Understanding the main challenges of Federated Learning

Alessandro Casella
Politecnico di Torino
Student ID: s306081
s306081@studenti.polito.it

Davide Di Mauro
Politecnico di Torino
Student ID: s306089
s306089@studenti.polito.it

Angelica Marrone
Politecnico di Torino
Student ID: s291261
s291261@studenti.polito.it

***Abstract*—In this report we analyze in depth different federated learning algorithms. We first start selecting a convolutional neural network and CIFAR-10 as dataset, and compute a baseline for reference; we then proceed to evaluate FedAVG, the first and most adopted solution in the federated scenario. Furthermore, other federated algorithms are considered, in order to address different issues, for instance stagglers, client model complexity and classifier calibration. For each of the aforementioned algorithms we consider one of the most challenging problems in federated learning, which is dealing with non-IID data distribution, and compare the results. All code is publicly available at: https://github.com/dadodimauro/MLDL-Federated-Learning**

I. Introduction

Recently, the progression in Machine Learning has increased its use: usually, these techniques have a centralized approach but in reality, most of the data (i.e. the data on mobile phones, etc...) is distributed among many devices, and collecting this data can be challenging; moreover since this data usually is user-sensitive, there are also privacy limitations.

Federated Learning is a paradigm that tries to address such problems.

Federated Learning (FL), also known as federated optimization, allows multiple parties to collaboratively train a model without data sharing. Similar to the centralized parallel optimization, FL lets the user devices perform most of the computation and a central parameter server update the model parameters using the descending directions returned by the user devices.

Three of the main challenges that FL has to face are: First, the training data are distributed over an incredibly large number of devices, and the connection between the central server and a device is slow, leading to a consequent slow communication. Second, the FL system does not have control over users' devices, which means that the system can do nothing but waiting or ignoring the stragglers. It is impractical to require all the devices to be active.

Third, the training data are non-iid, which means that the data available locally fail to represent the overall distribution. This does not only bring challenges to algorithm design but also makes theoretical analysis much harder.

Therefore our goal is to analyze and compare different algorithms while investigating the effect of data heterogeneity on them. ResNet50 [1] is chosen as CNN's architecture and the centralized model baseline is computed before the experiments. We start with FedAVG [2], which is the first and perhaps the most widely used FL algorithm due to its conceptual easiness; we then implement FairAVG [3] by simply changing the averaging scheme of FedAVG and FedProx [4], to alleviate the stragglers' effect on the algorithm. We then address the topic of systems heterogeneity with FedGKT [5], a model in which the client uses a "lighter" model on the clients with respect to the server one. We also analyze the problem of the biased classifier, that can be mitigated using CCVR [6], a classifier calibration technique. Finally we briefly analyze some privacy problems performing the Gradient Inversion Attack [7], [8] on the trained model of a client. In all our test the dataset used is CIFAR-10.

II. Preliminaries

A. Normalization

Batch normalization [9] normalizes the features by the mean and variance computed in a mini-batch. It is required for BatchNorm to work with a sufficiently large batch size. Batch Normalization exhibits drawbacks: if batch size is too small, the estimation of the batch statistics becomes unstable, in fact reducing the batch size increases the BN's error.

on the other hand, BN's error increases rapidly when reducing the batch size.

Group Normalization [10] is a strong alternative to BN. GN mitigates the problem, taking away the dependance on batch size. Instead of normalizing across the batch dimension, GN normalizes the features across the groups, so that its computation is independent of batch sizes.

B. Stragglers

The *"straggler's effect"* in real-world applications means that everyone waits for the slowest device. For example, if there are thousands of users' devices in the federated learning system, there is always a small portion of devices offline. Different approaches can be utilized to manage stragglers: the most common one is dropping them (*e.g.* FedAVG) but there are also different algorithms that allows us to use the partial updated model of a straggler (*e.g.* FedProx).

C. Effect of non-IID data

In real-world cases, local dataset could not follow the population distribution: handling the decentralized non-IID

data still remains a statistical challenge in the field of federated learning [11].
For instance, it can be observed that there are severe performance degradation in multi-class classification accuracy under highly skewed non-IID data [12]; since the distribution of each local dataset is highly different from the global distribution, the local objective of each party is inconsistent with the global optimum. Thus, there exists a drift in the local updates. In other words, in the local training stage, the local models are updated towards the local optimum, which can be far from the global one. The averaged model may also be far from the global optimum especially when the local updates are large. Eventually, the converged global model has much worse accuracy than IID settings [13].

