Assignment - 3 DAI
Federated Learning

G Mukund (B21CS092)
Drithi Davuluri (B21AI055)

Dataset

The CIFAR-10 dataset was used. This dataset used in the Federated Learning implementation reflects real-world data heterogeneity, non-IID data, and potential concept drift across clients. This heterogeneity poses challenges for the algorithm, requiring robust techniques to learn a globally consistent model from diverse local datasets.

The dataset is divided into 50 client datasets for our experiments.

Experimental Setup

Common Setup

Batch Size: 10 for clients and 1000 for test data
Learning Rate: 0.01
Number of Local Epochs: 5
Maximum Rounds: 10

Model Architecture

All experiments utilized a CNN model with the following architecture:

Convolutional Layer 1: 32 filters, kernel size 5x5
Convolutional Layer 2: 64 filters, kernel size 5x5
Fully Connected Layer: 512 neurons
Output Layer: 10 neurons

Model Architecture

For our Federated Learning implementation, we defined a Convolutional Neural Network (CNN) as the base model.
Our CNN architecture included two convolutional layers with 32 and 64 filters respectively, and a kernel size of 5x5, responsible for extracting meaningful features from the input images. Each convolutional layeris followed by a max-pooling layer of size 2x2 to construct the feature map and introduce translation invariance. The output of the convolutional and pooling layers is then flattened and fed into two fully connected layers, the first with 512 units and the final layer with 10 units corresponding to the CIFAR-10 classes. Used the ReLU activation function throughout the network to introduce non-linearity and improve the model's learning capacity.

Federated Average Algorithm

Methodology

Common Setup

In addition to the model architecture, our Federated Averaging implementation includes several common setup components. We used the PyTorch `datasets. CIFAR10` module to load and preprocess the CIFAR-10 dataset, and then split the training data into 50 client datasets, each containing 1,000 samples, to simulate a federated learning scenario.

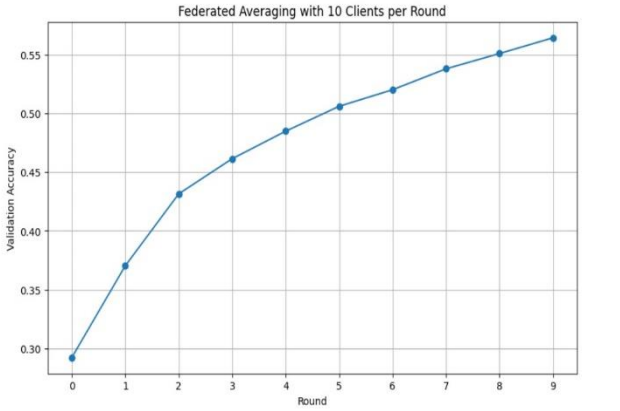
For the training parameters, we defined the loss function as Cross-Entropy Loss and used Stochastic Gradient Descent (SGD) as the optimizer for training the local and global models.
We then implemented a `validate` function to evaluate the global model's performance on the test dataset, to get the convergence and accuracy of the Federated Averaging algorithm over the training rounds.

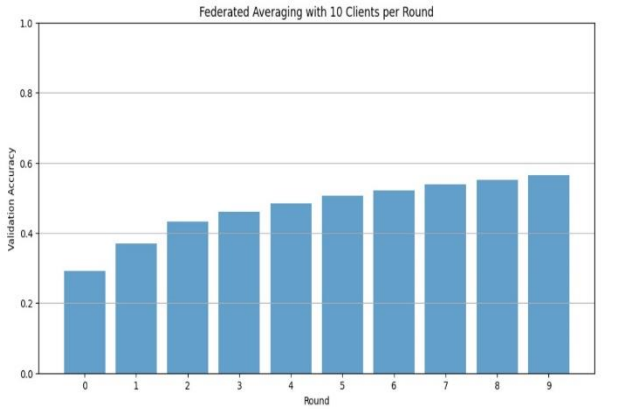
Federated Averaging Algorithm

The core of our implementation is the `fed_avg_experiment` function. This function selects clients, trains local models, aggregates updates using weighted averaging, updates the global model, and validates it on the test set, iterating this process to converge to a global model representing the collective knowledge of the participating clients.

Results

The Federated Averaging algorithm we implemented shows a steady increase in validation accuracy over the course of 10 training rounds. Starting from an initial accuracy of 0.2923, the model achieves a final accuracy of 0.5644, demonstrating the algorithm's ability to converge to a global model that effectively captures the shared patterns across the distributed client datasets.





Federated Personalized Learning (FedPer)

Methodology

Common Setup

The FedPer implementation integrates essential setup elements for consistent federated learning. Leveraging PyTorch's `datasets.CIFAR10`, we preprocess the dataset and split it into 50 client datasets, simulating a federated environment. Cross-Entropy Loss guides model evaluation, optimized via Stochastic Gradient Descent for both local and global models. The `validate` function monitors convergence and accuracy on the test dataset, ensuring robustness across training rounds.

