**Task 2 a)**

We ran a simulation for federated learning with 3 clients, we used a simple CNN implementation consisting of 4 convolution layers and one full connected layer.

We created a load dataset function to load the ChestMnist dataset in a way to create a list of *trainloaders* and *valloaders* with *length = number of clients*, the function also returns the testloader which is used at the server-side to evaluate the training

The metrics were aggregated using a simple weighted average function.

The evaluation was done at the client side with a small portion of the validation dataset (referred to as distributed in the table) and on the server-side with the test dataset (referred to as centralized in the table)

We tried this setup and documented the results while varying the batch size and learning rate as follows

* Batch size = 32 and learning rate = 1e-3
* Batch size = 8 and learning rate = 1e-3
* Batch size = 64 and learning rate = 1e-3
* Batch size = 32 and learning rate = 1e-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Round | Batch\_size=32  L\_R = 1e-3 | Batch\_size=8  L\_R = 1e-3 | Batch\_size=64  L\_R = 1e-3 | Batch\_size=32  L\_R = 1e-1 |
| Loss ↓ Distributed | 1 | 0.052 | 0.211 | **0.0261** | 0.0504 |
| 2 | 0.048 | 0.202 | **0.0243** | 0.0504 |
| 3 | 0.047 | 0.195 | **0.0241** | 0.0506 |
| Loss ↓ Centralized | *0* | *0.060* | *0.241* | *0.030* | *0.061* |
| 1 | 0.054 | 0.219 | **0.026** | 0.0524 |
| 2 | 0.050 | 0.210 | **0.025** | 0.0524 |
| 3 | 0.049 | 0.202 | **0.024** | 0.0525 |
| Accuracy ↑ Distributed | 1 | 12.59 | 12.594 | **12.6** | 12.59 |
| 2 | 13.96 | 12.594 | **20.3** | 12.59 |
| 3 | 24.28 | 15.874 | **24.8** | 12.59 |
| Accuracy ↑ Centralized | *0* | *3.25* | *12.035* | *0.51* | *2.23* |
| 1 | 12.03 | 12.035 | **12.04** | 12.035 |
| 2 | 13.40 | 12.035 | **19.74** | 12.035 |
| 3 | 23.81 | 15.517 | **24.54** | 12.035 |

Results, as expected, are better with larger batch size, the model is capable of converging faster, Comparing batch size of 64 batch size of 8, the accuracy with the first one was capable to jump from 12% to 19.8% in one round, while with batch size 8 accuracy remained 12% for 2 rounds.

The same is true for the learning rate with a high learning rate (last column); the model didn’t converge at all, and as shown, loss and accuracy are not changing round after the other.

**Task 2 b)**

In our federated learning study, we experimented with small image changes (single pixel perturbations) and intentionally incorrect labels (label poisoning) on one client's data, referred to as Client 1.

We trained the model using batches = 32 and a learning rate of 0.001, without altering the model itself throughout the process.

**Labels Poisoning**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Round | Labels Poisoning | Labels Poisoning | Labels Poisoning |
| Rate = 0 | Rate = 0.5 | Rate = 1 |
|  |  |  |
| Loss ¯ Distributed | 1 | 0.022 | 0.022 | 0.0213 |
| 2 | 0.019 | 0.0197 | 0.0197 |
| 3 | 0.019 | 0.0194 | 0.0193 |
| Loss ¯ Centralized | 1 | 0.0215 | 0.0214 | 0.0213 |
| 2 | 0.0195 | 0.0199 | 0.0198 |
| 3 | 0.0193 | 0.0199 | 0.0194 |
| Accuracy ­ ↑ Distributed | 1 | 58.222 | 54.327 | 55.060 |
| 2 | 67.064 | 65.692 | 65.638 |
| 3 | 66.975 | 66.645 | 66.628 |
| Accuracy ­ ↑ Centralized | 1 | 57.803 | 53.372 | 53.884 |
| 2 | 66.460 | 64.788 | 64.891 |
| 3 | 66.656 | 65.760 | 66.232 |

In both distributed and centralized scenarios, as the label poisoning rate increases from 0 to 0.5, there is generally an increase in loss and a decrease in accuracy, suggesting that the poisoning negatively impacts the our model's performance.

However, when the poisoning rate is at its maximum 1, the impact is not as straightforward. The loss is sometimes lower than at a poisoning rate of 0.5, and the accuracy is higher in some cases. This could be due to overfitting on the poisoned labels or other factors specific to the training algorithm or data.

Across rounds, you can see the training process is likely converging as the loss decreases and accuracy increases in a typical learning scenario.

**Single Pixel Perturbations (SPP)**

Our images are 28x28 in size. We changed the pixels in x=10 and y=10 coordinates.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Round | SPP | SPP | SPP |
| Rate = 0 | Rate = 0.5 | Rate = 1 |
|  | X= 10  Y =10 | X= 10  Y =10 |
| Loss ¯ Distributed | 1 | 0.022 | 0.0216 | 0.0214 |
| 2 | 0.019 | 0.0197 | 0.0196 |
| 3 | 0.019 | 0.0194 | 0.0193 |
| Loss ¯ Centralized | 1 | 0.0215 | 0.0214 | 0.0214 |
| 2 | 0.0195 | 0.0199 | 0.0196 |
| 3 | 0.0193 | 0.0195 | 0.0194 |
| Accuracy ­ Distributed | 1 | 58.222 | 54.327 | 56.422 |
| 2 | 67.064 | 65.692 | 66.209 |
| 3 | 66.975 | 66.645 | 66.816 |
| Accuracy ­ Centralized | 1 | 57.803 | 53.372 | 55.984 |
| 2 | 66.460 | 64.788 | 65.577 |
| 3 | 66.656 | 65.760 | 66.433 |

We can infer that the model is somewhat sensitive to single pixel perturbations, but it still retains a degree of learning capability even with the highest perturbation rate.

It's also notable that the distributed learning system consistently shows a slight advantage in accuracy over the centralized system under perturbed conditions.