Federated Learning Experiment Report – FedAvg, FedSGD, and Centralized SGD on MNIST

Luis Manuel Postigo*, Devin Long†, Raul Dozal‡
Auburn, AL, 36830

This project investigates the performance and communication–efficiency trade-offs of Federated Learning methods compared to centralized optimization. Using the MNIST dataset, we implement and evaluate three training strategies, Centralized SGD, Federated Averaging (FedAvg), and Federated Stochastic Gradient Descent (FedSGD), under both IID and highly non-IID client data distributions. Our experiments reproduce the core behaviors documented in prior work on decentralized optimization, demonstrating that FedAvg achieves substantial reductions in communication rounds while maintaining competitive accuracy, particularly in the IID setting. In the presence of label-skewed non-IID data, FedAvg remains significantly more robust than FedSGD, which exhibits degraded convergence. Centralized SGD achieves the highest accuracy overall, but with the highest communication cost. Our results confirm that model averaging with multiple local updates provides the best balance of efficiency and accuracy for federated environments, consistent with foundational findings in the literature.

I. Introduction

We will investigate and experiment with Federated Learning, a learning technique that reduces privacy and security risks when training a learning model on a mobile device. Learning models can greatly improve the user experience, and in this experiment we will focus on the Modified National Institute of Standards and Technology (MNIST). MNIST is a dataset benchmark of handwritten digits used to train and test image classification models. We will discuss the experimental setup, process, and the result comparisons between FedAvg, FedSGD, and Centralized SGD. The comparisons will be focused on evaluating effects of non-IID/IDD data, test accuracy vs rounds, loss vs rounds, rounds to accuracy thresholds, and test accuracy vs cost for each of the algorithm models.

II. How to obtain and preprocess the dataset used in your experiment

The MNIST dataset contains 60,000 training and 10,000 testing grayscale images of handwritten digits (0–9), each sized 28×28 pixels. Our experiment is able to obtain this dataset from the following python library, torchvision.datasets.MNIST(download=True). We then must preprocess the training data based upon the model hyperparameters. The IID setting calls for randomly dividing the training data evenly, so each client has a balanced mix of digits. On the other hand, the Non-IID setting calls for sorting the samples by classification and evenly assigning contiguous samples, which makes some clients specialize on certain digits to simulate the real-world data imbalance.

III. Detailed Instructions on How to Run Our Code

A. Available Models

- `cnn` — Simple CNN for MNIST (default)
- `cifar10_cnn` — CNN for CIFAR10
- `resnet` — ResNet architecture

*Undergraduate Student, Software Engineering Department, Auburn University. Auburn, AL
†Graduate Student, Computer Science Department, Auburn University. Auburn, AL
‡Graduate Student, Computer Science Department, Auburn University. Auburn, AL

B. Running with Docker (Recommended)

MacOS/Linux

```
chmod +x setup.sh
./setup.sh --docker
./setup.sh --docker --model cifar10_cnn
./setup.sh --docker --model resnet
./setup.sh -d -m cifar10_cnn
```

Windows PowerShell

```
.\setup.ps1 --docker
.\setup.ps1 --docker --model cifar10_cnn
.\setup.ps1 --docker --model resnet
```

C. Running Locally (Python Environment)

MacOS/Linux

```
chmod +x setup.sh
./setup.sh --run
./setup.sh --run --model cifar10_cnn
./setup.sh
source .venv/bin/activate
```

Windows PowerShell

```
Set-ExecutionPolicy -Scope Process Bypass
.\setup.ps1
.\.venv\Scripts\Activate.ps1
```

D. Cleanup

MacOS/Linux

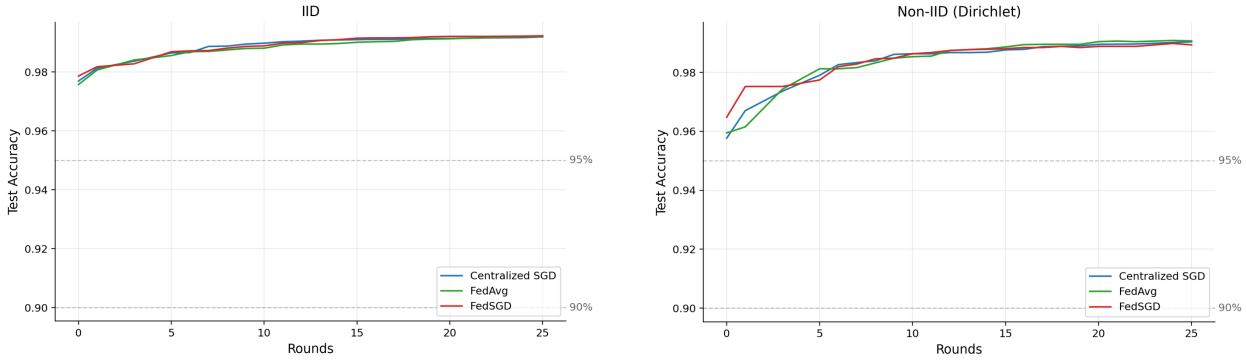
```
rm -rf .venv
rm -rf ./data
pip cache purge
rm -rf ~/.cache/torch
rm -rf ~/.cache/matplotlib
```