D. Data Distribution

In all the experiments, we analyzed 3 different data distribution cases: (i) IID case, (ii) non-IID case with balanced data, (iii) non-IID case with unbalanced data.

(i) In this case the data is simply i.i.d. distributed, so in the case of CIFAR-10 each client sees the same amount of images for each of the 10 classes (Figure 6).
(ii) In this case the same amount of images is assigned to each client and everyone of them has access to either one or two classes (Figure 7).
(iii) In this case each client has access to a different amount of images, and sees a different number of classes (Figure 8).

TABLE I: Data Distribution

	IID	**non-IID balanced**	**non-IID unbalanced**
mean	10	1.94	5.72
variance	0	0.24	2.63

III. Algorithms

A. FedAVG

Federated Averaging (FedAvg) was the first algorithm proposed by McMahan et al. (2017) [2]. For each global round, we first select k devices among all the clients, then we train each local model for E local epochs and send the updated models to the central server. The central server then conducts a weighted average over the models received from the selected devices and broadcasts the averaged model to all devices.

B. FairAVG

FedAVG aggregates weight of the models according to the amount of data of each client, claiming that models trained on more data are better. But it can be experimentally proved that a fair aggregation can improve both accuracy and convergence rate: this is the idea behind *FairAVG*, a modified version of FedAVG that aggregates the clients models simply doing the average instead of the weighted one.

C. FedProx

FedProx tries to improve FedAVG with some simple and lightweight modifications, mitigating the effect of statistical heterogeneity. It allows to incorporate partial information from stragglers (instead of dropping them), but this increases statistical heterogeneity and decreases convergence rate. To alleviate this issue it introduces an additional regularization term, called proximal term in the local objective function to effectively, limit the impact of variable local updates; a new hyperparameter μ is introduced to control the weight of regularization term.

D. FedGKT

Federated Group Knowledge Transfer is a FL framework that addresses the resource-constrained reality of edge devices. It reformulates FL as an alternating minimization approach to train small CNNs on edge nodes and periodically transfer their knowledge to a larger server-side CNN. We choose a ResNet8 as our edge CNN, which consists of a lightweight feature extractor and classifier, while our server model is a ResNet49 (a ResNet50 without the first convolutional layer because it receives as input the output of the smaller CNN). The main advantages offered by this method are the following: compared to FedAVG, FedGKT demands less computational power on edge devices and fewer parameters in the edge model (2300 times less in our case).

E. CCVR

Classifier Calibration with Virtual Representation is a technique consisting on adjusting the classifier using virtual representations sampled from an approximated Gaussian Mixture Model in the feature space with the learned feature extractor. The use of this method is justified by the "client drift": when the global model is optimized with different local objectives with local optimums far away from each other, the average of the clients update moves away from the true global optimum. The idea is to use the feature extractor of the previously trained model to extract features from the test data and estimate the feature distribution, then re-training only the classifier using generated virtual representations from the feature distribution.

IV. Privacy

Gradient Inversion Attack

Gradient inversion attack aims to undermine the privacy and security of federated learning. It consists of intercepting gradients sent by the clients to the server. The term "inversion" refers to the input that the attacker recovers from the gradient. The input is reconstructed from the client's private data. The attack is performed by a malicious participant who intercepts the client's communications. Geiping et al., (2020) [8] demonstrate that information can be retrieved with a fairly high degree of accuracy under these two assumptions:

- the attacker knows BatchNorm statistics;
- the attacker has the ability to infer private labels.

Huang et al. (2021) [7] relaxed both assumptions, pointing out that they don't represent a realistic case and are uncommon

in modern deep learning. They demonstrate that relaxing such assumptions can significantly weaken the power of the attacks. They found out that not sharing the BatchNorm statistics, using a large batch size and combining multiple defenses are the best ways to prevent this attack. Moreover Hatamizadeh et al. (2022) [14], examines in depth the attack and provides a summarizing scheme to distinguish what is more or less insecure for privacy. In addition to what is mentioned above, large training sets and updates from a large number of iterations over different images are conditions needed to improve security.