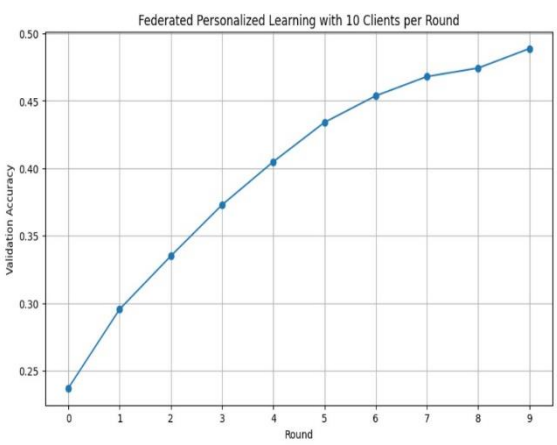
Federated Personalized Learning (FedPer)

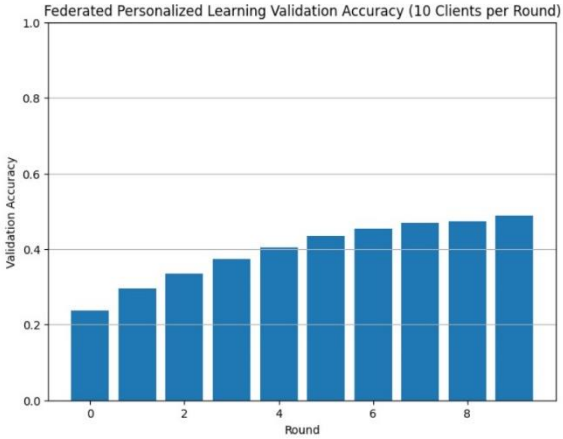
Federated personalized learning is an approach that combines the principles of federated learning and personalized learning. In this method, a global model is trained across multiple decentralized clients, and each client refines the model locally to cater to its

specific data. This ensures privacy, efficiency, and customization in model training for diverse datasets.

Results

In our federated personalized learning experiment, we trained a global model across decentralized clients. The validation accuracy showed a steady increase and improved over 10 rounds, starting from 23.7% and reaching 48.88%. This approach leverages client-specific refinements to enhance model performance, demonstrating the efficacy of decentralized training for personalized outcomes.





Federated Momentum Averaging (FedAvgM)

Methodology

Common Setup

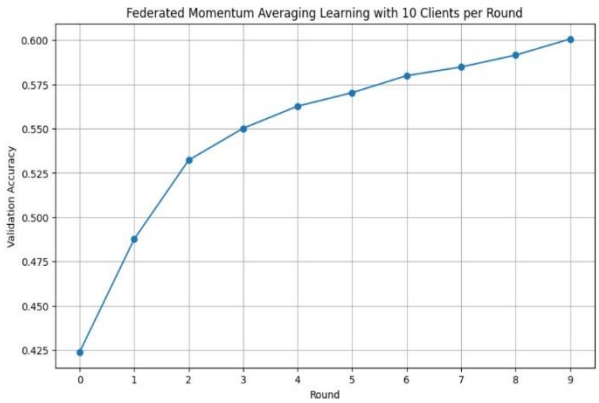
Our Federated Momentum Averaging (FedAvgM) setup incorporates several key components: a validation function, a local training function, a running model average function, and the federated learning experiment itself. We use the `validate` function to evaluate the model's accuracy on the test dataset by comparing its predictions against the true labels. The `train_client` function is responsible for training a local model on each client's dataset using stochastic gradient descent (SGD) with momentum, which helps in enhancing model convergence and stability. The `running_model_avg` function calculates the average of the model parameters across all clients during the federated learning process. In our `fed_avgm_experiment`, we train a global model across multiple rounds, with a random subset of clients updating their local models in each round. The global model is subsequently updated by averaging these local models. For our specific experiment, we employ a Convolutional Neural Network (CNN) as the global model and involve 10 clients in each round.

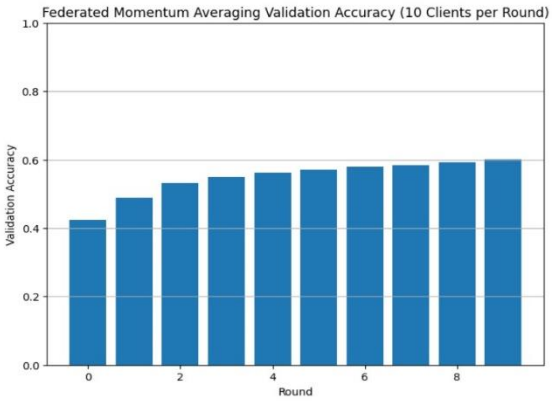
Federated Momentum Averaging (FedAvgM)

Federated Momentum Averaging (FedAvgM) is a federated learning approach that utilizes stochastic gradient descent (SGD) with momentum for local model updates. It calculates the average of model parameters across multiple client devices to update a global model, enhancing convergence and stability in distributed learning scenarios.

