Federated Learning for CIFAR-100

Name : Yachi Darji
Data Science, Illinois Institute of Technology; Chicago, IL, United States

Abstract : The paper represents the implementation of Federated learning in the CIFAR-100 dataset for image classification using Federated averaging algorithm (FedAvg) and PyTorch. System simulated 64 distributed clients with non - iid data partitioned using dirichlet distribution(α= 0.4). The global model implements Resnet18 architecture on 100 classes and trained using FedAvg with ThreadPoolExecutor to enable local training between 16 clients per communication round. Additionally, a web application developed using FastAPI & docker containerized on Chameleon Cloud, which shows remote inference capacity. After 100 rounds, global model achieves training accuracy **91%** and testing accuracy **66%**. These accuracy shows stable convergence under non iid data condition. The final deployed model allows user to upload image and recieves top 5 prediction within browser. Our system addresses key challenges in FL including Non - IID data distribution, client selection strategies and model deployment. The implementation shows practical FL from simulation to production & deployment.

I. Introduction

Federated learning (also known as collaborative learning) is a machine learning technique in a setting where multiple entities (often called clients) collaboratively train a model while keeping their data decentralized, rather than centrally stored.[?] Traditional centralize training requires data must aggregate to server that raises privacy, security and scalability issue. FL address these issue by enabling model to train locally and only share updated weights to server instead of data. FL has multiple applications like mobile device personalization, healthcare analytics, industrial IoT where data privacy matters. Conversely, FL is concerned with issues such as statistical heterogenity (non - IID data), data privacy and minimization. In real world settings, data distribution differs device to device, which can slow convergence and affect the global model's accuracy. This assignment aims to investigate performance of FL under non iid data condition using CIFAR-100 dataset.

FL simulation implemented in Python using PyTorch with 64 clients train local model in parallel using `ThreadPoolExecutor`. Every clients have unique data for training generated using Dirichlet distribution (α= 0.4) to simulate non - iid behaviour. The global model uses Resnet18 arch for 100 labels and optimized using FedAvg.

Objectives are:

1) Implement FedAvg algorithm with non-IID distribution.
2) Analyse performance under heterogeneous data condition.
3) Develop deployable web services for model inference.
4) Evaluate system scalability and remote accessibility.

II. Methodology

A. Federated Learning Simulation

The FL simulation is developed using Python (PyTorch) and FedAvg algorithm. The setup consists of 64 distributed clients trained concurrently using `ThreadPoolExecutor` module. The federated learning system was completely developed from scratch in PyTorch. In data preparation, distribution introduces heterogeneity within the classes. The server aggregates local model's parameter using weighted average, that gives more importance to client with large data size.

Model Architecture Resnet-18 model architecture used with specific modification. The first layer was replaced with 3×3 kernel. No max pooling operation to preserve feature of small images. Output layer change from 1000(Imagenet) to 100 (Label) correspond to CIFAR-100 dataset. Training follows FedAvg algorithm by McMahan et al. Each client trained locally using 2 epochs using Adam optimizer (learning rate = 0.001). Model weights are updated on central server through weighted average proportional to each client's datasize.

`ThreadPoolExecutor` was used with four maximum parallel worker (`MAX_WORKER = 4`). Each thread executes an independent local training task and returns updated model parameter to main thread for updation.

Non-IID data

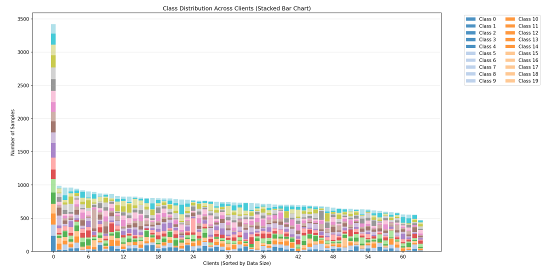


Fig. 1. Non-IID data distribution

Non-IID heterogeneity is difficult as some client dominates training due to class imbalance causes uneven gradient update and slower convergence. Running 64 clients together introduces communication and aggregation delay. The simulation used 64 batchsize and 2 local epoch per round as limited CPU power. All training data metrics saved in pandas dataframe for tracking.