V. Experiments

TABLE II: Notation

Symbol	Description
T	communication rounds
Ec	client epochs
Es	server epochs
B	batch size
BN	batch normalization
GN	group normalization
K	number of clients
C	fraction of clients used
S	percentage of strugglers
η	learning rate
μ	proximal term
Mc	virtual features generated

A. FedAVG

TABLE III: FedAVG

scheme	normalization	accuracy
IID	BN	0.68
	GN	0.42
non-IID balanced	BN	0.32
	GN	0.29
non-IID unbalanced	BN	0.29
	GN	0.21

T = 100, η = 0.001, K = 100, C = 0.1 B = 10, Ec = 1

We run the algorithm for 100 global epochs and 1 local epoch, using 3 different data distribution cases, both for BatchNorm and GroupNorm. The total number of clients is 100 and we select randomly 10 of them at each communication round. As we can see in Figure 1, Batch Normalization constantly gets us better results. We can also see that the non-IID cases perform worse than the IID ones and the balanced case better than the unbalanced ones (Table III).

We limit the number of global epochs to 100 for computational constraints on our side, but to prove that performances similar to the centralized baseline can be reached, we run FedAVG for 500 rounds (IID case) and confirm our hypothesis, as shown in Figure 9.

TABLE IV: FairAVG

scheme	algorithm	normalization	accuracy
non-IID unbalanced	FedAVG	BN	0.29
		GN	0.21
	FairAVG	BN	0.52 (+23%)
		GN	0.27 (+6%)

T = 100, η = 0.001, K = 100, C = 0.1 B = 10, Ec = 1

B. FairAVG

The experiment's setting is the same as the one described for FedAVG; we consider only the non-IID unbalanced case because it is the only scenario in which the new aggregation scheme behaves differently since the data in not balanced. The results are shown in Figure 9.

C. FedProx

TABLE V: FedProx

scheme	algorithm	μ	S	accuracy
IID	FedAVG	0	0	0.73
			0.5	0.71
			0.9	0.59
	FedProx		0.5	0.7
			0.9	0.64
		0.01	0	0.72
			0.5	0.71
			0.9	0.63
non-IID balanced	FedAVG	0	0	0.21
			0.5	0.14
			0.9	0.12
	FedProx		0.5	0.14
			0.9	0.18
		0.01	0	0.20
			0.5	0.16
			0.9	0.17
non-IID unbalanced	FedAVG	0	0	0.24
			0.5	0.42
			0.9	0.42
	FedProx		0.5	0.31
			0.9	0.30
		0.01	0	0.29
			0.5	0.33
			0.9	0.32

T = 30, η = 0.001, K = 100, C = 0.1 B = 10, Ec = 10

The settings for this experiment is the following: 30 rounds, 10 local epochs, and, for each data distribution case, we run the tests with different amounts of stragglers among the clients.

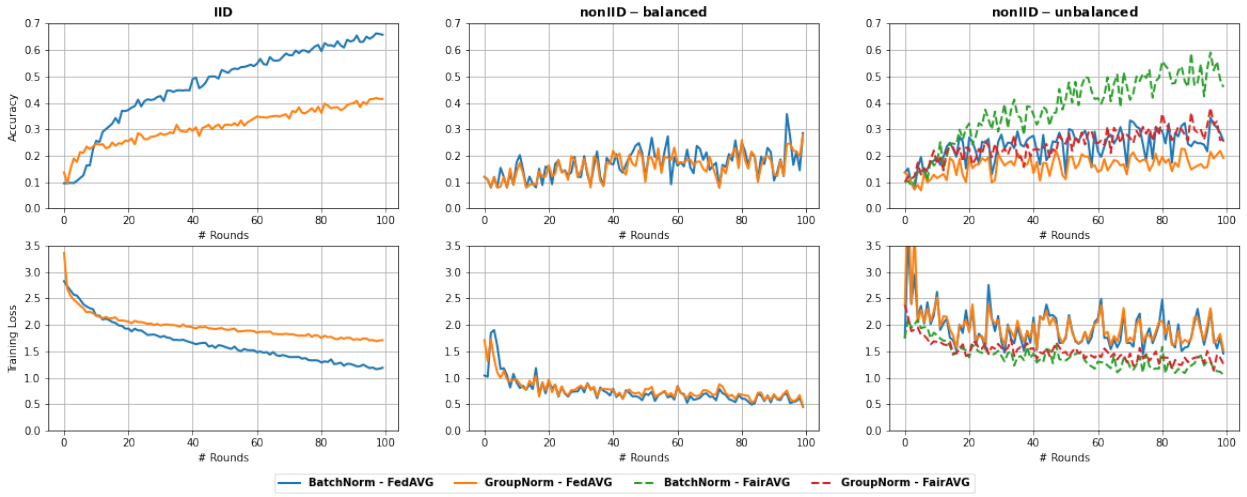


Fig. 1: FedAVG (and FairAVG) Accuracy and Training Loss for each data distribution setting

We compare three different procedures: FedAVG, FedProx with $\mu = 0$ and FedProx with $\mu > 0$ ($\mu = 0.01$). Notice that FedAVG and FedProx with $\mu = 0$ are the same in the case with no stragglers. We observe (Figure 2) that FedAVG performs slightly better in all cases, except for the IID one with 90% stragglers.

