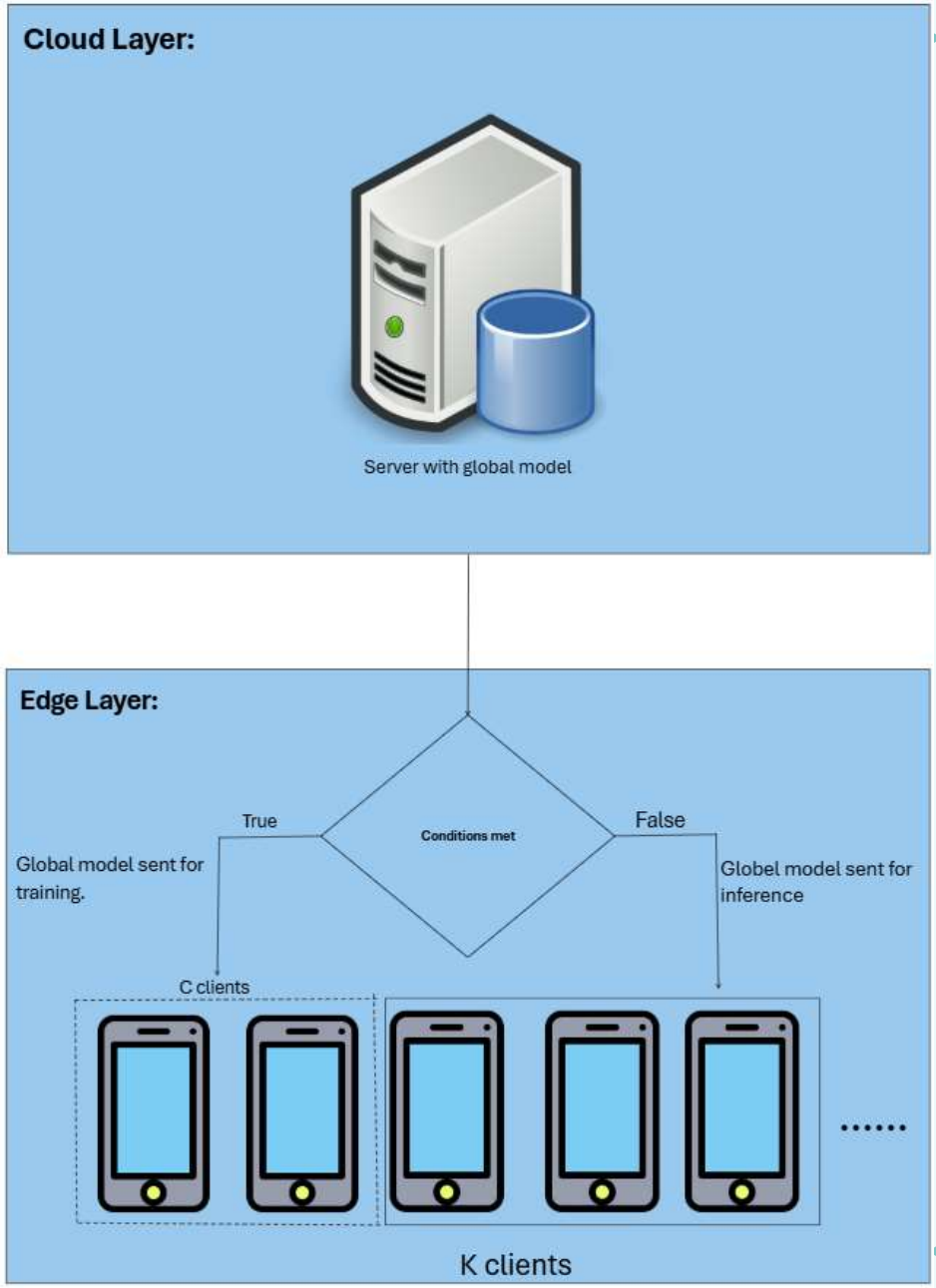
The updates to the global model are then sent to the server. The received updates are then aggregated using the FedAvg (Federated Averaging) algorithm proposed by the paper. This cycle continues until the model reaches convergence.

