COMP4805: Project

Final Report

Federated Learning on non-IID data

Wong Ngai Sum

UID: 3035380875

Supervisor: Dr. Heming Cui

Abstract

Federated Learning, a new decentralized collaborative Machine Learning framework presented by Google in 2017, enables end devices to collaboratively learn a shared Machine Learning model, while reducing privacy and security risks, attaining lower latency and providing personalized models to clients, but it suffers from performance issue with non-IID data distribution. This project is a comprehensive study of the phenomenon and related work. The results show that precision loses during averaging step due to the property of averaging and serious parameter divergence. Different training settings are tested to evaluate the effects on performance.

Contents

1 Introduction **5**

1.1 Background 5

1.2 Federated Learning 6

1.3 Federated Averaging 6

1.4 Non-IID Data 7

1.5 Review of Literature 8

1.5.1 Federated Learning with Non-IID Data 8

1.6 Scope and Deliverables 8

1.6.1 Scope 8

1.6.2 Deliverables 9

2 Methodology **11**

2.1 Main phases 11

2.1.1 Evaluations 11

2.1.2 Explanations 11

2.1.3 Suggestions 11

2.2 Simulation of Federated Learning 13

2.3 Datasets 13

2.4 Data distributions 13

2.5 Default training settings 14

3 Experiments and Results **15**

3.1 Traditional Machine Learning vs Federated Learning 15

3.2 Performance analysis of non-IID data 16

3.3 Predicted and target values under highly non-IID setting 20

3.4 Parameters divergence on non-IID setting 22

3.5 Reasons for poor performance 23

3.6 Effects of batch size 24

3.7 Effects of number of epochs 26

3.8 Full gradient descent using Federated Averaging 28

3.9 Effects of number of devices 29

3.10 Effects of fraction of devices selected for training 31

3.11 Effects of learning rate 33

3.12 Effects of optimizer 35

3.13 Performance on CIFAR-10 37

3.14 Effects of CNN architecture 38

3.15 Data sharing technique 39

3.16 Can transfer learning improve performance? 40

4 Conclusion **41**

5 Future works **42**

6 References **43**

1 Introduction

1.1 Background

As technology advances, processor chips become increasingly powerful and mobile processors are as fast as most desktop computers. Smartphones and IoT devices are everywhere and a large amount of data are being collected through sensors for analytics. They generate enormous valuable data and Machine Learning is applied to those data to improve the intelligence of services. Standard Machine Learning requires operations in data centers with a centralized dataset to attain fast processing and low latency for transmission. However, it comes with several constraints such as data privacy and the inability to process large datasets, as data are generated in end devices in most applications. Serverless architecture and distributed architecture have become ways for many developers to have more flexibility and improve their services. Federated Learning, a new decentralized collaborative Machine Learning framework presented by Google in 2017 [4], enables end devices to collaboratively learn a shared Machine Learning model, while reducing privacy and security risks, attaining lower latency and providing personalized models to clients. Federated Learning has been tested in Gboard on Android, which stores information about the current context to train its query suggestion model [2].

Federated Learning has several key properties that differentiate from typical distributed edge learning, including non-IID data, varying amounts of training data between clients, massive distribution of data and limited internet connection. Specifically, it has issues with serious communication overhead [3][4][5], low accuracy on non-IID data [5] and privacy attacks [6].

Some research has proposed optimization strategies to address these problems, such as subsampling, probabilistic quantization [5] and globally shared data [6]. However, there is a large room for improvement to make Federated Learning more efficient [8].

This project mainly focuses on issues with non-IID data. Experiments will be carried out and explanations of these phenomena will be given.

1.2 Federated Learning

Federated Learning can be divided into 4 steps [3][4]: i) server iteratively asks random clients to download parameters of a trainable model; ii) a client updates the model with their local data; iii) upload the new model parameters to the server while keeping all the training data on device; iv) server aggregates the updates to further improve the model. Due to the slow and unstable network of mobile devices, the Federated Averaging algorithm is typically used to train neural networks using 10-100x less communication than federated SGD (Stochastic Gradient Descent) [2]. However, the FedAvg algorithm reduces accuracy significantly with highly skewed non-IID data, up to 11% for MNIST, 51% for CIFAR-10 and 55% for keyword spotting datasets [6]. Improving accuracy with non-IID is still the main challenge for many researchers.

1.3 Federated Averaging

Most applications of deep learning have exclusively relied on variants of stochastic gradient descent for optimization [4]. It can be applied naively to the federated optimization problem, where a single batch gradient is calculated per communication round. However, we can add more computation to each client by more local updates before the averaging step. This approach is termed Federated Averaging. The algorithm consists of t rounds, where for each round t, faction C of K clients are randomly selected for training, resulting in a subset St of $max(C \cdot K,\ 1)$ clients. For each client $k \in St$, it updates the current model w_t^k by its local data and sends the new model w_{t+1}^k to server. The server then aggregates the updates $\sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$ to get the new model w_{t+1} .

1.4 Non-IID Data

IID data means independent and identically distributed data. Independent means the sample items are not connected in any way. Identically distributed means all sample items are taken from the same probability distribution. Some previous studies on Federated Learning treated data instances as independently and identically distributed [13]. This neglects the relation among the instances convey important structure information [14]. The instances on a device are rarely independent, and better performance can be achieved if data are treated in a non-IID way. Non-IID data is very common, such as search queries entered by a guy and his forum posts because they may be related to his background and the wordings are similar. If he is a mathematician, most of his data may be related to mathematics. Non-IID data noise is a common challenge in many disciplines including neuroscience [12]. It may cause biased estimation in Federated Learning due to weight divergence [6].

1.5 Review of Literature

1.5.1 Federated Learning with Non-IID Data

One paper has been discussed here related to this project, “Federated Learning with Non-IID Data” [6]. This paper shows that the accuracy reduction is less for the two-class non-IID data than for one-class non-IID data, so the accuracy of the Federated Averaging algorithm will be affected by the data distribution. The accuracy reduction can also be explained by the weight divergence, which can be evaluated by the earth mover’s distance between the distributions. To tackle the problem of weight divergence, globally shared data are used to train a warm-up CNN model to validation accuracy of ~60%, then shared data and the trained model are distributed to each client. As a result, accuracy can be increased by ~30% for the CIFAR-10 dataset with only 5% global shared data.

However, this paper only tested the data-sharing technique in one use case and assume the amount of data distributed to each client is equal. Also, it does not explain why a warm-up model should be trained on shared data and why test accuracy should be trained to ~60%. More investigations can be made to evaluate the effectiveness of this strategy.

1.6 Scope and Deliverables

1.6.1 Scope

Trade-off analysis. There are lots of learning parameters that may affect performance. It is crucial to figure out the relationships between parameters and performance. For example, there are two intuitive ways to increase computation: i) increased parallelism and ii) increased computation on clients [4]. It is interesting to test how increasing and decreasing computation will affect performance with non-IID data, and what will be sacrificed. Moreover, several parameters need to be fine-tuned in different use cases [6], so different values of learning parameters will be tested.

Accuracy analysis. There are other Machine Learning options, such as distributed training in data centers. There are lots of combinations of data distributions, such as IID distribution, one class per device, two classes per device and so on. Also, data distribution may be unbalanced, which means some clients may have more or fewer data than others in real life. The recent paper proposed using globally shared IID data and warm-up model to improve accuracy by ~30% with only 5% shared data [6]. However, it may not reflect real-world scenarios as amounts of data on each device are assumed identical and different types of learning tasks have not been tested. Therefore, several experiments with different training settings are done to analyze the performance.

Reasons for poor accuracy with non-IID data. Federated Learning is suitable for IID data instead of non-IID data [4]. The primary goal of this project is to explain why Federated Learning with non-IID data has poor accuracy. It can be done by analyzing test results. Reasons will be proposed to explain the phenomena.

1.6.2 Deliverables

This project is a comprehensive study of the performance issue and related work. It will deliver: i) a trade-off analysis for different training settings like the fraction of devices selected for training [6]; ii) an analysis of test accuracy with Federated Learning on different data distributions as well as results with other training settings; iii) reasons to explain why Federated Learning on non-IID data has poor accuracy; iv) results of optimization trials.

2 Methodology

To improve the reliability and validity of this project, three main phases are divided and the methodology of each phase are as follows:

2.1 Main phases

2.1.1 Evaluations

Some controlled experiments similar to real-life use cases will be conducted, such as image classification problems. I may try to present datasets like MNIST [9], VGG FACE[10] and CIFAR-10 [11]. Different values for parameters will be experimented to do trade-off analysis. Evaluations on different architectures will be done and different data distributions will be tested. The amounts of data on each client may be different or the same.

2.1.2 Explanations

Experimental results will be presented and analyzed. The description of tests and the processes to gather and analyze data using quantitative methods will be elaborated. As a large amount of information will be presented, the information will be organized sub-sections.

2.1.3 Suggestions

I will read more papers to see what techniques were proposed to improve performance on non-IID data. After doing the literature review, I can know what obstacles they are facing to make Federated Learning perform better. I may try several approaches to improve Federated

Digit	MNIST	
	Train	Test
0	5923	980
1	6742	1135
2	5958	1032
3	6163	1010
4	5842	982
5	5421	892
6	5918	958
7	6265	1028
8	5151	974
9	5949	1009
Total	60000	10000

Figure 1: Distribution of digits in MNIST

Learning on non-IID data. In case these suggestions fail, analysis can be done to explain the reasons behind it.

2.2 Simulation of Federated Learning

Since the focus of this project is on performance issue, network speed will not be considered, which means actual convergence time and number of bits to send will not be evaluated. There are several open source libraries of Machine Learning on decentralized data, such as Tensorflow Federated [15], PySyft [16] and PyTorch [17]. PyTorch will be used because of its gentle learning curve.

2.3 Datasets

The image classification problem is adopted in this project to test the performance of Federated Learning on non-IID data. Two datasets are used, namely MNIST and CIFAR-10. MNIST consists of black and white handwriting digits, including 60000 training images and 10000 testing images. CIFAR-10 consists of color images, including 50000 training images and 10000 testing images. Both of them have 10 classes. In CIFAR-10, there are same amount of data among different classes. But in MNIST, the amount of data in each class varies, as can be seen in figure 1. Default train and test split are used for simplicity.

2.4 Data distributions

Several types of data distribution will be used for experiments, including IID, $x\%$ non-IID and y-classes of data per client. $x\%$ non-IID is used to measure the degree of non-IID, $x\%$ non-IID means $x\%$ of data are sorted by labels before distribution and remaining data are randomly distributed. In $x\%$ non-IID setting, data distribution is unbalanced, whereas in y-classes of data per client setting, data distribution is balanced.

2.5 Default training settings

There are lots of training parameters. In this project, mini-batch size $B = 64$, number of local epochs on device $E = 10$, number of communication rounds $R = 50$, dataset = MNIST, optimizer = vanilla stochastic gradient descent, learning rate $\alpha = 0.01$, loss function = cross-entropy loss, number of clients $K = 100$, the fraction of clients selected for training $C = 0.1$, CNN architecture = 5-layer CNN (similar to the traditional LeNet-5), Federated Learning algorithm = Federated Averaging, non-IID data distributions including 1, 2, 5 classes of data per client will be tested. Random seed is manually set so that results are reproducible. These are used as default settings unless explicitly specified. The accuracy in this report is Top-1 validation accuracy.

3 Experiments and Results

3.1 Traditional Machine Learning vs Federated Learning

First, we examine performance on MNIST under the IID setting, using traditional Machine Learning and Federated Learning. As shown in Figure 2 after 50 communication rounds, tradition Machine Learning gains 99.05% validation accuracy after 50 epochs and Federated Learning gains 97.56% after 50 rounds. It reveals that in Federated Learning convergence speed is slower than in traditional Machine Learning, though they get similar accuracies after several rounds. In Federated Learning, at first, its accuracy keeps increasing, then underfitting occurs, so the learning speed becomes slower. Federated Learning with IID data is impossible in most situations so this is only for comparison studies.