D. FedGKT

TABLE VI: FedGKT

scheme	normalization	accuracy
IID	BN	0.71
	GN	0.73
non-IID balanced	BN	0.31
	GN	0.24
non-IID unbalanced	BN	0.52
	GN	0.37

T = 10, K = 100, C = 0.1 B = 128, Es = 10 Ec = 1

In this scenario we run 10 communication rounds of FedGKT, comparing the two normalization techniques described above; the server performs 10 epochs, while each client performs 1 local epoch. As in the FedAVG case, batch normalization performs better than group normalization, and the non-IID unbalanced case achieves better results than the balanced case as illustrated in Figure 3 and Table VI.

E. CCVR

As shown in table x using CCVR we significantly improve the performance of FedAVG and FairAVG in the NON IID settings, while slightly decreasing the performance of the IID case. We also notice that varying the number of virtual features generated, the accuracy of the classifier is approximately the same.

TABLE VII: CCVR

algorithm	scheme	Mc	accuracy
FedAVG + CCVR	IID	100	66.43 (-1.6)
		500	66.39 (-1.6)
		1000	66.38 (-1.6)
	non-IID balanced	100	42.52 (+10.5)
		500	42.55 (+10.5)
		1000	42.45 (+10.4)
	non-IID unbalanced	100	37.17 (+8.2)
		500	37.10 (+8.1)
		1000	37.07 (+8.0)
FairAVG + CCVR	non-IID unbalanced	100	60.92 (+8.9)
		500	61.52 (+9.5)
		1000	61.54 (+9.5)

F. Gradient Inversion Attack

In order to address the privacy concerns related to FedAVG, we run a Gradient Inversion Attack on a trained client model (ResNet50) using the Breaching repository[1] and we see that reconstructing the original images is possible if a malicious attacker is able to gain access to the trained model of a user.

VI. Findings

After analyzing the results of our experiments, these are our findings:

1) Algorithm comparison

We can observe that there isn't a clear winner among the different algorithms:

