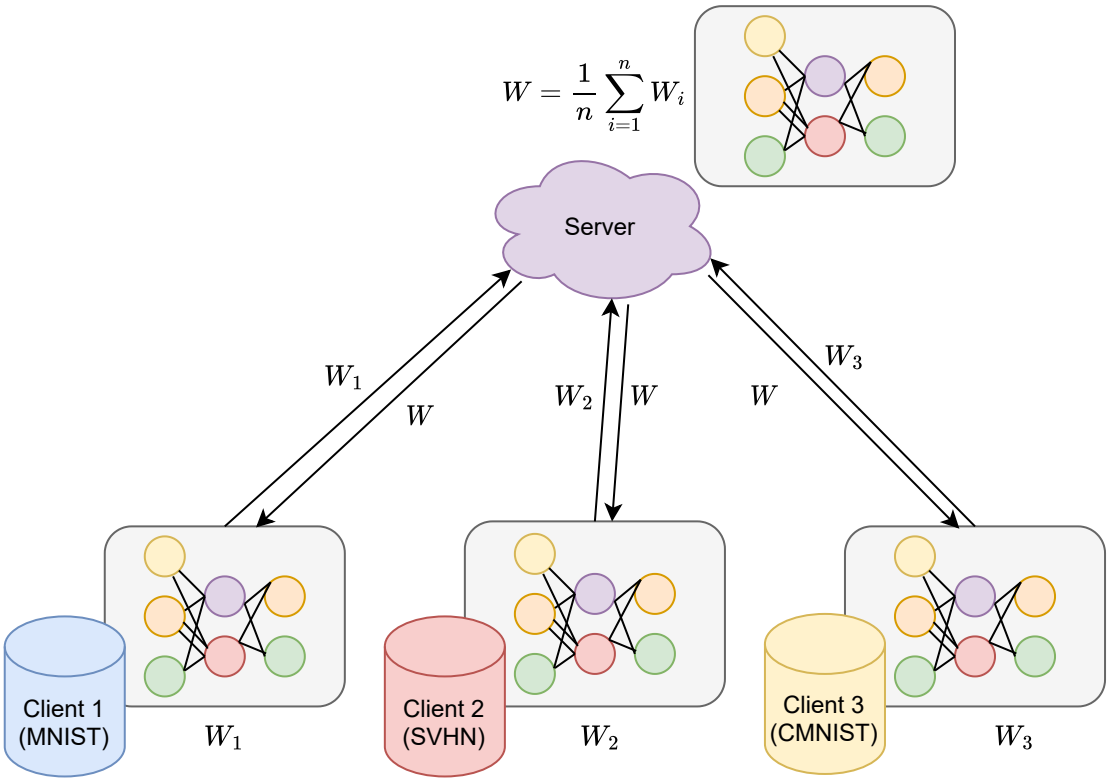
Nirbhay Sharma (B19CSE114)

PA-3 Dependable AI

FedAvg Algorithm



(a) Workflow denoting FedAvg algorithm

The above illustration shows the workflow of FedAvg algorithm. Each client has its own dataset with it. The server first send its model weight W to each clients. The clients replace their models weights with W as $W_i \leftarrow W$. After replacement they train their models on their respective dataset D_i and update the model weights as $W_i = W_i - \eta \triangledown L(X, Y; W_i)$. Finally they send their respective model weights to server. The server on receiving the weights aggregates them as $W = \frac{1}{n}\sum_{i=1}^{n} W_i$. The aggregated weights are then again send to each client for further communication rounds.

Mathematical explanation of the FedAvg function

The function used for aggregation in Fedavg is as follows:

$$W = \frac{1}{n}\sum_{i=1}^{n} W_i$$

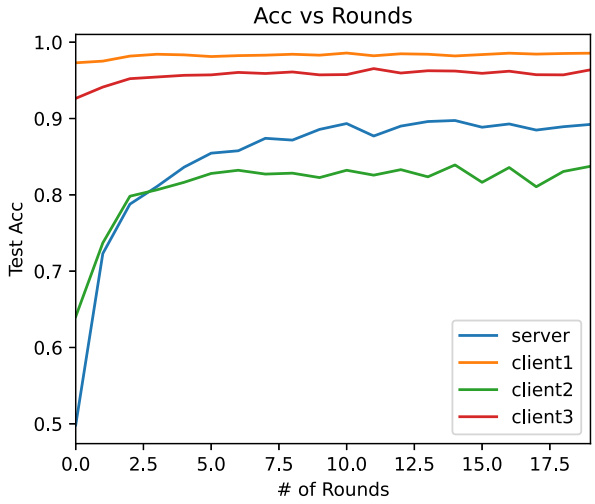
This function is the aggregation of the updated weights and biases matrix of a neural network. Each neural network can be basically characterized by its weight and bias matrix for each operation such as convolution etc. The above mathematical operations combines the knowledge of each network by averaging their weight and bias matrix element wise.