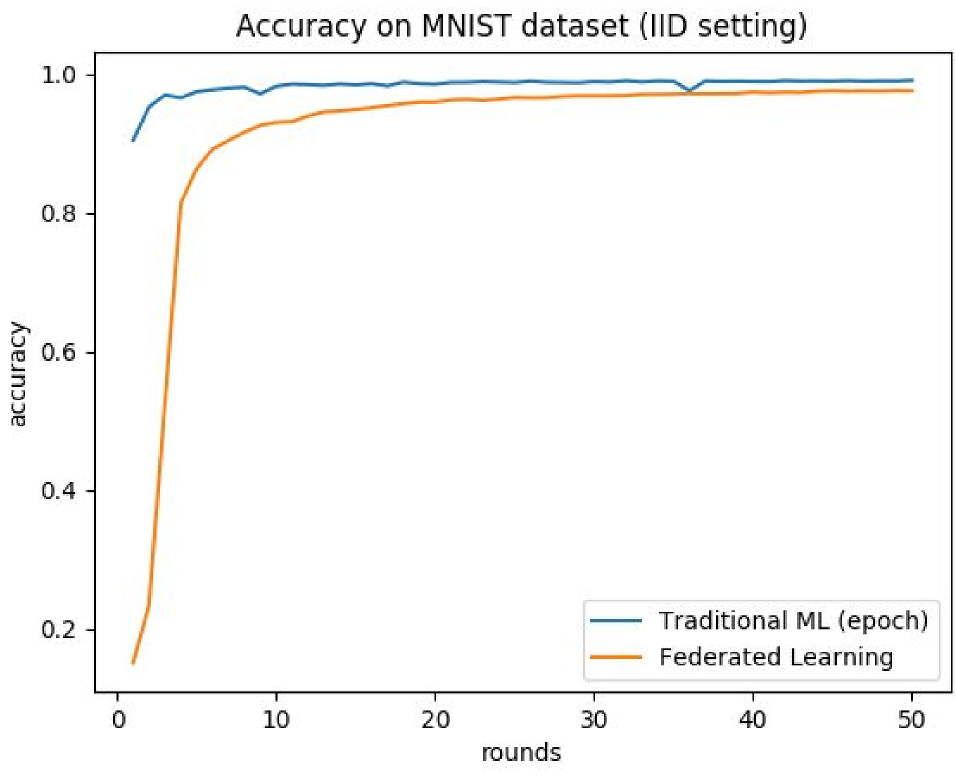


Figure 2: Accuracy on MNIST under IID setting

3.2 Performance analysis of non-IID data

Then we examine performance on MNIST under the non-IID setting. First, we test in $x\%$ non-IID setting. As shown in Figure 3 after 50 communication, if data distribution is 100% non-IID, validation accuracy can arrive 34.62%; if data distribution is 80% non-IID, validation accuracy can arrive 79.85%; if data distribution is 60% non-IID, validation accuracy can arrive 89.01%; if data distribution is 40% non-IID, validation accuracy can arrive 96.02%; if data distribution is 20% non-IID, validation accuracy can arrive 97.5%. It reflects that if the data distribution is slightly non-IID, we can still get performance next to IID. if data distribution is seriously non-IID, we may get poor performance. It loses 62.94% validation accuracy in the 100% non-IID setting but is still better than random guessing. But balanced data distribution may be uncommon.

In real-life situations, data distribution is unbalanced and data distribution in MNIST is a bit unbalanced as shown in figure 1. In this case, the amount of data of a class on a device is the same as the total amount of data of a class divided by the number of devices share that class, which means every class has the same probability to be assigned to each client. As shown in Figure 4 after 50 communication, if data distribution is 1 class per device, validation accuracy can arrive 19.69%; if data distribution is 2 classes per device, validation accuracy can arrive 67.61%; if data distribution is 3 classes per device, validation accuracy can arrive 93.41%; if data distribution is 5 classes per device, validation accuracy can arrive 93.21%; if data distribution is 10 classes per device, validation accuracy can arrive 97.49%. When there is 1 class of data on each device and data distribution is unbalanced, it loses 14.93% validation accuracy. When there are 10 classes of data on each device and data distribution is unbalanced, it loses 0.07% validation accuracy. It reveals that unbalanced data distribution can harm performance seriously under the non-IID setting. So it is expected that Federated Learning will perform extremely bad using Federated Averaging, since data distribution is generally highly skewed and non-IID. As shown in figure 6 and 7, a model can finally converge to high accuracy with slightly non-IID data distribution. It is just a matter of time.

Figure 5 measures inter-rater agreement using the kappa statistic. Kappa is normalized at the baseline of random chance. When there is one class of data per device, no matter data distribution is balanced or unbalanced, performance is lagging behind others. It shows that it is likely to be some level of agreement among devices when they do not know answers but are merely guessing. According to Cohen's hypothesis [1], a certain number of the guesses would be congruent, and that reliability statistics should account for that random agreement. Therefore, the predicted values of the model are collected for evaluation.

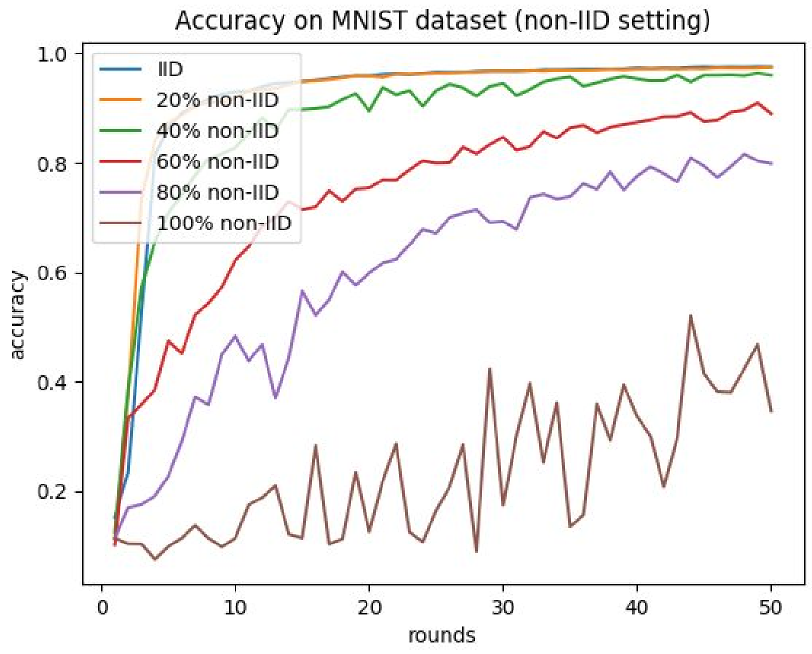


Figure 3: Accuracy on MNIST under the non-IID setting using x% non-IID

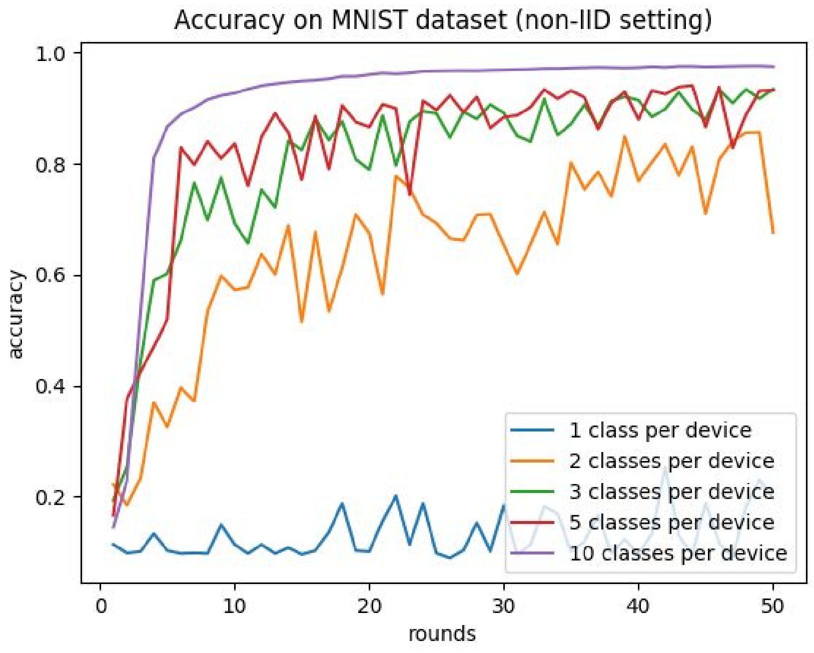


Figure 4: Accuracy on MNIST under the non-IID setting using y classes per device

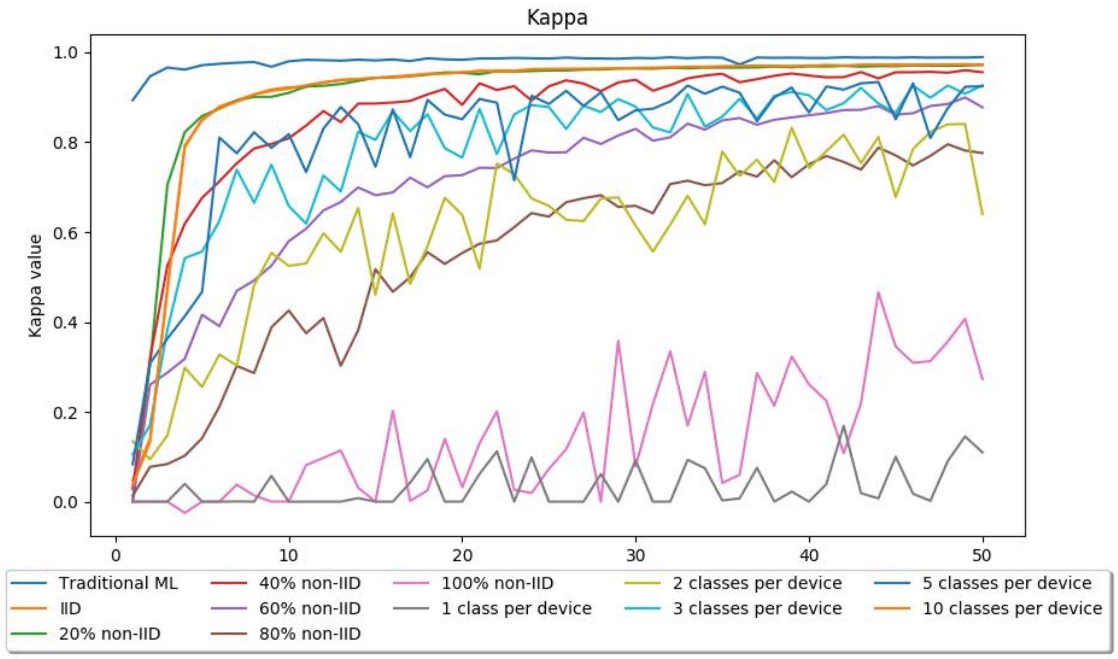
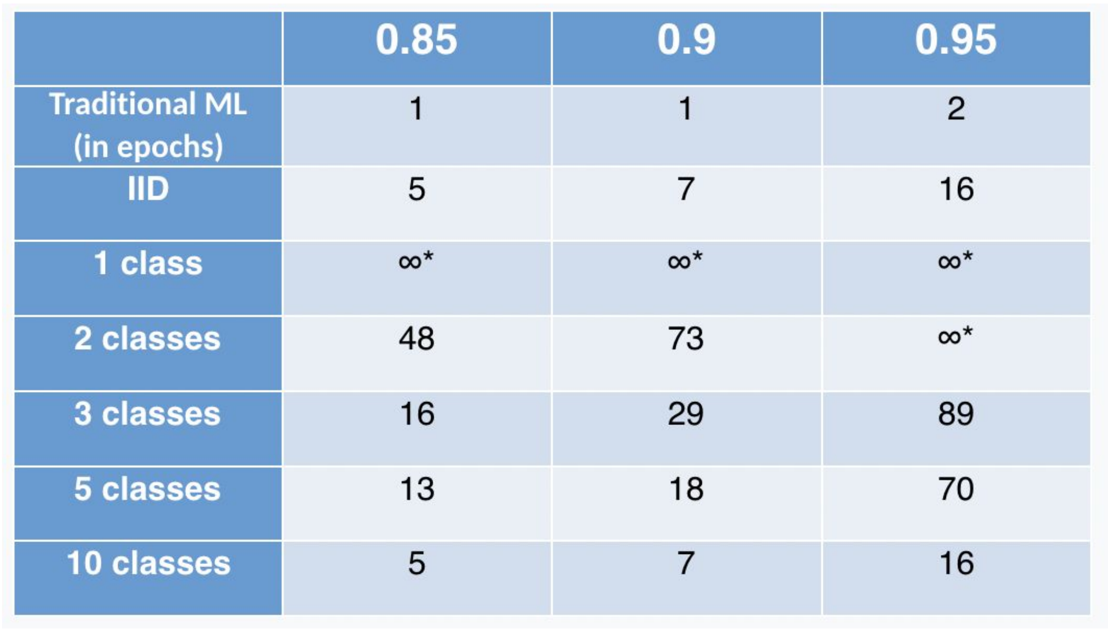
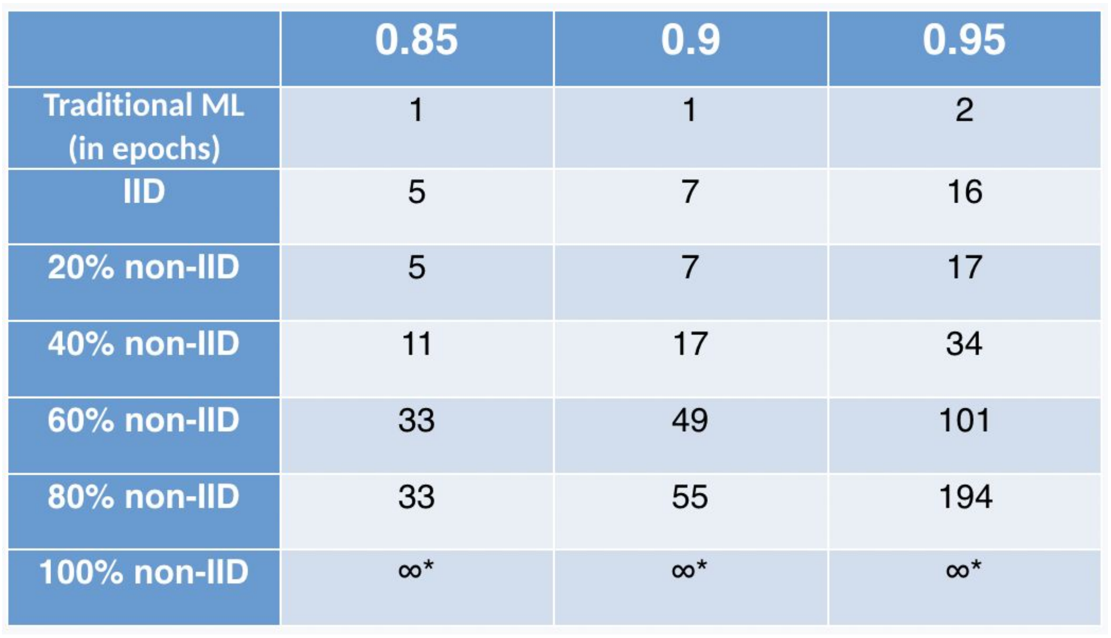