[1] https://github.com/JonasGeiping/breaching

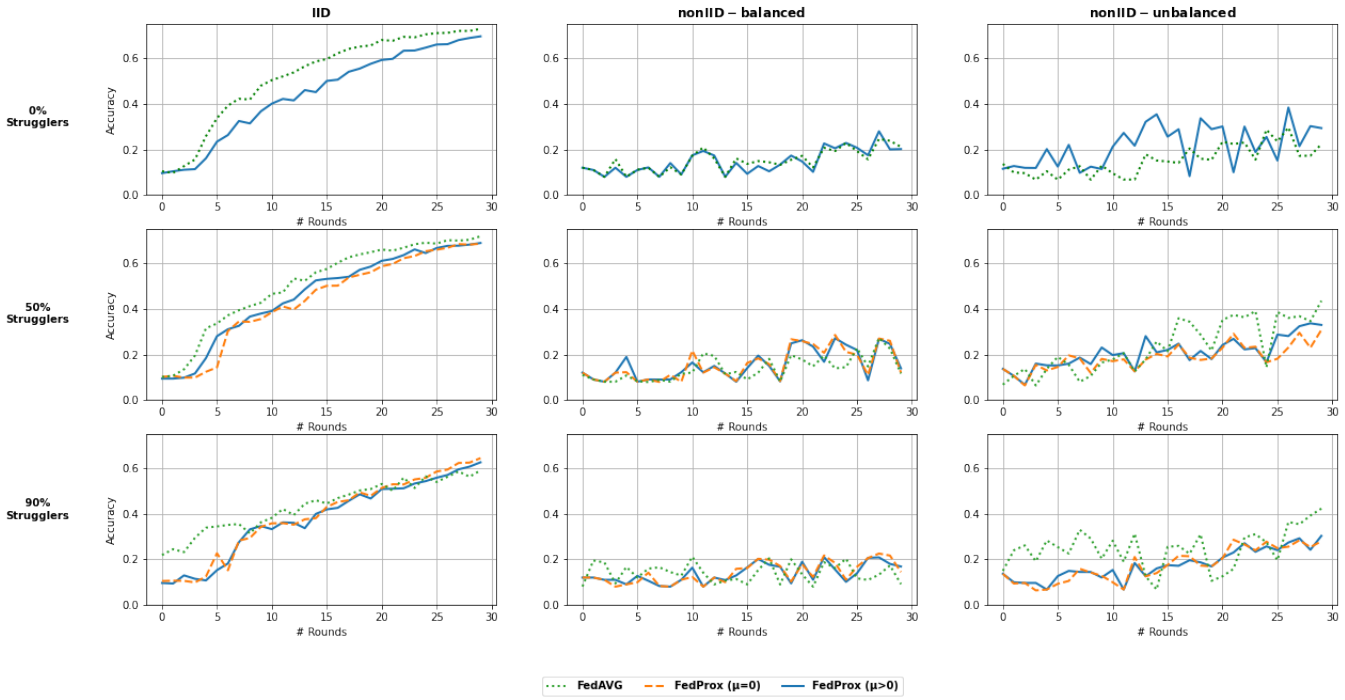


Fig. 2: FedProx Accuracy results in each setting considering different amounts of stragglers

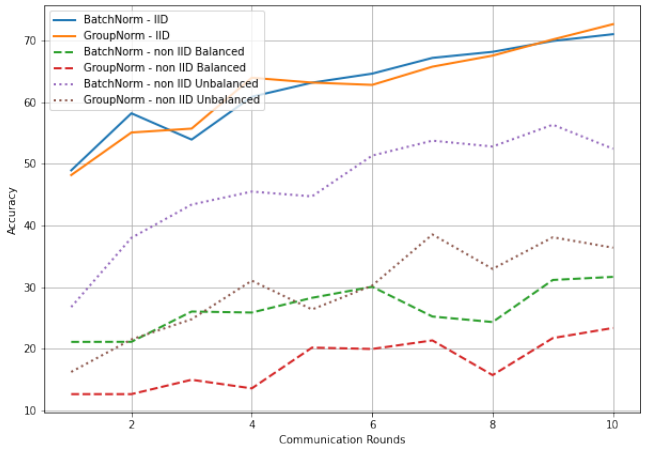


Fig. 3: FedGKT Accuracy results

- FedProx always achieves slightly worse results than FedAVG in all settings, improving performances only with an unrealistic amount (90%) of stragglers. We also show that not dropping stragglers in the non-IID cases makes FedProx more prone to client drifting.
- FairAVG increses substantially the performance of FedAVG in the non-IID unbalanced case, especially utilizing Batch Normalization and it also produce a much more

Fig. 4: Gradient Inversion Attack original images

Fig. 5: Gradient Inversion Attack reconstructed images

stable loss curve.
- FedGKT while not improving the performance of FedAVG represents an interesting approach, thanks to the low complexity of the client model, resulting in a more feasible solution to implement in real-world applications, for example the ones involving mobile phones as clients.

2) Effect of class distribution among clients

We notice that that the non-IID unbalanced results for FedGKT are significantly better than the non-IID balanced case. Our assumptions are motivated by the observation of the mean and standard deviation of the class distribution in the two different cases (Table I, Figures 7, 8):

- In the case of FedGKT the performance improvement is caused by the fact that each client has, on average, access to more classes, allowing to train a better model.
- We think that, for the same reason, the FedProx performances in the non-IID unbalanced case are slightly better than the non-IID balanced one, but the results are too unstable to draw a precise conclusion.
- We also notice that FedAVG outperforms FedProx in the non-IID unbalanced case with stragglers. Our theory to explain the reason why in the non-IID unbalanced case FedAVG seems to perform better is that using fewer clients at each communication round (selecting 10 clients and having 90% stragglers means dropping 9 of them) makes the algorithm less influenced by client drifting. The performances are better only in the unbalanced case because of the largest number of classes available to the clients.

Furthermore, considering the case of FairAVG, we found that having a fair aggregation scheme improves significantly the performance (up to 27%); this is caused by the fact that users with fewer samples, but with potentially high statistical variability, are considered equally during the aggregation.