Experiments

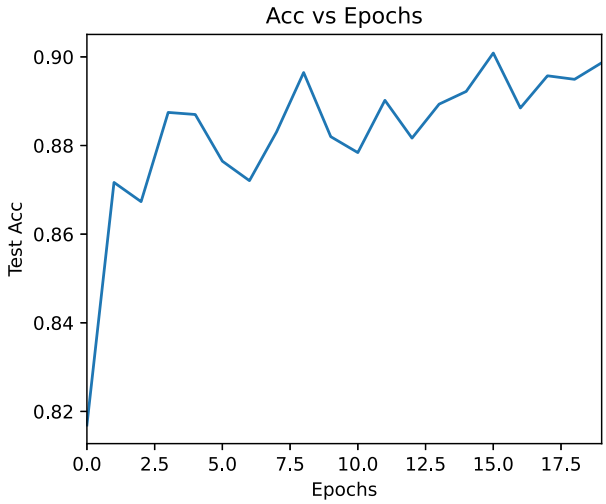
We perform experiments with Resnet18 (abbreviated as R18) model as the global models. We use three different datasets MNIST, SVHN, Coloured-MNIST (abbreviated as CMNIST) at three different clients having 1000 images per class for a 10 class digit classification task. Client1 holds MNIST dataset, client2 holds SVHN dataset, and client3 holds CMNIST dataset. The test data for each client contains 500 images per class and the test data for server model is the concatenation of the individual clients dataset. We also compare the FedAvg method with the baseline method i.e. training with all the three datasets in centralized fashion.

Results

We first report the test accuracy curve for the test dataset at each client and server for their respective datasets. We also report the test accuracy of the centralized training.



(a) FedAvg R18



(b) Non Fed R18

If we see the test accuracy improvement we can observe that in FedAvg case the server and clients accuracy are continuously increasing. In baseline also the test accuracy is increasing as the epochs progress

We report the test accuracies of server and client models for their test dataset. We also report test accuracies of baseline method.

Resnet18 Acc	Server	Client1 (MNIST)	Client2 (SVHN)	Client3 (CMNIST)	Baseline
Class 0	91	100	88	98	90
Class 1	91	99	83	98	93
Class 2	88	99	83	98	89
Class 3	92	99	69	98	89
Class 4	91	99	88	95	88
Class 5	83	99	89	95	88
Class 6	86	98	87	98	90

Resnet18 Acc	Server	Client1 (MNIST)	Client2 (SVHN)	Client3 (CMNIST)	Baseline
Class 7	90	98	91	95	92
Class 8	92	98	76	96	84
Class 9	87	97	84	94	95
Avg Acc	89.2	98.6	83.7	96.4	89.9

In this we can infer that for FedAvg case the accuracy average and class wise accuracy of clients and server seems to be considerable. The aggregated models and the local models have learnt the representations better. For baseline as well it learns better representations and performs well.

We also report the precision, recall, f1score of server, clients and baseline methods

	precision	recall	f1-score
0	0.99	1.00	0.99
1	1.00	0.99	0.99
2	0.98	0.99	0.99
3	0.98	0.99	0.99
4	0.99	0.99	0.99
5	0.99	0.99	0.99
6	0.99	0.98	0.99
7	0.98	0.98	0.98
8	0.98	0.98	0.98
9	0.97	0.97	0.97

(a) R18 C1 (MNIST)

	precision	recall	f1-score
0	0.91	0.88	0.90
1	0.93	0.83	0.88
2	0.90	0.83	0.86
3	0.77	0.69	0.73
4	0.90	0.88	0.89
5	0.73	0.89	0.81
6	0.74	0.87	0.80
7	0.92	0.91	0.91
8	0.82	0.76	0.79
9	0.79	0.84	0.81

(b) R18 C2 (SVHN)

	precision	recall	f1-score
0	0.98	0.98	0.98
1	0.98	0.98	0.98
2	0.96	0.98	0.97
3	0.92	0.98	0.95
4	0.98	0.95	0.97
5	0.99	0.95	0.97
6	0.98	0.98	0.98
7	0.97	0.92	0.95
8	0.96	0.96	0.96
9	0.93	0.95	0.94

(c) R18 C3 (CMNIST)

	precision	recall	f1-score
0	0.95	0.91	0.93
1	0.93	0.91	0.92
2	0.96	0.88	0.91
3	0.72	0.92	0.81
4	0.92	0.91	0.91
5	0.94	0.83	0.88
6	0.95	0.86	0.90
7	0.96	0.90	0.93
8	0.78	0.92	0.85
9	0.92	0.87	0.89

(d) R18 Server

	precision	recall	f1-score
0	0.95	0.90	0.92
1	0.88	0.93	0.90
2	0.96	0.89	0.92
3	0.87	0.89	0.88
4	0.93	0.88	0.91
5	0.93	0.88	0.90
6	0.88	0.90	0.89
7	0.93	0.92	0.93
8	0.90	0.84	0.87
9	0.79	0.95	0.86

(e) R18 Baseline

For client and server the precision recall for each class is also considerably better and each clients and server shows up promising performance. The baseline method also seems to be working well.