Figure 5: Kappa statistic on MNIST with different data distributions



	0.85	0.9	0.95
Traditional ML (in epochs)	1	1	2
IID	5	7	16
1 class	∞*	∞*	∞*
2 classes	48	73	∞*
3 classes	16	29	89
5 classes	13	18	70
10 classes	5	7	16

Figure 6: Number of rounds to achieve certain accuracies



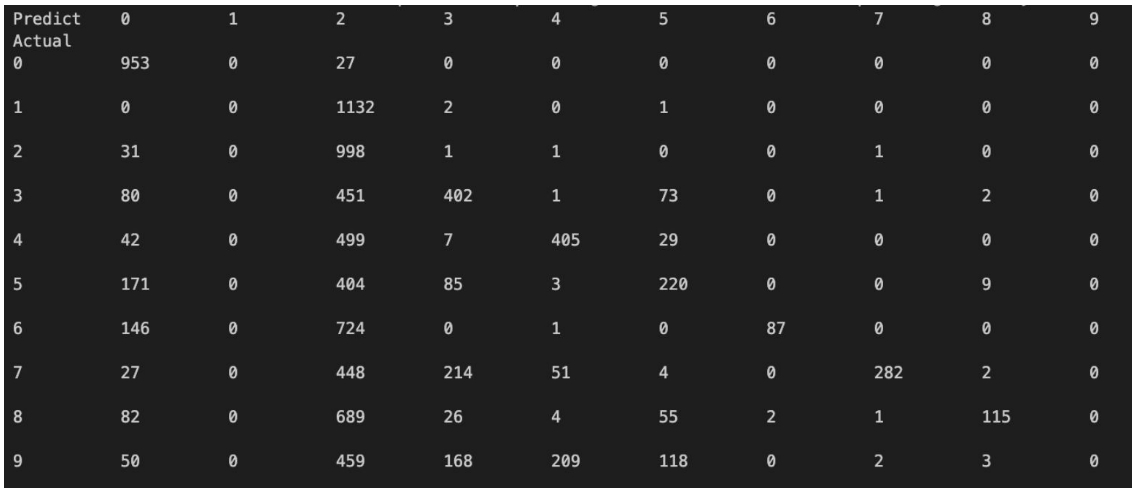
	0.85	0.9	0.95
Traditional ML (in epochs)	1	1	2
IID	5	7	16
20% non-IID	5	7	17
40% non-IID	11	17	34
60% non-IID	33	49	101
80% non-IID	33	55	194
100% non-IID	∞*	∞*	∞*

Figure 7: Number of rounds to achieve certain accuracies

3.3 Predicted and target values under highly non-IID setting

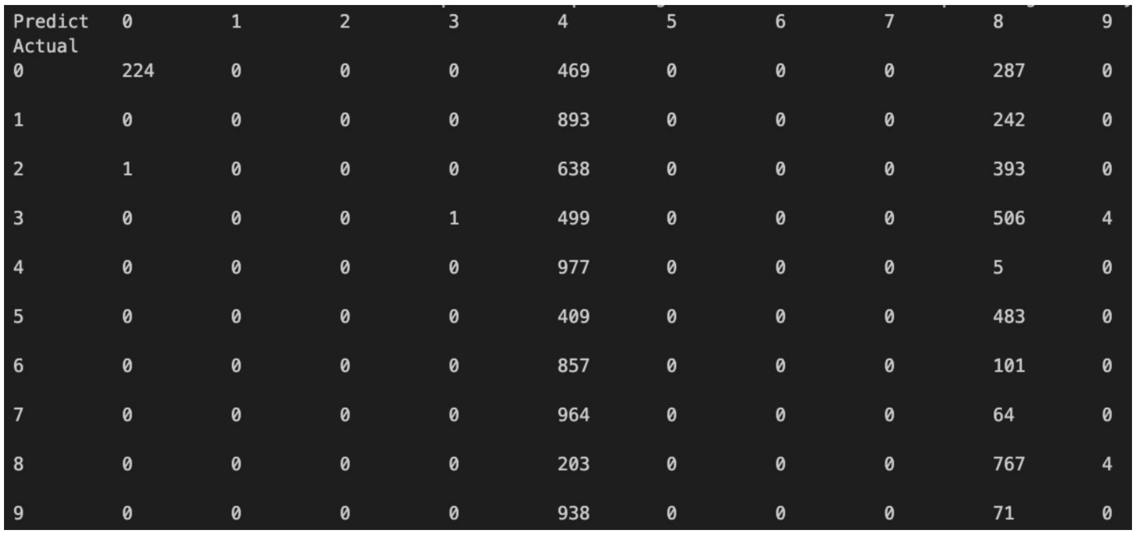
As shown in figure 8 in round 50 under the 100% non-IID setting, most of the predicted values are 0 and 2, no predictions are made on 1 and 9. This matches Cohen's hypothesis that a certain number of guesses would be congruent. All classes of data are distributed to devices every communication round but the model cannot identify some classes, including false positives. It reveals that precision loses during the averaging step. Table 1 shows the area under the ROC curve. Most of them are larger than 0.5, which means it is better than random guessing.

As shown in figure 9 in round 50 under the 1 class per device setting, 97.74% of the predicted values are 4 and 8, no predictions are made on 1, 2, 5, 6 and 7. This matches Cohen's hypothesis again. It shows with unbalanced data distribution, the performance of the trained model is worse. It also reveals that precision loses during the averaging step. This may be explained by the models on some devices are not well-trained, the models are close to the old model, thus averaging models makes the situation worse and convergence speed is substantially slower, especially under the highly non-IID setting. Table 2 shows the area under the ROC curve. Only three of them are slightly larger than 0.5, which means it is slightly better than random guessing.



```
Predict   0     1     2     3     4     5     6     7     8     9
Actual
0         953   0     27    0     0     0     0     0     0     0

1         0     0     1132  2     0     1     0     0     0     0

2         31    0     998   1     1     0     0     1     0     0

3         80    0     451   402   1     73    0     1     2     0

4         42    0     499   7     405   29    0     0     0     0

5         171   0     404   85    3     220   0     0     9     0

6         146   0     724   0     1     0     87    0     0     0

7         27    0     448   214   51    4     0     282   2     0

8         82    0     689   26    4     55    2     1     115   0

9         50    0     459   168   209   118   0     2     3     0
```

Figure 8: Predicted and target values in round 50 under the 100% non-IID setting



```
Predict   0     1     2     3     4     5     6     7     8     9
Actual
0         224   0     0     0     469   0     0     0     287   0

1         0     0     0     0     893   0     0     0     242   0

2         1     0     0     0     638   0     0     0     393   0

3         0     0     0     1     499   0     0     0     506   4

4         0     0     0     0     977   0     0     0     5     0

5         0     0     0     0     409   0     0     0     483   0

6         0     0     0     0     857   0     0     0     101   0

7         0     0     0     0     964   0     0     0     64    0

8         0     0     0     0     203   0     0     0     767   4

9         0     0     0     0     938   0     0     0     71    0
```

Figure 9: Predicted and target values in round 50 under the 1 class per device setting

Class	0	1	2	3	4	5	6	7	8	9
Area	0.951	0.5	0.714	0.671	0.691	0.608	0.545	0.637	0.558	0.5

Table 1: Area under the ROC curve in round 50 under the 100% non-IID setting

Class	0	1	2	3	4	5	6	7	8	9
Area	0.614	0.5	0.5	0.5	0.672	0.5	0.5	0.5	0.775	0.5

Table 2: Area under the ROC curve in round 50 under the 1 class per device setting

3.4 Parameters divergence on non-IID setting

Parameters divergence in this project is calculated by the mean absolute percentage error. It may not be a robust choice but it does provide some insights. As shown in figure 10 and 11 divergence is bigger with non-IID data distribution. It shows that divergences under the non-IID settings are generally larger. Model parameters can be positive or negative and small changes in model parameters may result in a large change in the output. When data distribution is highly non-IID, devices train for local optima rather than global optima. Large parameters divergence means the model they train is far away from the target model, so performance may be poor.