3) Classifier calibration

As shown in the results of Table VII, CCVR represents a really effective way to increase the accuracy of the model, also considering that it requires only to re-train the classifier and not the complete model. Regarding why the performance improvement is only present in the non-IID cases, our hypothesis is that the classifier in the IID case (the simplest one) doesn't take advantages from the training with the virtual features since there is not client drift, i.e. the issue CCVR tries to solve.

Conclusions

In this work we first provide a brief description of different federated algorithms and the issues that each of them aims to address, more precisely data statistical heterogeneity, systems heterogeneity and privacy concerns. We then proceed to show the results obtained and comment them. We conclude pointing out what our findings are:

- there is not an algorithm the stands out as a clear winner;
- the statistical heterogeneity of data has a big impact on performances and remains probably the biggest challenge to overcome;
- we need to pay attenton to the fairness of the aggregation step, since FairAVG proves that performance can improve substantially;
- FedGKT represents the most real-world oriented algorithm of the group;
- CCVR is proven to be a state-of-the-art method capable of significantly increasing the accuracy of the model;
- The main goal of FL is to use sensitive data from the users in a privacy-constrained scenario, nevertheless we conclude that reconstruction of the original data is still possible and thus the privacy concerns are not to be taken lightly.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015. [Online]. Available: https://arxiv.org/abs/1512.03385

[2] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," 2017.

[3] U. Michieli and M. Ozay, "Are all users treated fairly in federated learning systems?" in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2021, pp. 2318–2322.

[4] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," 2018. [Online]. Available: https://arxiv.org/abs/1812.06127

[5] C. He, M. Annavaram, and S. Avestimehr, "Group knowledge transfer: Federated learning of large cnns at the edge," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 14 068–14 080. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/a1d4c20b182ad7137ab3606f0e3fc8a4-Paper.pdf

[6] M. Luo, F. Chen, D. Hu, Y. Zhang, J. Liang, and J. Feng, "No fear of heterogeneity: Classifier calibration for federated learning with non-iid data," 2021. [Online]. Available: https://arxiv.org/abs/2106.05001

[7] Y. Huang, S. Gupta, Z. Song, K. Li, and S. Arora, "Evaluating gradient inversion attacks and defenses in federated learning," 2021. [Online]. Available: https://arxiv.org/abs/2112.00059

[8] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients - how easy is it to break privacy in federated learning?" in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 16 937–16 947. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/c4ede56bbd98819ae6112b20ac6bf145-Paper.pdf

[9] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015. [Online]. Available: https://arxiv.org/abs/1502.03167

[10] Y. Wu and K. He, "Group normalization," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[11] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," 2020.

[12] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," 2018.

[13] Q. Li, Y. Diao, Q. Chen, and B. He, "Federated learning on non-iid data silos: An experimental study," 2021.

[14] A. Hatamizadeh, H. Yin, P. Molchanov, A. Myronenko, W. Li, P. Dogra, A. Feng, M. G. Flores, J. Kautz, D. Xu, and H. R. Roth, "Do gradient inversion attacks make federated learning unsafe?" 2022. [Online]. Available: https://arxiv.org/abs/2202.06924