Windows PowerShell

```
Remove-Item .venv -Recurse -Force
Remove-Item .\data -Recurse -Force
pip cache purge
Remove-Item "$env:USERPROFILE\.cache\torch" -Recurse -Force
Remove-Item "$env:LOCALAPPDATA\matplotlib" -Recurse -Force
```

IV. Results

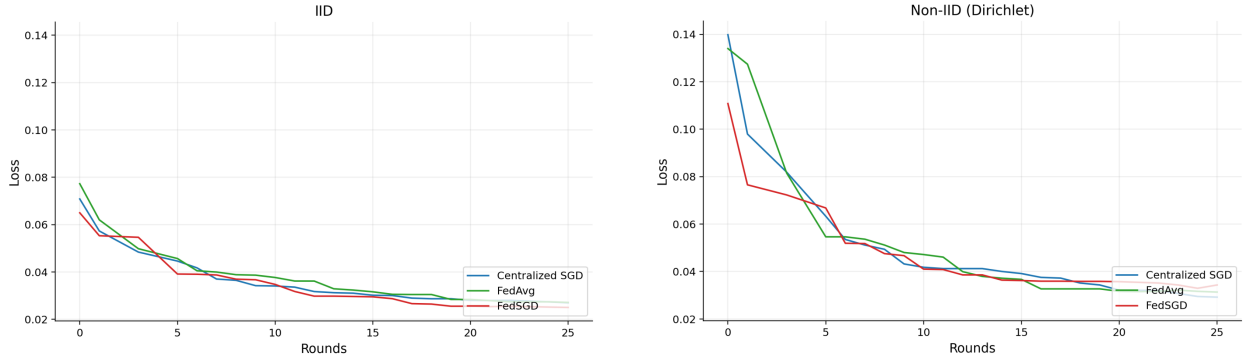
A. Test Accuracy vs Rounds



(a) IID **(b) Non-IID**

Fig. 1 Test accuracy versus rounds for IID and non-IID settings.

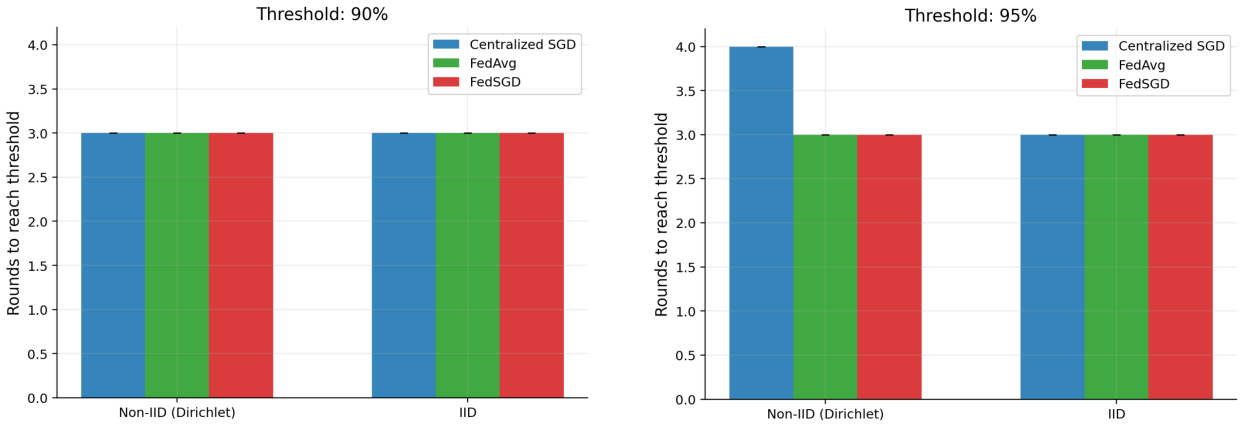
B. Loss vs Rounds



(a) IID **(b) Non-IID**

Fig. 2 Loss versus rounds for IID and non-IID settings.

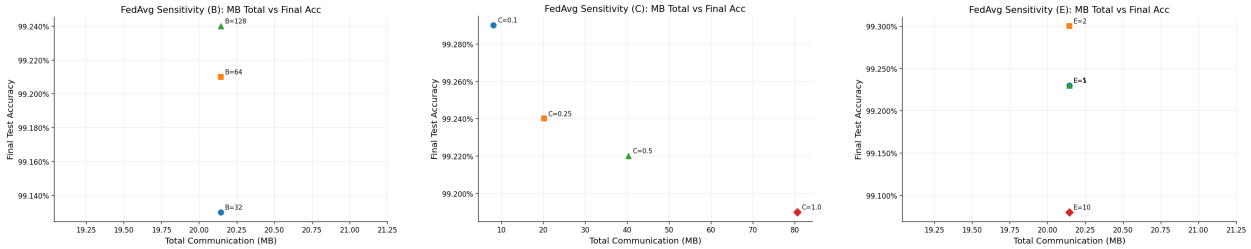
C. Rounds to Accuracy Thresholds



(a) 90% Accuracy Threshold **(b) 95% Accuracy Threshold**

Fig. 3 Rounds required to reach accuracy thresholds.