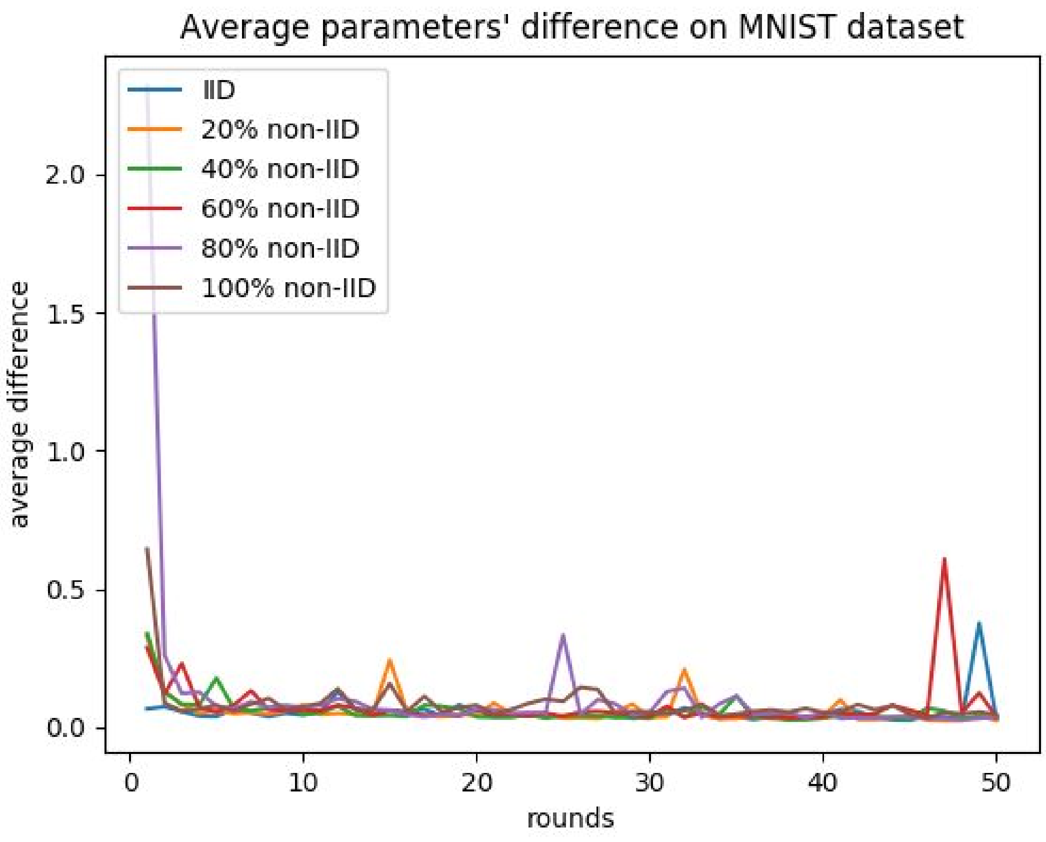


Figure 10: Parameters divergence on MNIST

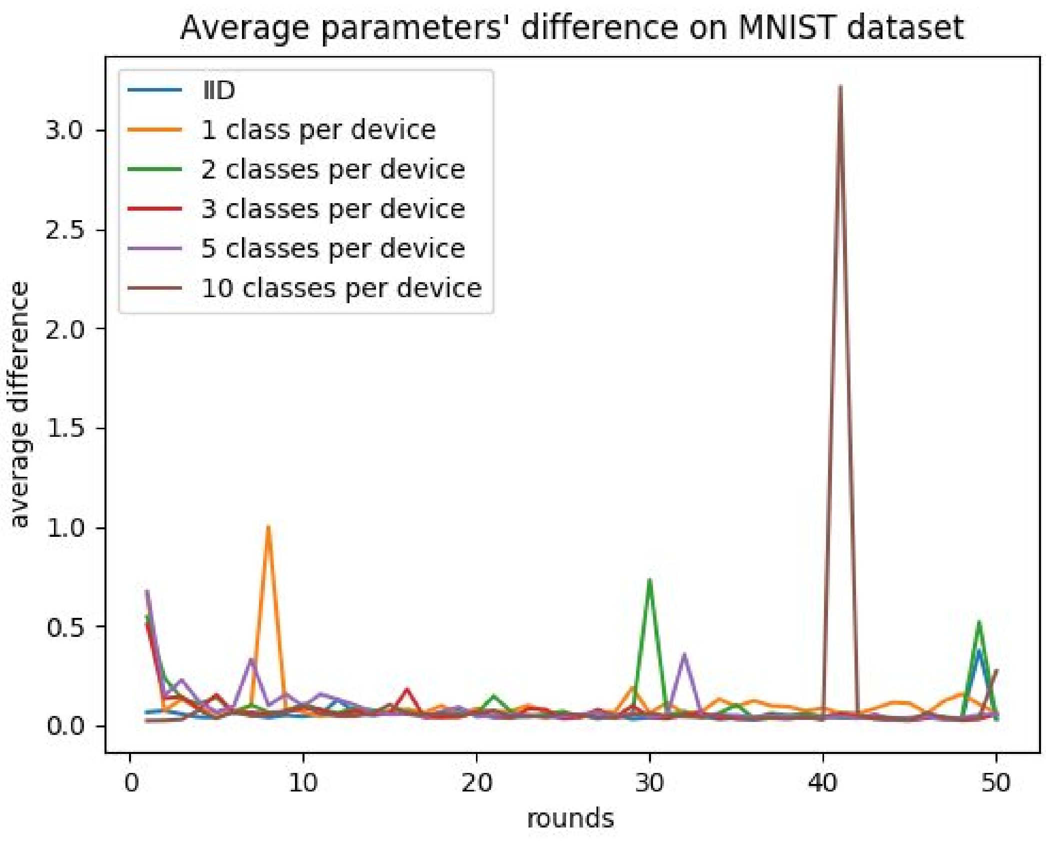


Figure 11: Parameters divergence on MNIST

3.5 Reasons for poor performance

There are several obvious reasons, poor model, training settings, training parameters, systems heterogeneity, highly non-IID data distribution because devices may be training for local objectives, not for global objectives so they are training different models from target learning model. Also, Federated Averaging is bad with non-IID data distribution, precision loses during the averaging step.

3.6 Effects of batch size

Five mini-batch sizes, namely 32, 64, 128, 256 and 512, are tested in 1, 2, 5 classes per device setting to evaluate the effects on performance. As shown in figure 12, 13,14, if mini-batch size is large, performance will be worse and unstable in this case because large batch lacks the explorative properties mini-batch has and tends to zoom in on the minimizer closest to the initial point and may converge to a sharp minimum, so general performance is poor. If the level of non-IID is high, then changing batch size does not affect performance much. But if the level of non-IID is low, batch size matters. Optimal choice may be related to the number of data on devices, which is 32 in these five values.

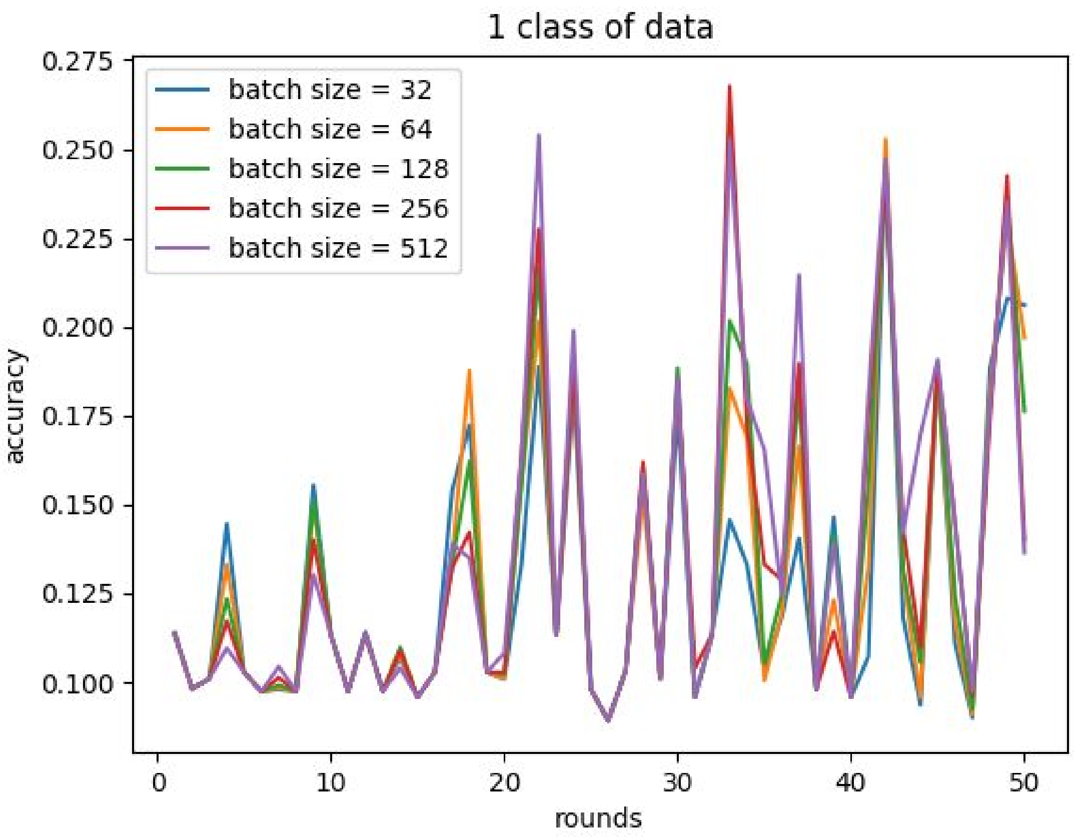


Figure 11: Performance with different batch sizes under the 1 class per device setting

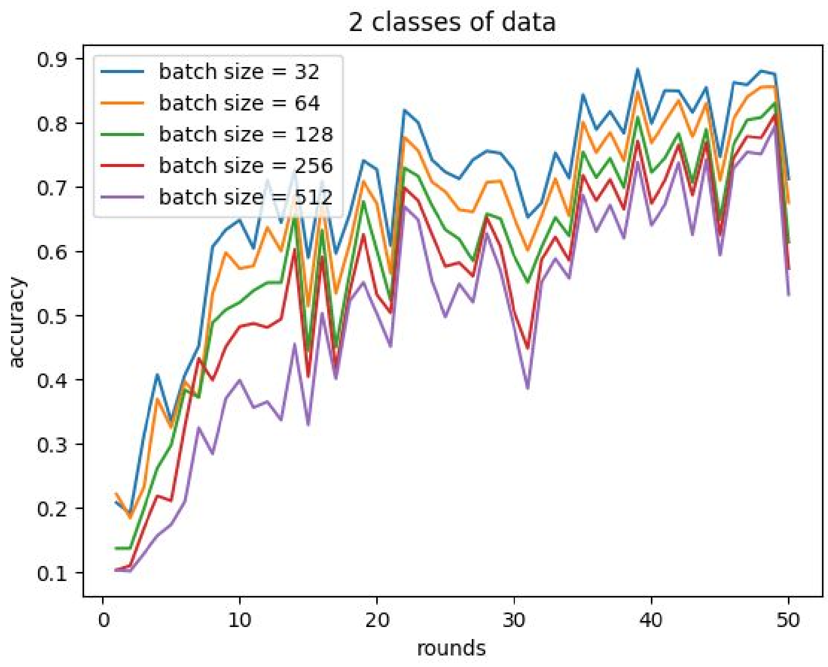


Figure 12: Performance with different batch sizes under the 2 classes per device setting

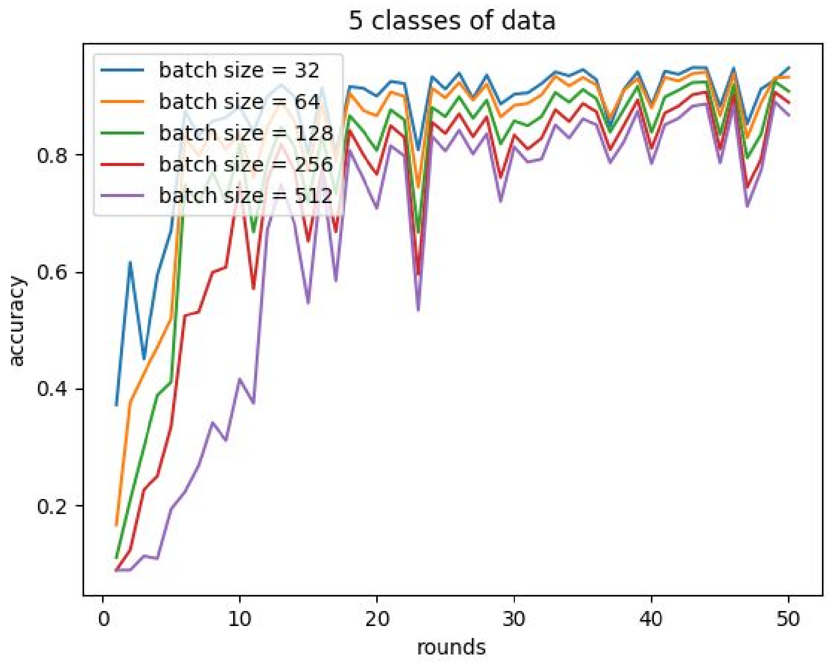


Figure 13: Performance with different batch sizes under the 5 classes per device setting

3.7 Effects of number of epochs

Three number of epochs, namely 5, 10 and 20, are tested in 1, 2, 5 classes per device setting to evaluate the effects on performance. As shown in figure 14, 15, 16, we can get slightly better performance if the level of non-IID is low, and nearly no improvement if the level of non-IID is high. Surprisingly, changing the number of epochs does not affect performance much as in tradition Machine Learning. Thus, the number of local epochs can be small to get similar model performance without affecting the performance of training devices. Also, too many local epochs may not be good as parameter divergence may be more serious.