APPENDIX

Fig. 6: IID

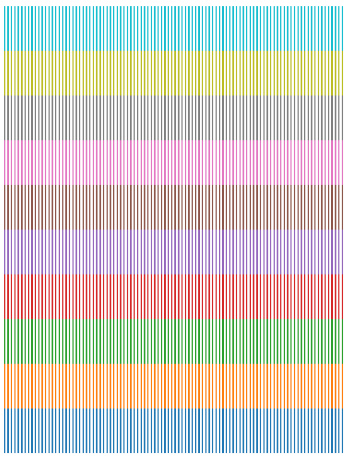


Fig. 7: non-IID Balanced

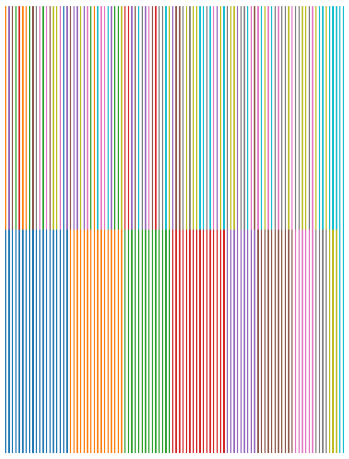
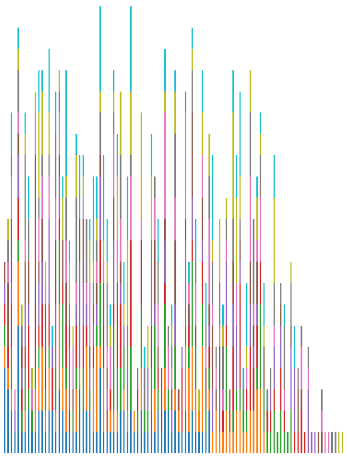
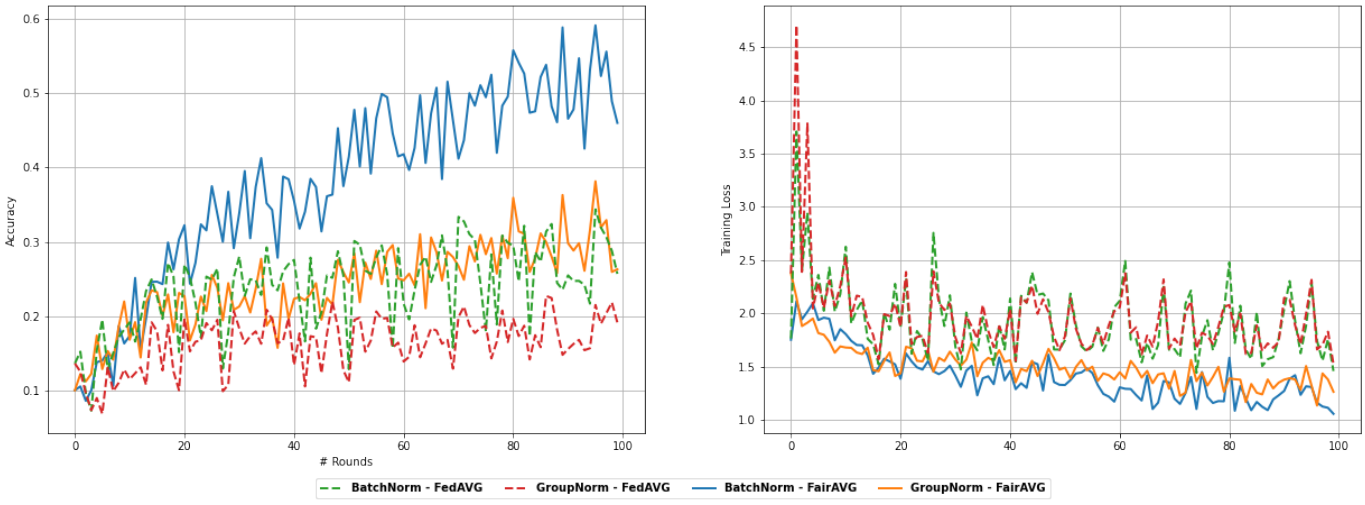


Fig. 8: non-IID Unbalanced



These figures show how the data is distributed among different clients in the three different data distribution cases we have considered. For each bar (client), the colors represent the image category (10 in total) and the the number of images is expressed through the height of the bars.

Fig. 9: FairAVG vs FedAVG

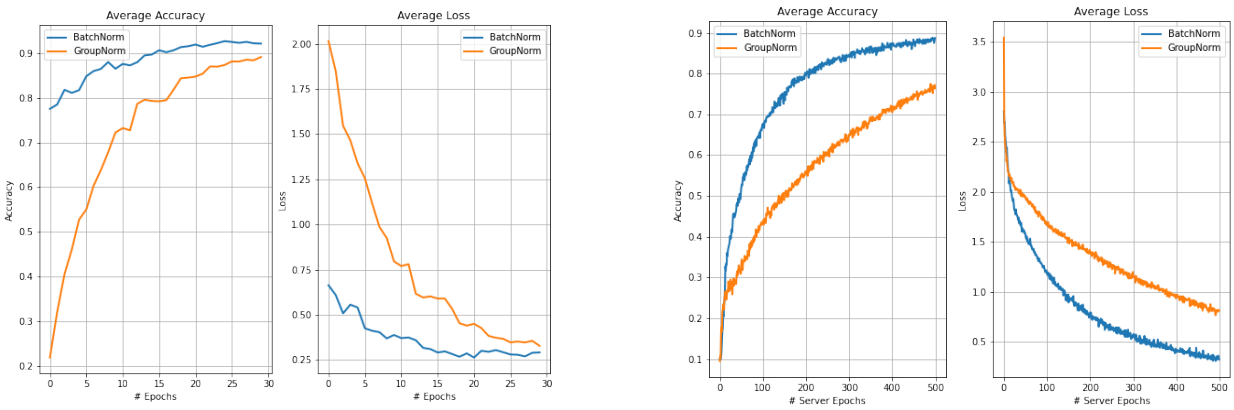


FairAVG increase significantly the the accuracy (mainly when using Batch Normalization) compared to FedAVG; also the loss curve is more stable.