Fig 1 represents how uneven data is distributed across all clients. Some client hold many samples from very few class, while other clients have very few samples of many classes. This non - iid distribution makes simulation more realistic and directly affects training convergence and generalization. Clients on left side have highly skewed classes while clients on right side have smalller and diverse classes. This uneven distribution perfectly represents real world federated scenario as IoT network where userdata is biased.

III. Experimental Design & Results

a. Training progress and convergence The performance of federated system is evaluated by tracking loss and accuracy metrics over 250 rounds for training and testing. The observation below analysis are based on two main plots : accuracy curves and loss curves.

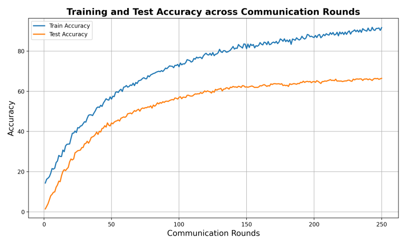


Fig. 2. Accuracy Curves

As shown in *Fig 2*, training and testing accuracy rises over communication rounds. Training accuracy reaches to 91.70%, wherease test accuracy levels off at 66.47%. The upward trend shows model is learning well but could not generalizes effectively. The steady gap between training and testing represents how client data imbalance and distribution affect the federated learning.Overall improvements demonstrates that FedAvg algorithm can learn global representation under heterogeneous data conditions.

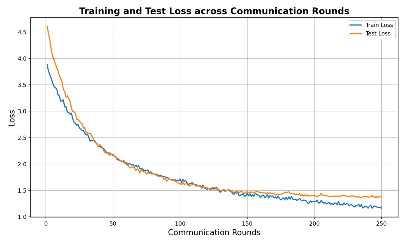


Fig. 3. Loss Curves

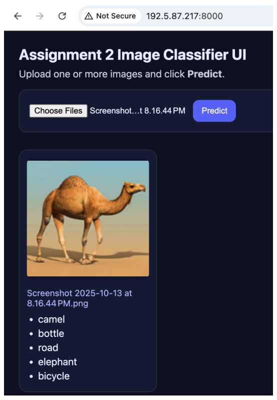
From *Fig 3*, training and testing loss started to drop sharply during early rounds that shows model converges efficiently and followed by gradual stabilization after 150 rounds. The training loss decrease below 1.16, but test loss still stays high around 1.36. This behaviour indicates that model generalizes well despite of non - iid data.

Together,the plots confirms federated learning setup works as intended and provides clear insights into model convergence and generilizability under non iid behaviour.

B. Web Application and Deployment

The global model (*model_global.pth*) was integrated to FastAPI app design for real time inference. The app provided minimal User Interface where user can upload image, preprocess them and get top 5 prediction on the same page. Backend performs preprocessing and executes inference through Resnet18 model after recieving image.

Fig 4 displays Web app on Chameleon Cloud that required careful Docker configuration to ensure portability and network accesibillity. The container was configured to listen on `http://192.5.87.217:8000/` and utilize a floating IP for remote access. A `docker-compose.yml` file arranges both services : 1 one container for training process (`fl-training`) and another for web interface (`fl-webapp`).



Assignment 2 Image Classifier UI

Upload one or more images and click **Predict**.

Screenshot 2025-10-13 at 8.16.44 PM.png

- camel
- bottle
- road
- elephant
- bicycle

Fig. 4. Web Application

IV. Conclusions and Future Work

The system performed effectively, but several challenges were observed. Heterogeneous data distribution causes slower convergence and performance gaps between training and testing. These result emphasize trade-off between data privacy and global model generalization in FL system.

Futurework will focus on improving convergence rate. Integrate optimization such as FedProx or adaptive client weighting could reduce the impact of non - IID data.

Acknowledgment

This work was part of course CS595 – Decentralized Machine Learning Systems at Illinois Institute of Technology. I acknowledge the use of Chameleon Cloud resources and thank the Professor for their guidance.