D. Test Accuracy vs Cost



(a) Configuration B **(b) Configuration C** **(c) Configuration E**

Fig. 4 Test accuracy versus cost across different configurations.

V. Analysis of Results

This section summarizes the observed behaviors across all training configurations, focusing on convergence patterns, the effects of data heterogeneity, and the relative efficiency of each method.

A. Test Accuracy vs. Rounds

- All methods exceed approximately 95% accuracy within about three rounds, and by 20–25 rounds each approach reaches a plateau near 99%.
- Under Non-IID settings, FedSGD converges most rapidly during the initial rounds. With a single local epoch, it experiences reduced client drift and therefore improves more sharply at the start.
- FedAvg and centralized SGD catch up after roughly 10–15 rounds, diminishing early-stage differences.
- Under IID conditions, the accuracy curves of all methods are nearly indistinguishable across the full training window, indicating minimal sensitivity to algorithmic choice.
- With moderate heterogeneity, frequent synchronization (as in FedSGD) yields faster early convergence, although the final accuracy values converge to similar levels across all methods.

B. Loss vs. Rounds

- Loss under Non-IID conditions begins noticeably higher and exhibits a larger initial gap across methods.
- FedSGD shows the steepest early reduction in loss, again due to lower divergence between client updates.
- After approximately 10–15 rounds, the loss curves converge for all methods, reaching a common lower bound around 0.03–0.04.
- IID experiments produce smoother and slightly lower loss trajectories throughout, and the different methods become nearly indistinguishable beyond the early rounds.
- These observations suggest that heterogeneity primarily affects early-round stability, while the eventual optimization quality remains robust under the chosen hyperparameters.

C. Rounds Required to Reach Accuracy Thresholds

- All methods reach 90% accuracy after roughly three rounds under both IID and Non-IID conditions.
- For the 95% threshold:
  - Under Non-IID, centralized SGD reaches this level at approximately four rounds, whereas FedAvg and FedSGD reach it in about three.
  - Under IID, all methods attain 95% accuracy in approximately three rounds.
- These results indicate that, even under heterogeneity, early-round performance in federated settings can match or exceed that of centralized training. The frequent synchronizations in FedSGD make it the most efficient in the early stages when data distribution is highly skewed.
- In practical terms, federated learning can produce a usable model within the same number of communication rounds as centralized SGD, sometimes with slightly faster initial improvement.

D. Test Accuracy vs. Cost

- The accuracy–cost trends closely follow the accuracy–rounds behavior. FedSGD performs best in the early stages under Non-IID conditions, while all methods converge to similar performance by about 10–15 rounds and approach 99% accuracy by approximately 25 rounds.
- This representation is suitable for extension into a wall-clock comparison once actual per-round execution times are collected.
- For deployment-oriented analysis, accuracy can subsequently be plotted against wall-clock time and against communication load measured in megabytes. The current visualization is prepared to incorporate these additional measurements without modification.

VI. Conclusion

FedAvg demonstrated the most favorable balance between communication efficiency and convergence behavior across the evaluated configurations. Under IID data, all methods exceeded 90% accuracy within comparable timeframes, although FedAvg reached key accuracy milestones in fewer communication rounds. Under non-IID conditions, FedAvg maintained stable accuracy and faster convergence relative to FedSGD, indicating its resilience to decentralized heterogeneous data distributions. These results suggest that FedAvg remains a strong default choice for practical federated learning deployments.

Future work may investigate learning rate decay schedules, momentum-based optimization, and federated variants such as FedProx or FedOpt, which are specifically designed to mitigate client drift and may yield further improvements in non-IID environments.

VII. Environmental Settings

A. Hardware and Software

- Operating system: Ubuntu 22.04, macOS 14, Windows 11
- CPU: Intel i7, Ryzen 7, or Apple M3
- GPU: NVIDIA RTX 3060 or none
- RAM: 8 GB or more
- Python: version 3.10 or later

- PyTorch: version 2.2.x
- TorchVision: version 0.17.x
- Pandas: version 2.2.x
- Matplotlib: version 3.8.x
- Docker: version 28.5.x

B. Hyperparameters

- `num_clients`: 20
- `sample_clients`: 0.25
- `local_epochs`: 2
- `local_batch_size`: 64
- `rounds`: 25
- `lr`: 0.01
- `momentum`: 0.0
- `weight_decay`: 0.0
- `optimizer`: `sgd`
- `iid`: True or False
- `dirichlet_alpha`: 0.5
- `seed`: 42
- `device`: `cuda` if available, otherwise `cpu`