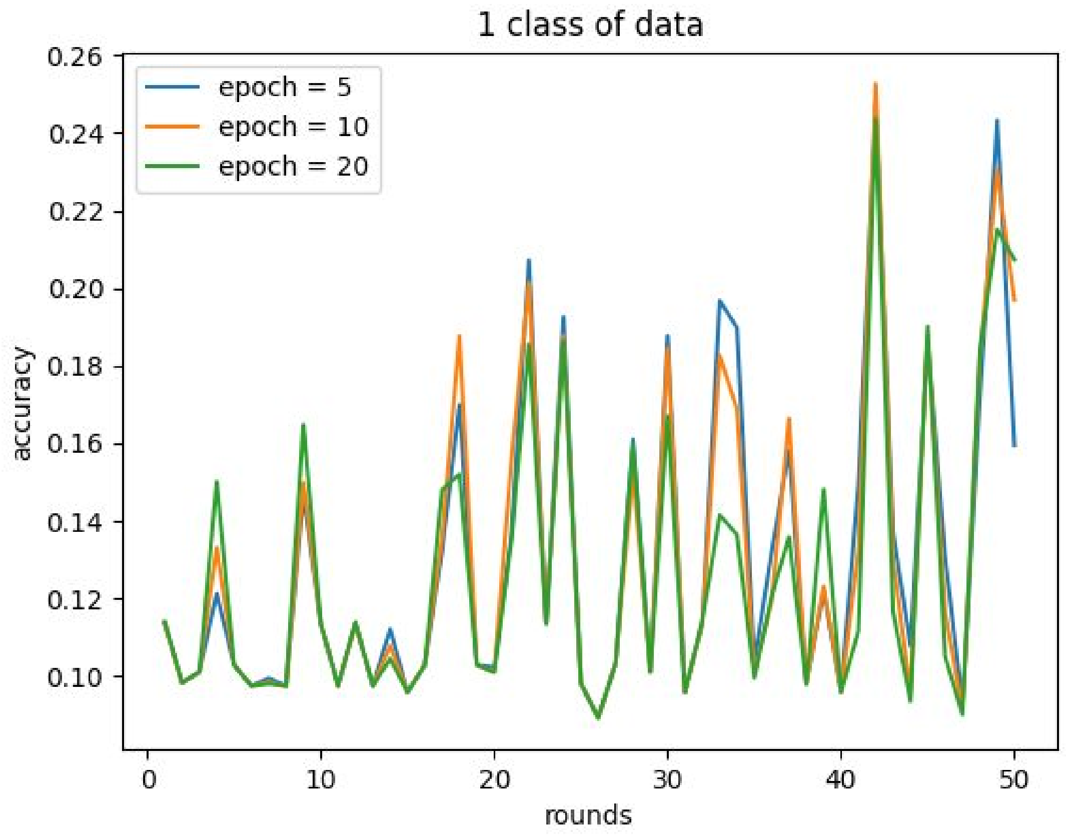


Figure 14: Performance with different number of epochs under the 1 class per device setting

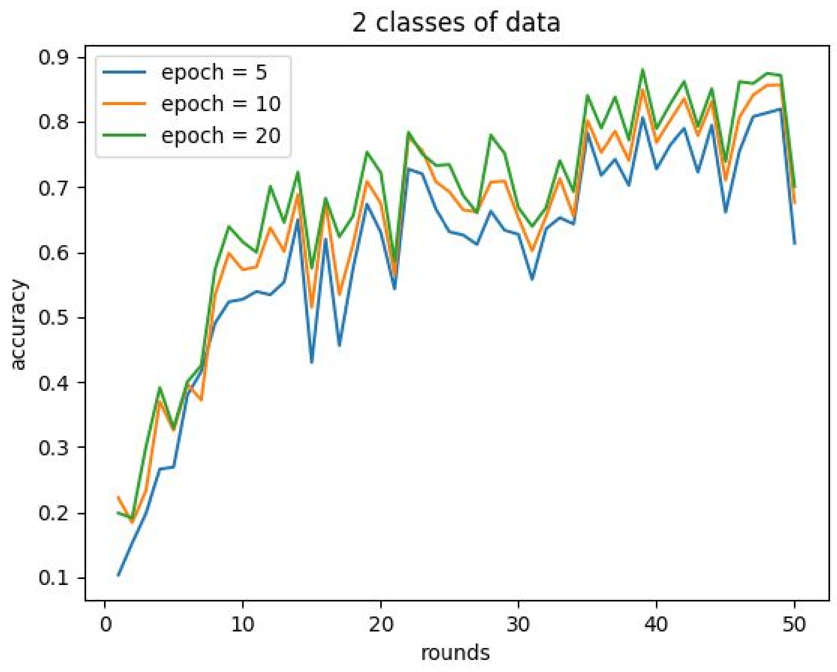


Figure 15: Performance with different number of epochs under the 2 classes per device setting

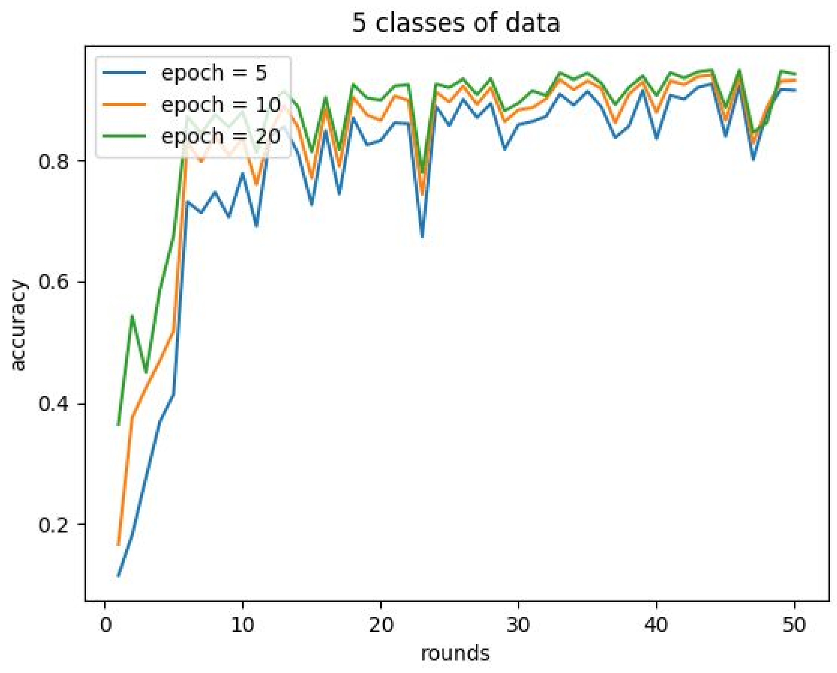


Figure 16: Performance with different number of epochs under the 5 classes per device setting

3.8 Full gradient descent using Federated Averaging

In this experiment, the batch size is ∞ and $E = 1$. It is to simulate full gradient descent and computational time is shorter. It is expected it may give more accurate gradients which are good for datasets. However, accuracy is poor. It requires a longer training time for a given accuracy. This is just an extreme case, but it shows the need to be careful to test different combinations of model parameters, such as grid search.

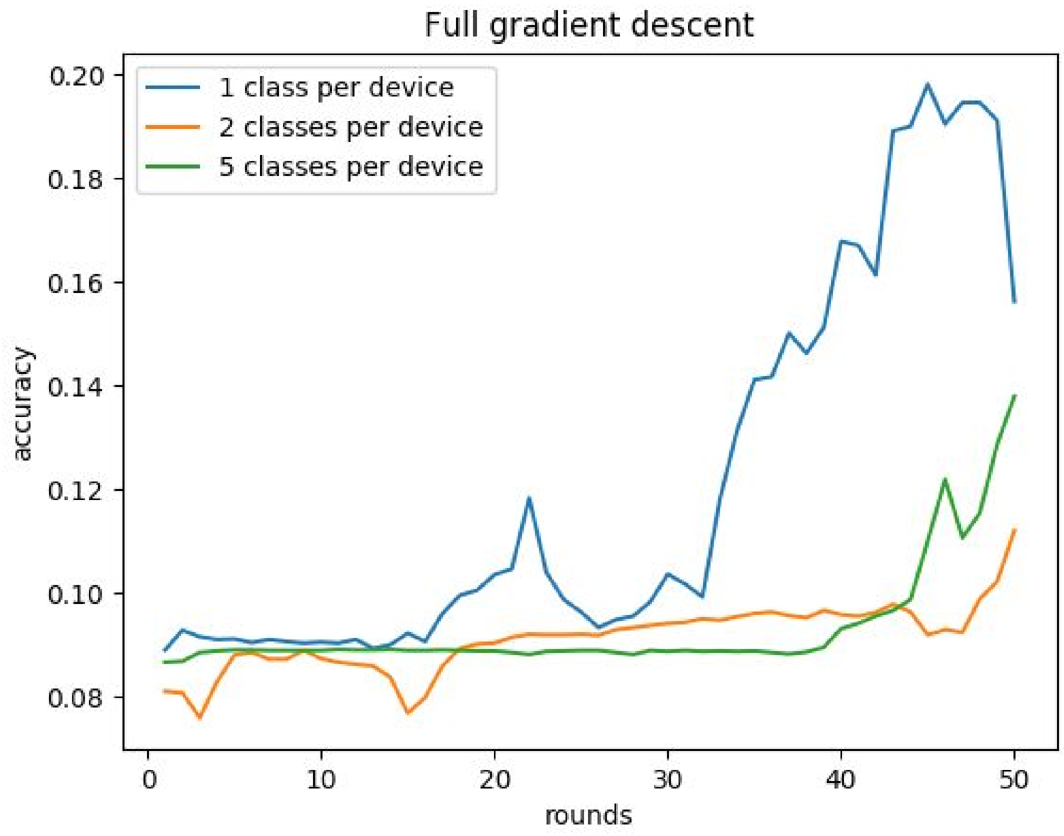


Figure 17: Performance with different number of epochs under the 5 classes per device setting

3.9 Effects of number of devices

In this experiment, each device has at most 200 data, which is reduced by $\sim \frac{2}{3}$ compared to the default setting. Three number of devices, namely 100, 200 and 300, are tested in 1, 2, 5 classes per device setting to evaluate the effects on performance. As shown in figure 18, 19, 20, when there are a large amount of devices training models, accuracy can be more stable and higher. This is expected as averaging is more sensitive to outliers when the number of devices is small.

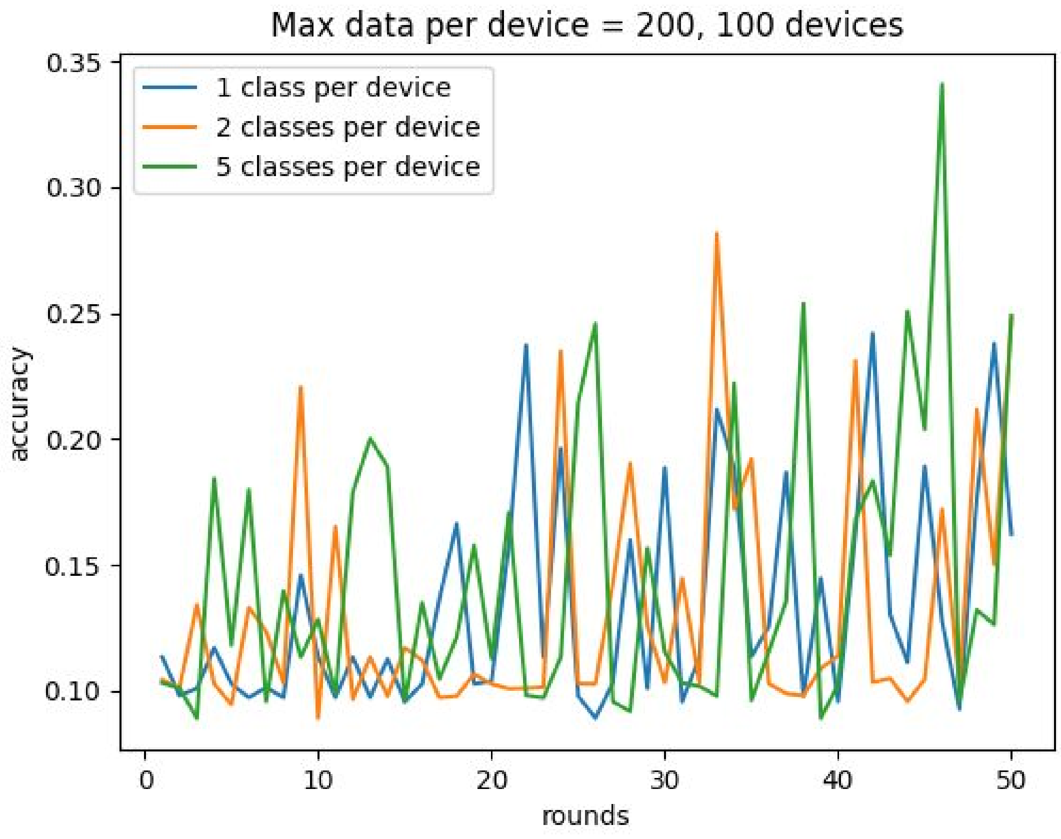


Figure 18: Performance with 100 devices and maximum 200 data under the non-IID setting