Results

In our Federated Momentum Averaging (FedAvgM) experiment, we trained a global model across 10 rounds, with 10 randomly chosen clients participating each time. We observed a consistent improvement in validation accuracy, starting from 0.4239 in the initial round and reaching 0.6006 by the tenth round. This showcases the efficacy of the federated learning method in boosting model performance with distributed client data.





Comparing the performances of the 3 Learning Methods

Initial Performance:

FedAvgM starts with the highest initial validation accuracy of 0.4239, followed by FedAvg at 0.2923 and Federated Personalization Learning at 0.237.

Rate of Improvement:

- FedAvgM shows the fastest improvement rate, reaching 0.6006 in 10 rounds.

- FedAvg shows a consistent improvement but at a slower rate compared to FedAvgM, reaching 0.5644.
- Federated Personalization Learning has the slowest rate of improvement, reaching 0.4888 by the tenth round.

Consistency:

- FedAvg has a steady increase in accuracy across all rounds, demonstrating consistent learning across clients.
- FedAvgM shows a rapid and consistent improvement, indicating effective utilization of momentum in the averaging process.
- Federated Personalization Learning also shows consistent improvement, but at a slower pace compared to the other two methods.

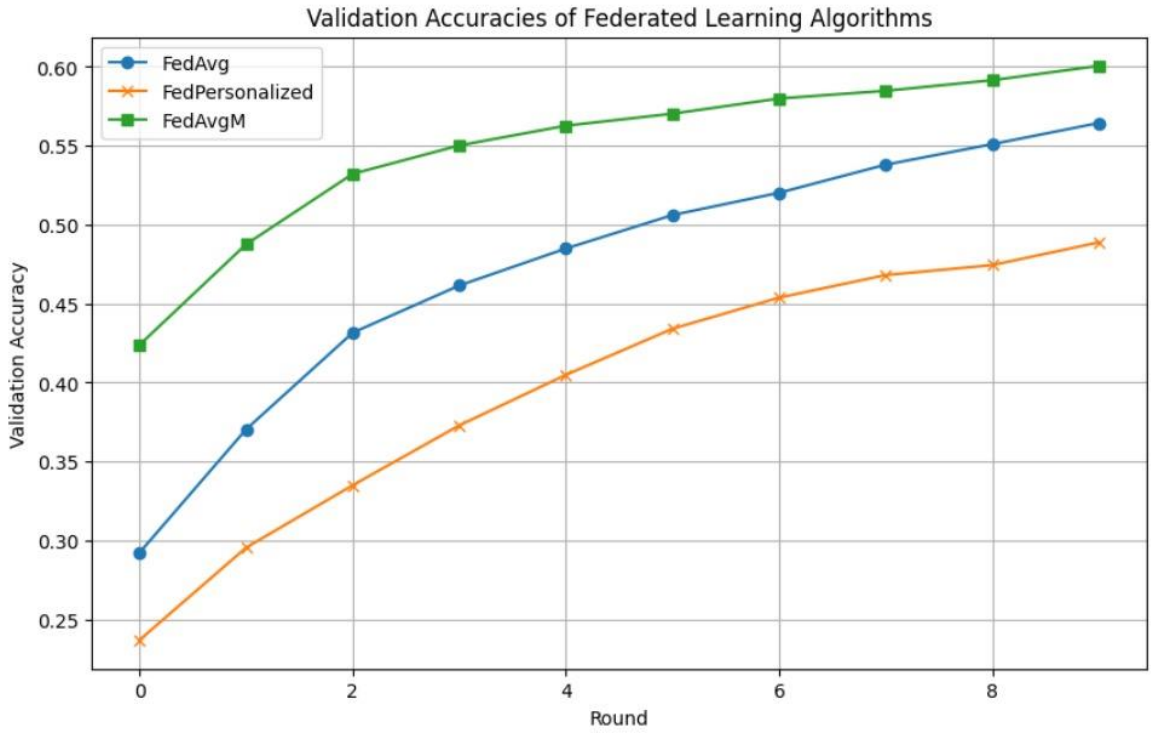
Final Performance:

- FedAvgM achieves the highest validation accuracy of 0.6006.
- FedAvg achieves an accuracy of 0.5644.
- Federated Personalization Learning achieves an accuracy of 0.4888.

Conclusion

- Federated Momentum Averaging (FedAvgM) demonstrates superior performance among the three methods in terms of both initial validation accuracy and rate of improvement, achieving the highest final accuracy of 0.6006.
-
- Federated Averaging (FedAvg) also shows effective performance with consistent improvement but at a slower rate compared to FedAvgM.
-
- Federated Personalization Learning lags behind the other two methods in terms of performance, showing slower improvement and achieving the lowest final accuracy of 0.4888.

Therefore, based on these results, Federated Momentum Averaging (FedAvgM) is the most effective method for this federated learning task, followed by Federated Averaging (FedAvg), while Federated Personalization Learning is the least effective among the three.


Validation Accuracies of Federated Learning Algorithms
FedAvg
FedPersonalized
FedAvgM
Validation Accuracy
Round
0.60
0.55
0.50
0.45
0.40
0.35
0.30
0.25
0
2
4
6
8