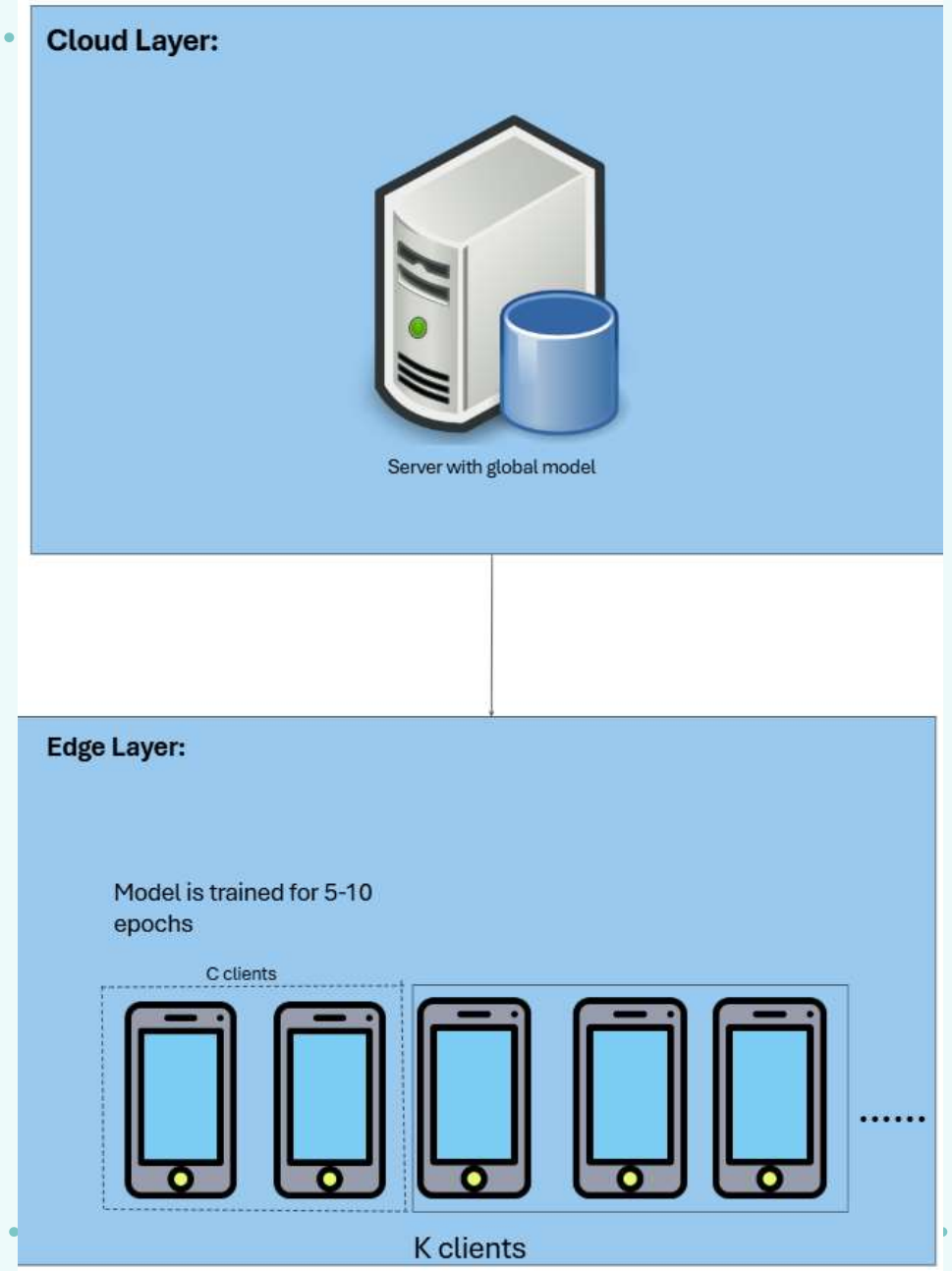
Problems dealt

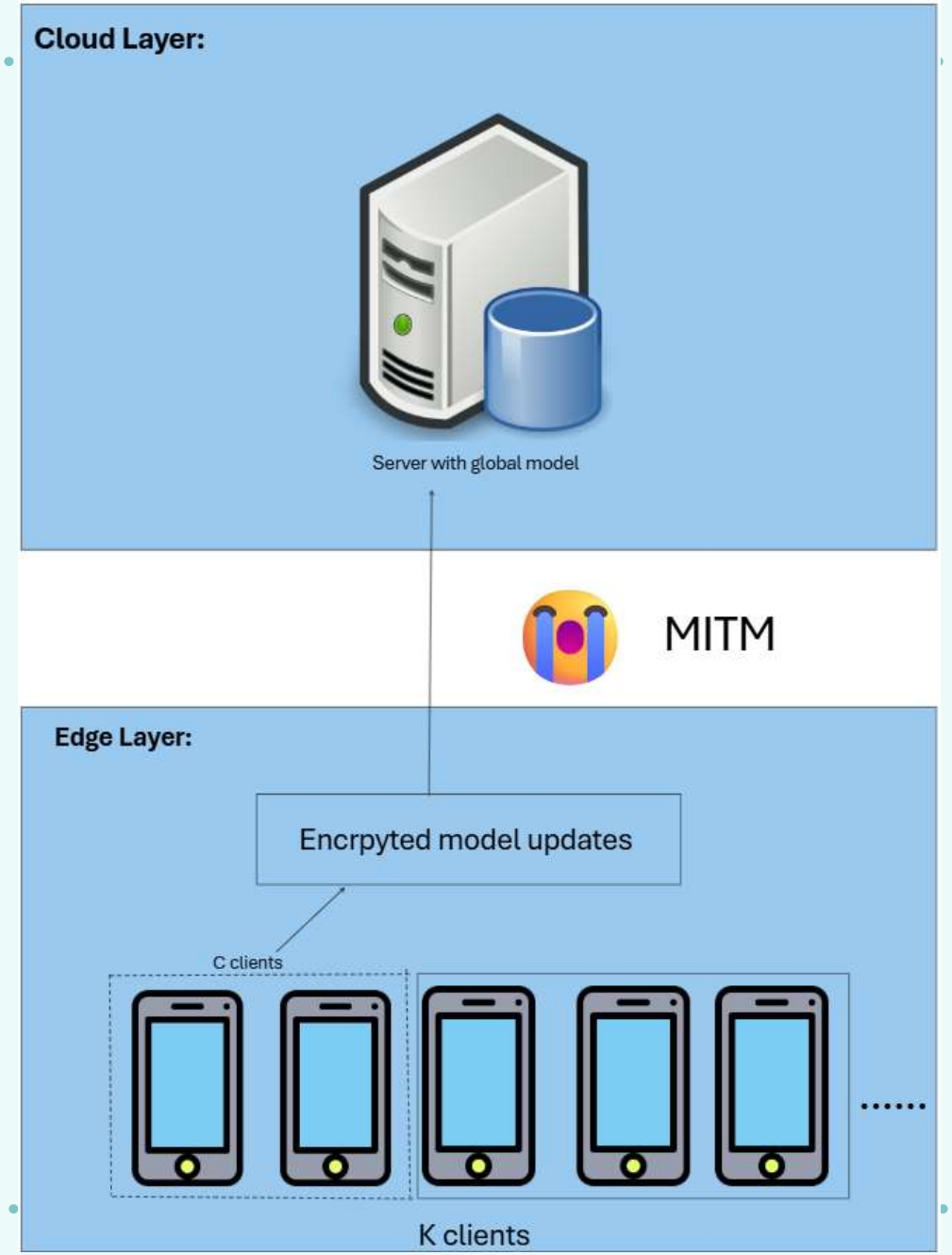
1) Number of clients to be considered
2) The communication cost is higher than in the centralized setting
3) Privacy measures to prevent reconstruction of data
4) Limited interference with client usage
5) Clients with non-IID and unbalanced data