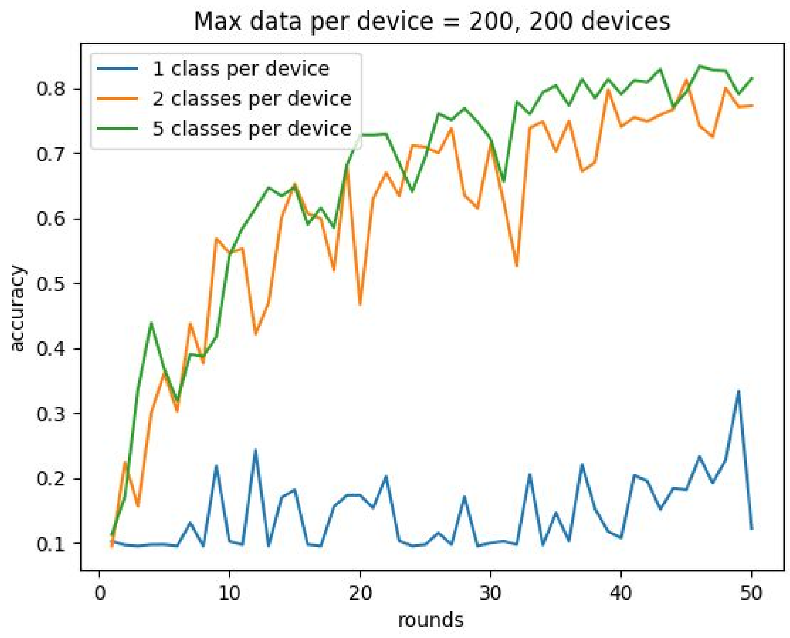


Figure 19: Performance with 200 devices and maximum 200 data under the non-IID setting

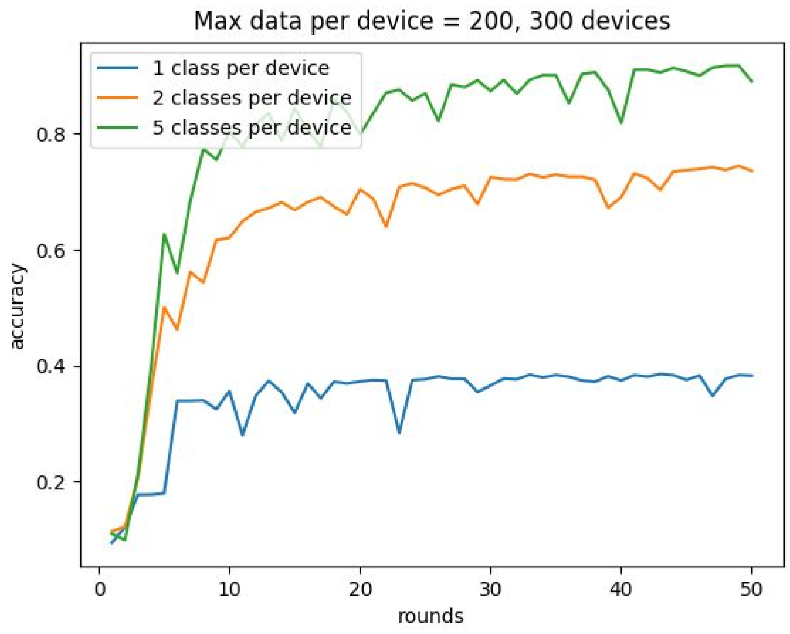


Figure 20: Performance with 300 devices and maximum 200 data under the non-IID setting

3.10 Effects of fraction of devices selected for training

Four numbers of the fraction of devices, namely 10%, 20%, 50%, and 100%, are tested in 1, 2, 5 classes per device setting to evaluate the effects on performance. As shown in figure 21, 22, 23, selecting more devices may be useful if data distribution under highly non-IID settings, but a high level of participation is generally impossible due to variability in network connectivity, hardware, and power among devices. As Federated Averaging assumes data distribution on devices selected is the same as the target learning model, the reason why performance increase with more devices selected for training is that data distribution on selected devices is similar to target distribution of the learning model.

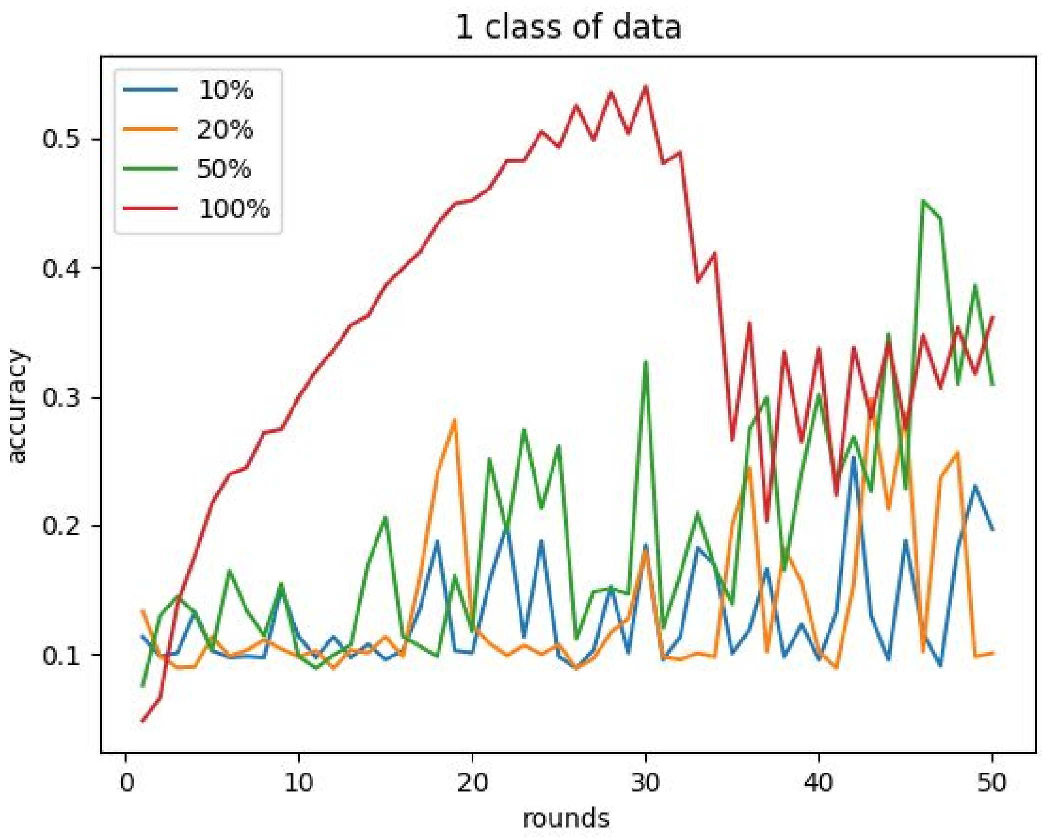


Figure 21: Performance with a different fraction of devices selected under the 1 class per device setting

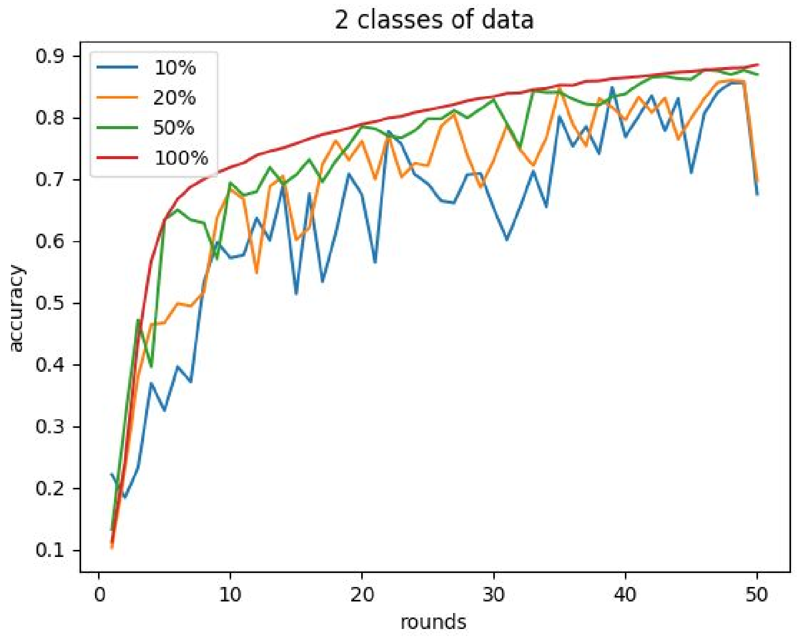


Figure 22: Performance with a different fraction of devices selected under the 2 classes per device setting

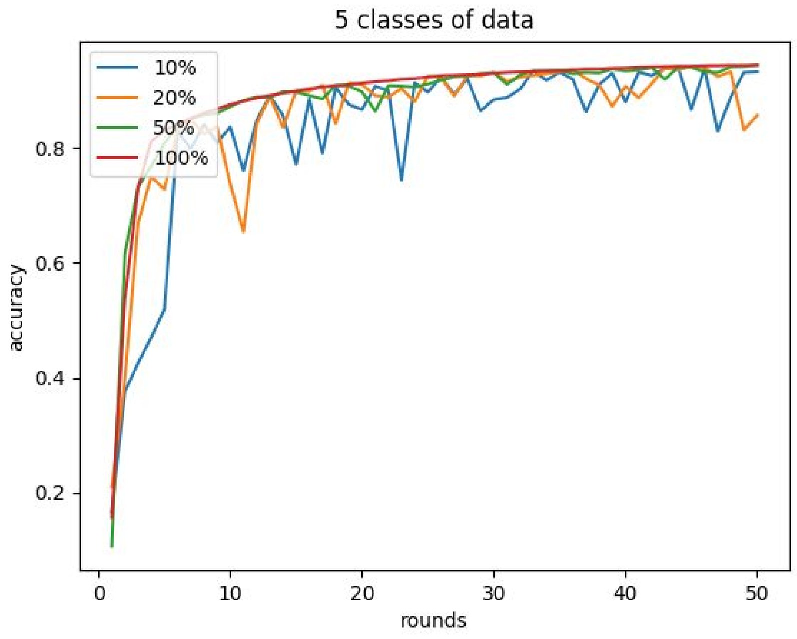


Figure 23: Performance with a different fraction of devices selected under the 5 classes per device setting

3.11 Effects of learning rate

Three learning rates, namely 0.001, 0.01, 0.1, are tested in 1, 2, 5 classes per device setting to evaluate the effects on performance. As shown in figure 24, 25, 26, it affects performance tremendously, especially under highly non-IID settings. In this case, 0.1 is optimal in 3 choices. It is a crucial learning parameter that needs to be tuned by recording loss. We may consider using it with decay and other optimization techniques.

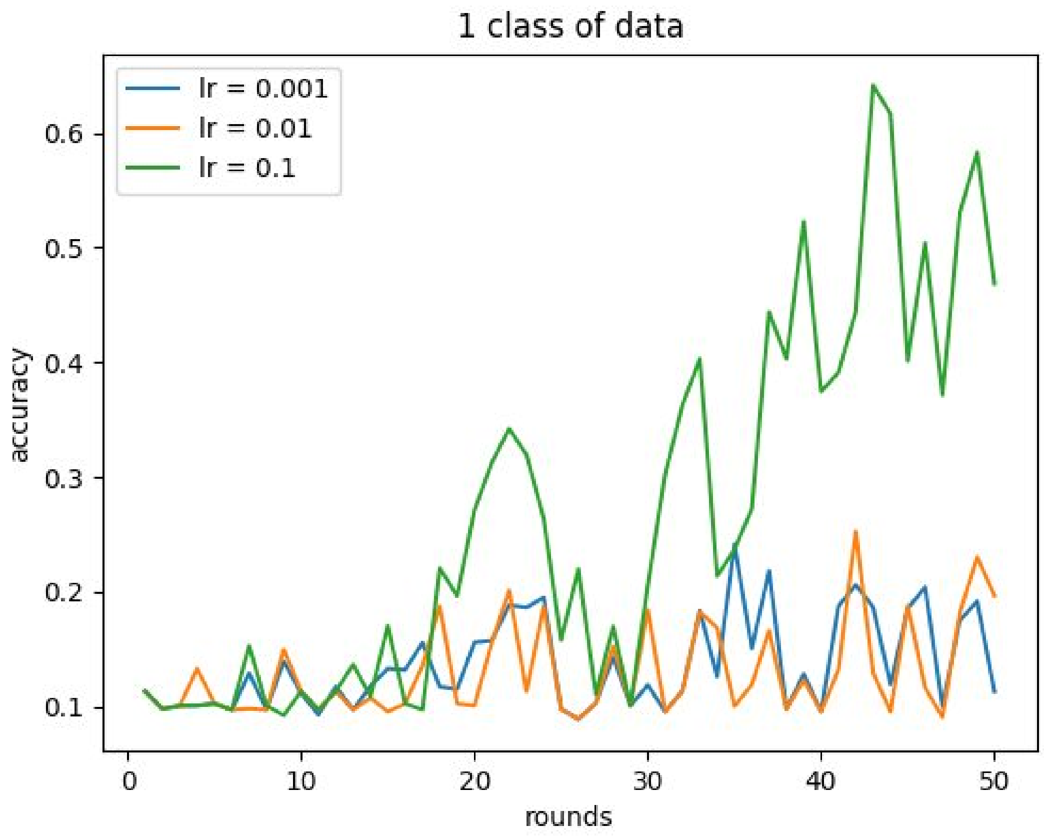


Figure 24: Performance with different learning rates under the 1 class per device setting