Fig. 10: Baseline vs. FedAVG

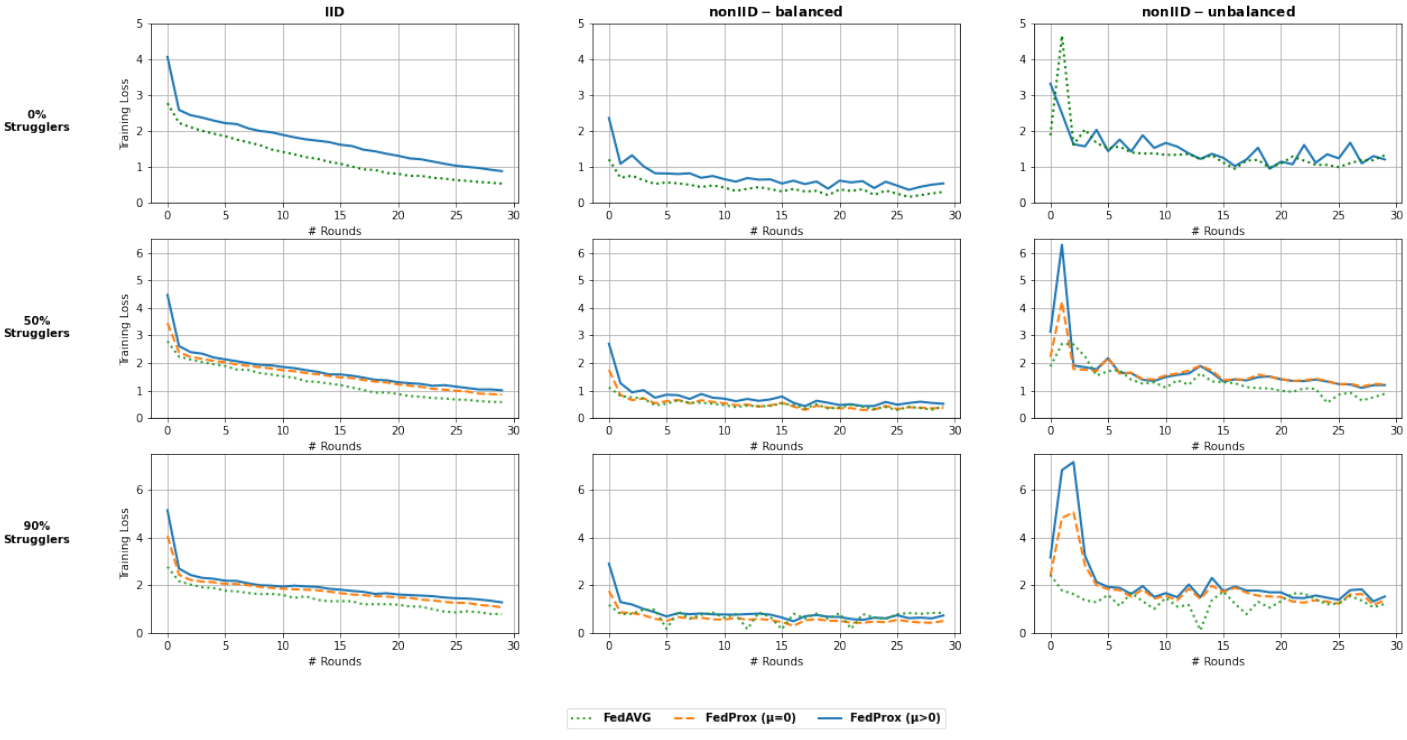
(a) Centralized Baseline

(b) FedAVG IID case



Given enough epochs, we show that in the IID case FedAVG reaches convergence, with an accuracy comparable to the one of the centralized baseline.

Fig. 11: FedProx - training loss



We simulate different levels of systems heterogeneity by forcing 0%, 50%, and 90% devices to be the stragglers (dropped by FedAvg). We show that in our case FedProx doesn't improve convergence compared to FedAVG and that the proximal term μ has no meaningful impact.