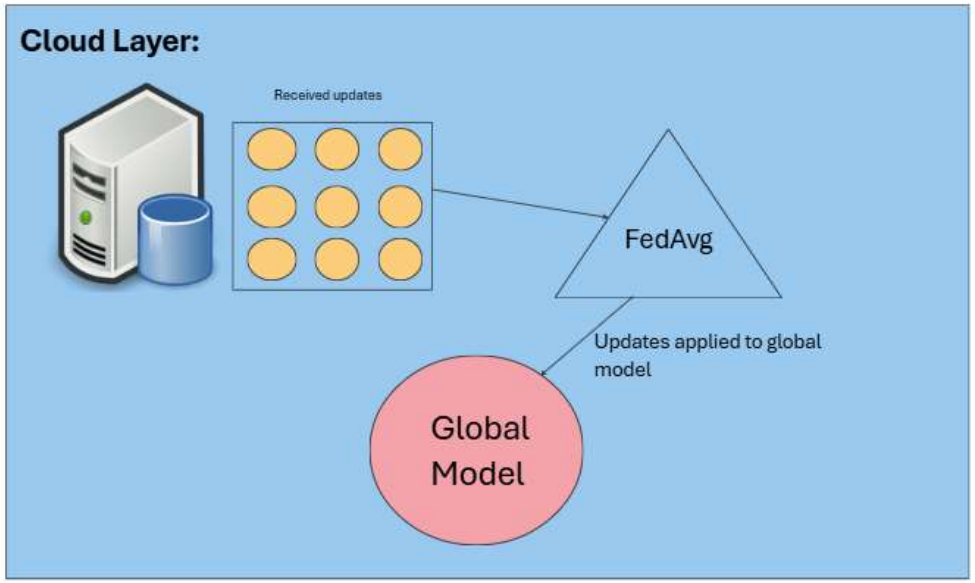
Implementation

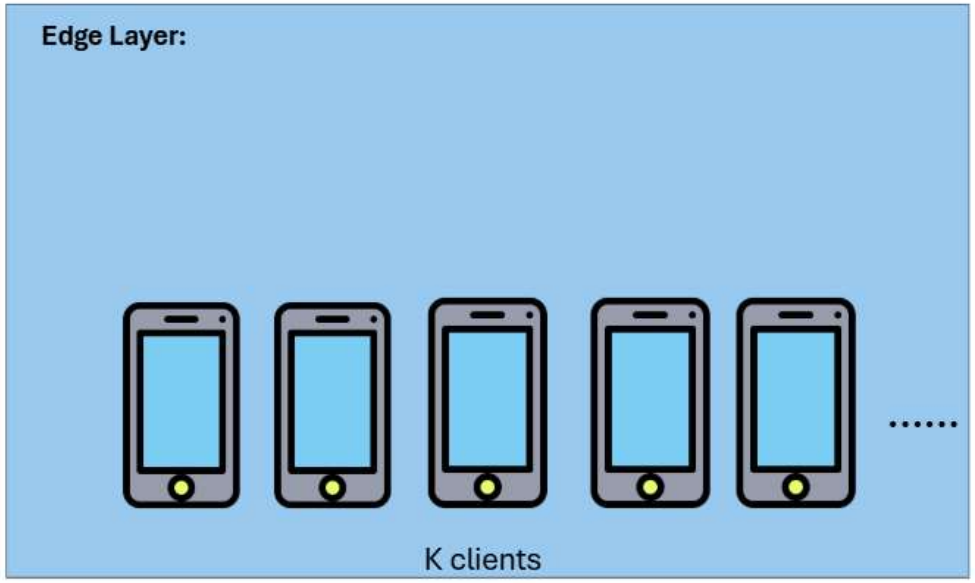
& Architecture


Cloud Layer:
Server with global model
Edge Layer:
Conditions met
True
False
Global model sent for training.
Globel model sent for inference
C clients
K clients


Cloud Layer:
Server with global model
Edge Layer:
Model is trained for 5-10 epochs
C clients
K clients


Cloud Layer:
Server with global model
MITM
Edge Layer:
Encrpyted model updates
C clients
K clients


Cloud Layer:
Received updates
FedAvg
Updates applied to global model
Global Model


Edge Layer:
.....
K clients

Federated Averaging Algorithm

-> FedAvg is the foundational algorithm in federated learning.
-> A central server initializes a model (e.g., 199K-parameter perceptron on MNIST, 1.6M CNN on CIFAR-10, or 866K LSTM for text).
-> A subset of clients is randomly selected; each performs local training using SGD on private data for a few epochs.
-> Clients return only their updated weights.
-> The server performs weighted averaging (by data size) to form a global model.
-> The process repeats across communication rounds.
-> Reduces communication rounds by up to 100× vs. centralized SGD.

Enhanced FedAvg Algorithm

To address FedAvg limitations, several improvements have been proposed under Enhanced FedAvg:

Resource-aware selection: Chooses clients based on CPU, bandwidth, and battery to speed up training (40% faster in IoT tests). (e.g., Resource-Aware FedAvg)Stability under non-IID data: Adds a regularization term to constrain local updates, improving convergence. (e.g., FedProx)Adaptive local workloads: Dynamically adjusts batch sizes and epochs using update history and latency, reducing required rounds by 30%. (e.g., AdaFedOpt)Smart client selection: Uses learning-based policies to choose clients that maximize performance per time/energy unit (25% faster convergence). (e.g., RL-based Scheduling)

Limitations of Federated Learning Algorithms

-> Data heterogeneity can lead to unstable convergence and biased global models.
-> Encryption overhead adds compute cost, though necessary for privacy.
-> Malicious clients may poison updates, but detecting them remains hard.
-> Inconsistent data formats or accents (e.g., American vs. British speech) hurt performance if real-world data differs from assumptions.
-> In federated setups, communication (not compute) is the main bottleneck. Network reliability and bandwidth are crucial constraints in deployment.

State-of-the-art literature

- ***SCAFFOLD*** (Karimireddy et al.) uses control-variates (variance reduction) to correct client drift (non IID); provably fewer rounds under heterogeneity
- ***Practical Secure Aggregation*** (Bonawitz et al., Google) — an Secure Multi Party Computation protocol built for FL scale (failure-robust, communication-efficient).
- ***Adaptive Federated Optimization*** (Reddi et al.) — federated versions of Adam/Adagrad/Yogi show empirical gains and better stability on non-IID data.

Task Split

- **Adithyaa Seyyone:** Environment & Topology setup
- **Kapil Kanna:** FedAvg integration with workflow
- **Yuvan:** Client selection algorithm based on parameters
- **Akshaay:** Metrics & Experimentation

THANK YOU