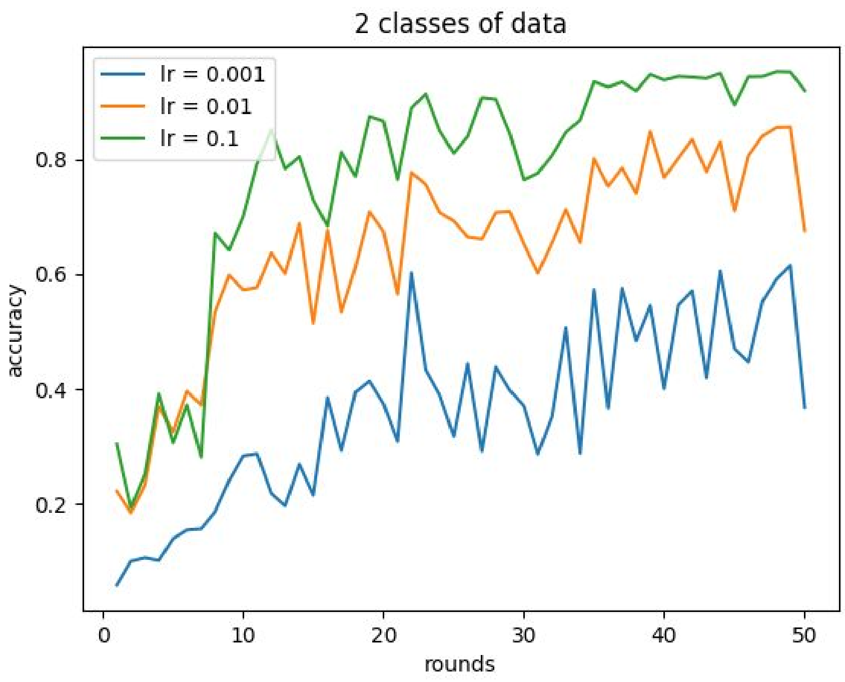


Figure 25: Performance with different learning rates under the 2 classes per device setting

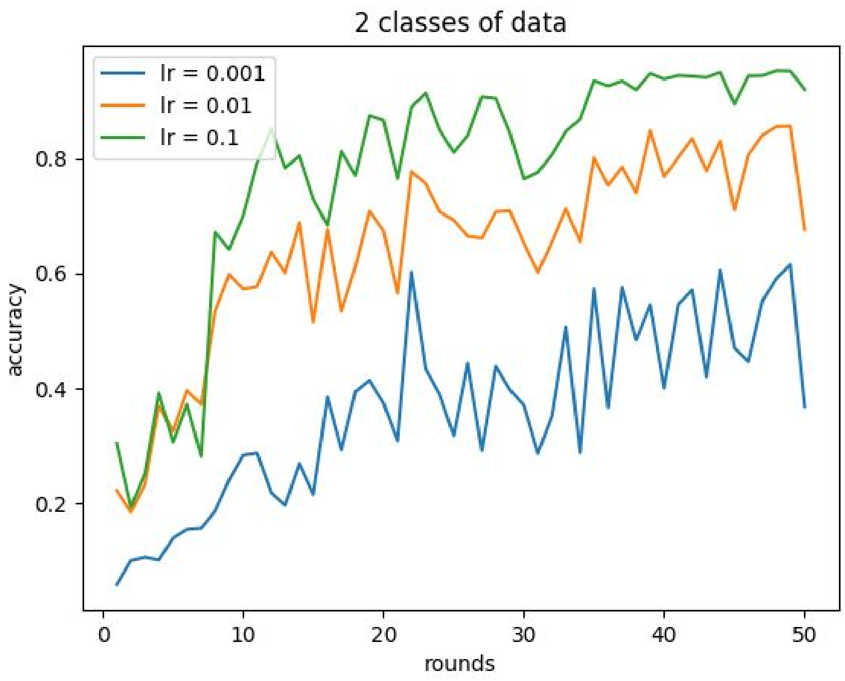


Figure 26: Performance with different learning rates under the 5 classes per device setting

3.12 Effects of optimizer

Four optimizers, namely vanilla SGD, SGD with momentum 0.9, Adagrad, Adam, are tested in 1, 2, 5 classes per device setting to evaluate the effects on performance. As shown in figure 27, 28, 29, the optimizer is important if not seriously non-IID. Nowadays many people like using adaptive optimizers like Adam because they use adaptive learning rates, which provide faster convergence speed and reduce the time of training. However, the results show that, in the federated setting, adaptive optimizers may be sensitive to model parameters being averaged. So SGD is still advantageous in performance.

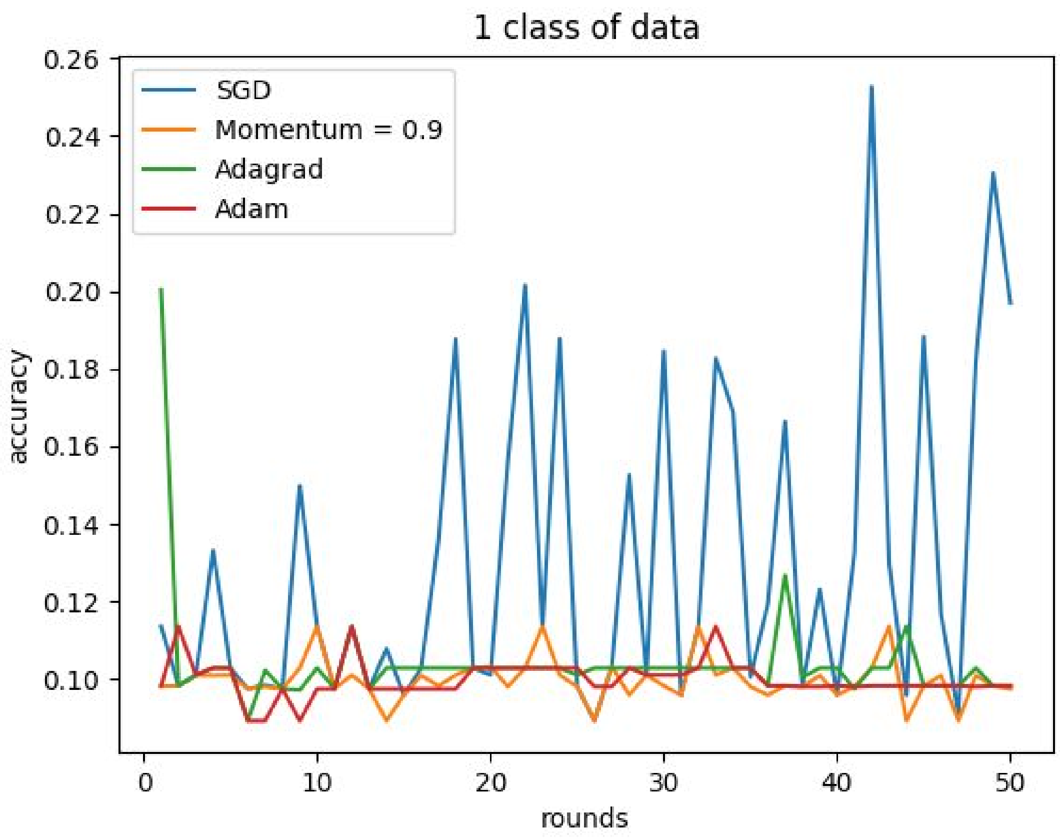


Figure 26: Performance with different optimizers under the 1 class per device setting

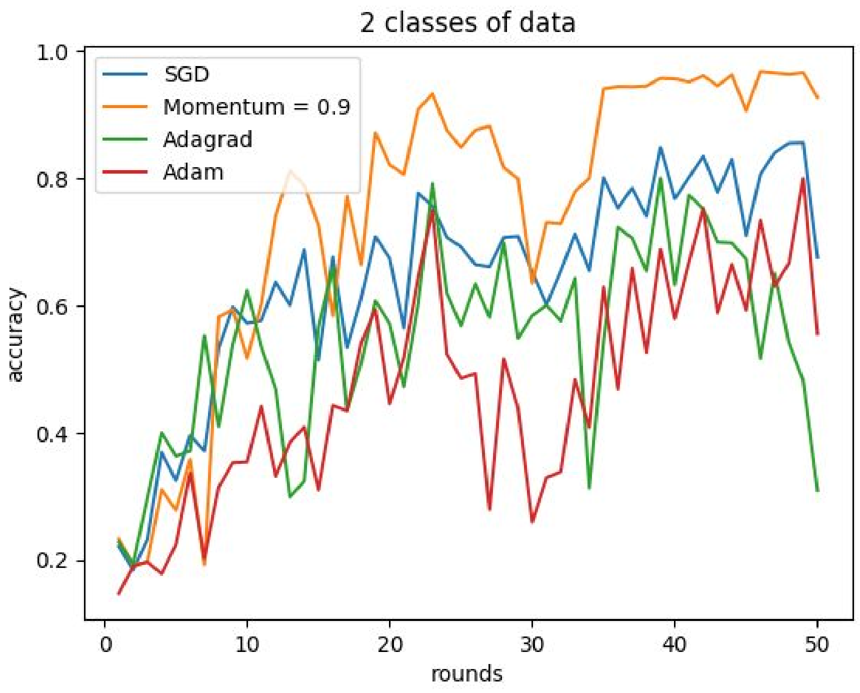


Figure 27: Performance with different optimizers under the 2 classes per device setting

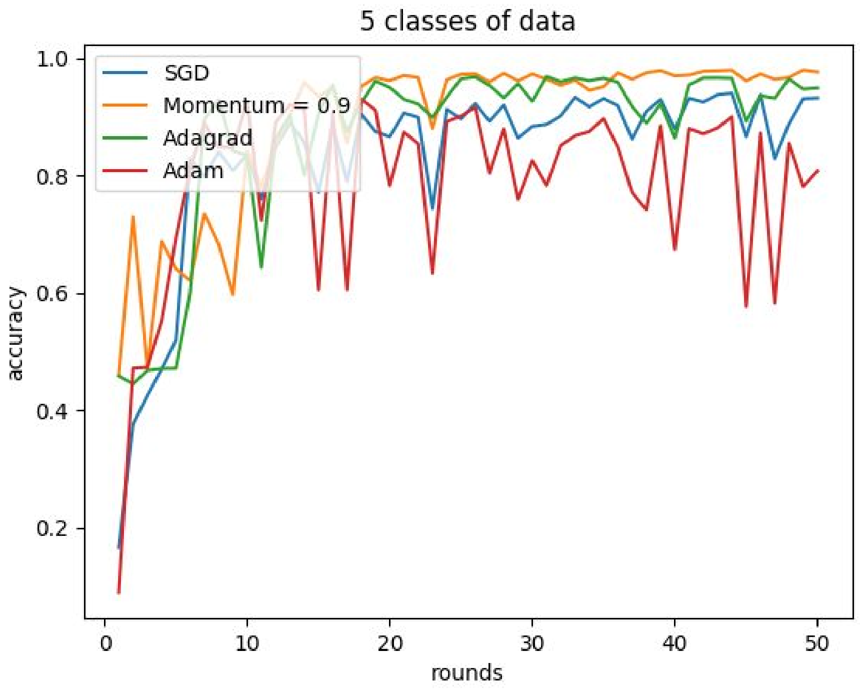


Figure 28: Performance with different optimizers under the 5 classes per device setting

3.13 Performance on CIFAR-10

Although most of the experiments are done on MNIST dataset, experiments on CIFAR-10 dataset are done to see if the performance issue exists. The CIFAR-10 dataset is slightly more complex than the MNIST dataset as images from CIFAR-10 have 3 color channels. As shown in figure 29 and 30 training on it also suffers from the performance issue. By observing the statistics, we can see that the performance is monotonically improving for slightly non-IID distribution, but it is limited by the weak 5-layer CNN model. Validation accuracy can only achieve 52.43% after 50 communication rounds.. It is expected datasets with more classes may have worse results, such as CIFAR-100.