Confusion matrix is also presented below

```
[[498   0   0   0   0   0   1   0   0   1]
 [  0 493   3   0   1   0   2   1   0   0]
 [  0   0 495   1   0   0   0   3   1   0]
 [  0   0   3 493   0   3   0   0   0   1]
 [  0   0   0   0 493   0   1   0   0   6]
 [  0   0   0   4   0 494   0   2   0   0]
 [  5   0   0   0   0   1 492   0   2   0]
 [  1   1   1   0   3   0   0 491   1   2]
 [  1   0   0   2   0   0   0   2 492   3]
 [  0   0   1   1   2   2   0   2   5 487]]
```

(a) R18 C1 (MNIST)

```
[[440   3   4   2   0   3  23   1   9  15]
 [  7 415   6   9  17   9  10  15   6   6]
 [  0   4 414  26  10  13   4  11   6  12]
 [  2   5   4 344   4  72  10   0  24  35]
 [  1   3   3   9 440   4  12   3  12  13]
 [  4   1   1  13   3 446  26   1   2   3]
 [  5   3   0   6   4  29 436   0  13   4]
 [  1   8  13  10   2   4   0 454   0   8]
 [  1   5   3  18   3  15  59   2 378  16]
 [ 21   0  12   7   6  12   6   7   9 420]]
```

(b) R18 C2 (SVHN)

```
[[492   2   1   0   0   0   2   0   2   1]
 [  0 488   1   1   4   0   5   0   1   0]
 [  1   2 490   0   0   0   0   2   4   1]
 [  0   0   3 490   0   0   0   3   3   1]
 [  0   0   1   0 477   0   1   2   1  18]
 [  1   0   0  17   0 474   1   2   4   1]
 [  4   0   0   0   1   2 491   0   2   0]
 [  0   3  12   8   3   1   0 462   1  10]
 [  3   2   1   6   1   0   1   1 479   6]
 [  0   3   1   8   2   2   1   3   4 476]]
```

(c) R18 C3 (CMNIST)

```
[[1371   23    1   34    9    2   14    2   33   11]
 [   5 1370    2   32   38    4   11   11   25    2]
 [   5   10 1316   78   14    0    2   16   55    4]
 [   3    5    5 1382    8   29    4    4   48   12]
 [   1    6    6   26 1371    1    7    8   38   36]
 [   5    4    1  195    6 1243   15    2   22    7]
 [  22    5    0   32   17   26 1290    0  105    3]
 [   3   33   31   36    8    5    0 1346   12   26]
 [   8    6    3   52    7    4   12    5 1386   17]
 [  24    8   12   57   20    9    4    6   51 1309]]
```

(d) R18 Server

```
[[1350   21    5   13    3    2   34    2   33   37]
 [   2 1393    3   14   25    4    3   26    9   21]
 [   6   24 1335   30    8   13    6   28   14   36]
 [   5   20    8 1336    0   41    9    3   19   59]
 [   6   41    7    8 1321    6   13   18    8   72]
 [   7   12    2   58    5 1320   53    4   10   29]
 [  22   15    1   15   28   11 1357    0   43    8]
 [   0   44   20   11    8    6    1 1382    4   24]
 [   9   15   10   37   10    9   51    6 1265   88]
 [  19    4    4    8    5   10    9   13    7 1421]]
```

(e) R18 Baseline

For client and server the confusion matrix is also considerably better and each clients and server shows up promising performance. The baseline method also seems to be working well. Some misclassification in each case has happended but that is very less in number as compared to the True positive predictions which leads to the class wise accuracy good enough.

Comparison of FedAvg with Baseline

As represented in the table There is not a very high difference in FedAvg setup and centralized setup. Please note that in FedAvg setup we don't even send the client's data to the server. However, If we compare closely The centralized accuracy is little bit higher than the FedAvg accuracy. This is due to the fact that a good amount of data is availabe at one place. The difference in Baseline and FedAvg setup will start to show up when data is a bit more non-IID distributed. Here all the clients have around 10000 images which is enough to learn a 10 class classification problem. However if the data is more non-IID distributed then the baseline accuracy will still be the same but the FedAvg accuracy is likely to reduce a little bit. However, for this case both methods are working nicely.