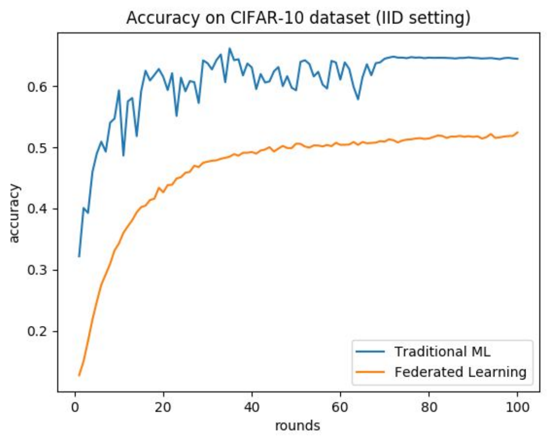


Figure 29: Performance on CIFAR-10 under IID setting

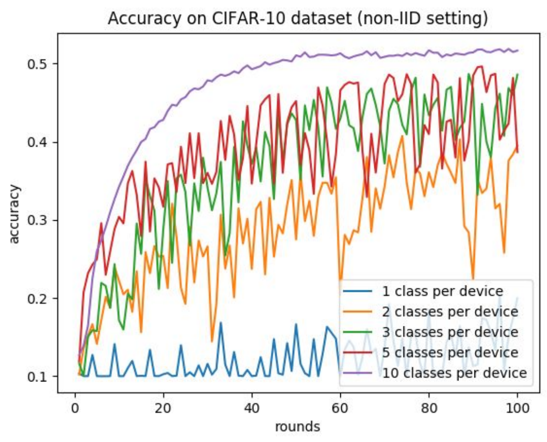


Figure 30: Performance on CIFAR-10 under the non-IID setting

3.14 Effects of CNN architecture

How about using advanced CNN architecture such as VGG19, GoogleNet, and ResNet152? They provide higher accuracy and address overfitting by data augmentation and dropout. But can we use these on low-powered devices like smartphones? There is variability in network connectivity, hardware, and power. Devices like smartphones and sensors may not be able to train on the model due to extremely long training time and many parameters, which may use up lots of storage space and memory and affect the performance of devices. Therefore, a small number of local epochs may be used. Using a better CNN model may provide better performance but has lots of drawbacks, so it may not be practical in some cases.

Two CNN architectures, namely 5-layer CNN, VGG11, are tested in 1, 2, 5 classes per device setting to evaluate the effects on performance. In this experiment, $E = 1$. As shown in figure 31 using VGG11 gives better performance, accuracy can reach 80.69% under the 1 class per device setting, which is higher than 47.85% using 5-layer CNN. Accuracies are more stable and it is expected with better choices of training parameters such as learning rate, we may get better performance. Using advanced CNN architecture can get better performance, as adding more layers beyond a certain threshold may lead to finding irregularities in the data and reduce overfitting.

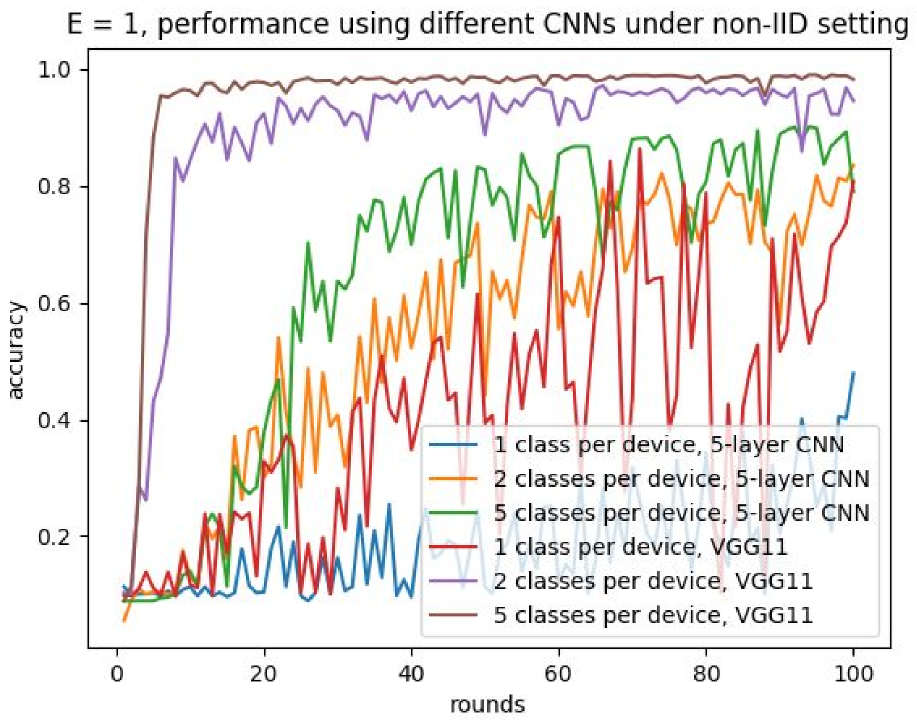


Figure 31: E = 1, performance using different CNN architectures under the non-IID setting

3.15 Data sharing technique

As mentioned in section 1.5.1, scholars suggested data sharing technique to improve performance under the non-IID setting. They stated in the paper that "experiments show that accuracy can be increased by ~30% for the CIFAR-10 dataset with only 5% globally shared data". The server distributes a small subset of global data and a warm-up model to the clients at the initialization stage. The principle is that devices have more than 1 class of data, so diversity of data increases and data on each device is less non-IID. Therefore, weight divergence can be reduced and performance can be better. Although shared data may be collected before the initialization stage of Federated Learning, it may violate previous users' privacy as the server sends their data to other users.

However, amount of globally shared data mentioned in the paper is expressed in terms of the fraction of total amount of data on all devices, which is not intuitive and flexible in case number of devices is large, so amount of globally shared data may be expressed in terms of the fraction of average amount of data on device. As shown in figure 3 if there are 80% non-IID data on devices, 20% data are shared IID data, then overall accuracy can boost by about 40%. This strategy can improve performance tremendously, but it should also be tested on highly skewed datasets.

3.16 Can transfer learning improve performance?

A pre-trained model with 60.8% accuracy is used in this experiment. As shown in figure 32, the accuracy of 1 class per device boosted from ~20% to ~60%. Compared to figure 4, there is nearly no difference for 2 and 5 classes per device. This strategy seems useful, but not, because 1 class per device can only achieve ~60% accuracy, and accuracy may drop over time, due to the properties of Federated Averaging.

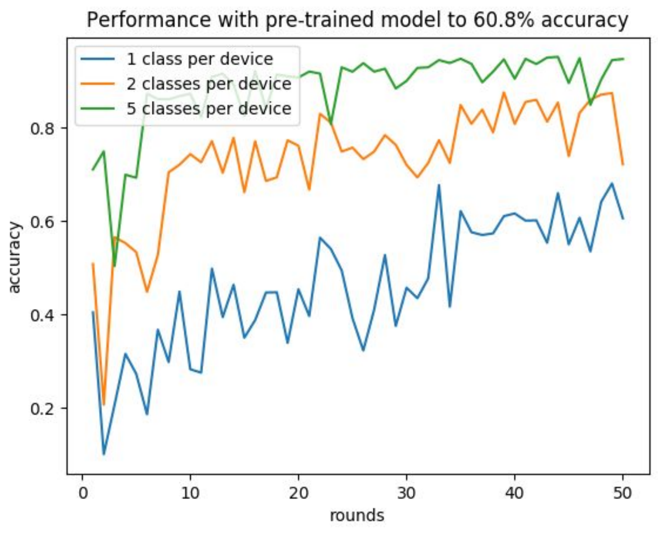


Figure 32: Performance with a pre-trained model to 60.8% accuracy

4 Conclusion

There are lots of reasons for poor accuracy: bad training setting, non-IID data distribution, and bad Federated Learning algorithm, which is the most important one. Tuning training parameters and good training settings are important in traditional Machine Learning and Federated Learning. Even with good model, training parameters, optimizer, you cannot solve the problem, you need to use alternative FL algorithms, like FedCurv [18], which builds on ideas from Lifelong Learning and adapts the Elastic Weight Consolidation algorithm from a sequential algorithm to a parallel one, and FedProx [19], which provides a way for the server to account for the heterogeneity associated with partial information, with proved robustness and stability.

There are some limitations to this project. First, cross-validation is not used because the goal is to estimate performance among devices and describe the relationship between performance and training settings, real-world scenarios are more complex and may require cross-validation. Complex datasets such as ImageNet are not used in this project because of time constraints. Communication constraints are not considered but lower accuracy is expected with communication optimizations like random masks and probabilistic quantization.

5 Future works

More effects of training settings can be tested, such as other CNN architectures, batch normalization, different combinations of optimal learning parameters. More metrics and explanations may be added in the future. Datasets with more classes or highly-skewed datasets can also be tested, such as ImageNet. Other Federated Learning algorithms and optimization techniques may also be studied and evaluated, such as applying lifelong learning, agnostic learning and multi-task learning to Federated Learning. Communication constraints may also be considered so the problem of Federated Averaging may be better reflected.

6 References

[1] Cohen, Jacob. "A coefficient of agreement for nominal scales." Educational and psychological measurement 20, no. 1 (1960): 37-46.

[2] Konečný, Jakub, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. "Federated learning: Strategies for improving communication efficiency." arXiv preprint arXiv:1610.05492 (2016).

[3] Nishio, Takayuki, and Ryo Yonetani. "Client selection for federated learning with heterogeneous resources in mobile edge." In ICC 2019-2019 IEEE International Conference on Communications (ICC), pp. 1-7. IEEE, 2019.

[4] McMahan, H. Brendan, Eider Moore, Daniel Ramage, and Seth Hampson. "Communication-efficient learning of deep networks from decentralized data." arXiv preprint arXiv:1602.05629 (2016).

[5] Konečný, Jakub, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. "Federated learning: Strategies for improving communication efficiency." arXiv preprint arXiv:1610.05492 (2016).

[6] Zhao, Yue, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. "Federated learning with non-iid data." arXiv preprint arXiv:1806.00582 (2018).

[7] Geyer, Robin C., Tassilo Klein, and Moin Nabi. "Differentially private federated learning: A client level perspective." arXiv preprint arXiv:1712.07557 (2017).

[8] Sattler, Felix, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. "Robust and communication-efficient federated learning from non-iid data." arXiv preprint arXiv:1903.02891 (2019).

[9] “THE MNIST DATABASE.” MNIST handwritten digit database, Yann LeCun, Corinna Cortes and Chris Burges. Accessed January 5, 2020. http://yann.lecun.com/exdb/mnist/.

[10] “VGG Face Dataset.” Visual Geometry Group - University of Oxford. Accessed January 5, 2020. http://www.robots.ox.ac.uk/~vgg/data/vgg_face/.

[11] CIFAR-10 and CIFAR-100 datasets. Accessed January 5, 2020. https://www.cs.toronto.edu/~kriz/cifar.html.

[12] Georgiev, Kostadin, and Preslav Nakov. "A non-iid framework for collaborative filtering with restricted boltzmann machines." In International conference on machine learning, pp. 1148-1156. 2013.

[13] Konečný, Jakub, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. "Federated learning: Strategies for improving communication efficiency." arXiv preprint arXiv:1610.05492 (2016).

[14] Zhou, Zhi-Hua, Yu-Yin Sun, and Yu-Feng Li. "Multi-instance learning by treating instances as non-iid samples." In Proceedings of the 26th annual international conference on machine learning, pp. 1249-1256. ACM, 2009.

[15] “TensorFlow Federated.” TensorFlow. Accessed January 5, 2020. https://www.tensorflow.org/federated.

[16] Ryffel, Theo, Andrew Trask, Morten Dahl, Bobby Wagner, Jason Mancuso, Daniel Rueckert, and Jonathan Passerat-Palmbach. "A Generic Framework for Privacy Preserving Deep Learning." 2018.

[17] Paszke, Gross, Massa, Lerer, Bradbury, Chanan, Killeen, Lin, Gimelshein, Antiga, Desmaison, Köpf, Yang, DeVito, Raison, Tejani, Chilamkurthy, Steiner, Fang, Bai, and Chintala. "PyTorch: An Imperative Style, High-Performance Deep Learning Library." 2019.

[18] Shoham, Neta, Tomer Avidor, Aviv Keren, Nadav Israel, Daniel Benditkis, Liron Mor-Yosef, and Itai Zeitak. "Overcoming Forgetting in Federated Learning on Non-IID Data." 2019.

[19] Li, Tian, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. "Federated Optimization in Heterogeneous Networks." 2018.