Understanding the Main Challenges of Federated Learning

1st Abdollah Foroutan
Politecnico di Torino
Torino, Italy
S300302@studenti.polito.it

2nd Ivan Contreras
Politecnico di Torino
Torino, Italy
S301962@studenti.polito.it

3rd Fatemeh Ahmadvand
Politecnico di Torino
Torino, Italy
S301384@studenti.polito.it

***Abstract*—Federated learning has introduced a new distributed setting to machine learning over remote devices. In the early versions of this algorithm, a global server takes the lead in processing data on different local clients. The initial algorithm does not consider the effects of statistical data heterogeneity, systems heterogeneity and gradient inversion attack. In this paper, we first create a centralized resnet-50 model as the basis of the comparison. Second, we simulate some introduced methods to improve the mentioned effects on the federated baseline and address the primary challenges of FL, consisting of Privacy and data system heterogeneity.Our project is available on: https://gitlab.com/abdollah.frootan/main-challenges-of-federated-learning**

I. Problem overview

The growing use of IoT and intelligent devices opened a new window to accurate and noteworthy data. However, due to privacy-preserving personal data, we cannot collect them on a centralised server to process. To overcome this issue, McMahan, in 2017 [1] introduced the state of the art of Federated Learning. Federated decompose and share training process across devices without sending raw data to Data Processor. In other words, each client processes its data and communicates the result with the server, and the server trains its global model using this information. As a general approach initial version of Federated Learning by McMahan [1] was a powerful idea, but various empirical subjects remain open to debate. Choosing an accurate deep model for clients, dealing with system heterogeneity on each client, statistical data heterogeneity, and client-server communication security are the main challenges we debate in this paper.

II. Related works

A. State of The Art

McMahan identified decentralised training as a significant research area in the machine learning. They create an algorithm called **FedAVG** [1], which performs local stochastic gradient descent on each client and sends local gradients after tunable rounds of local iteration to a central server as the aggregator. Indeed the paper discusses non-IID data and different models as the core of the deep network; the method did not work as expected in a real-world environment. [2] [3]

B. Deep Residual Network

Surprisingly, accuracy decreases marginally when we increase the deepness of a neural network by adding to the number of layers. The problem does not originate from Overfitting because both accuracies on both train and test set to experience a fall. Authors called this problem **vanishing/exploding** gradient. They suggest the residual block concept. Each block forwards the output of activation of a layer to a Further layer by skipping some layers in between. Consequently, **ResNet** created by connecting these blocks. [4]

C. Normalization

Batch Normalization frequently used as the foundation of state-of-the-art algorithms. In our area **FedBN [5]** and **ResNet** applied BN layer as default normalizer in their architecture. Nonetheless, BN errors rise when batch size is small, a common problem in federated learning. Yuksin in 2018 proposed a simple alternative named **Group Normalization** [6]. It divides channels into groups and normalises Features inside each group. GN is independent of the batch size and consistently outperforms BN when the Batch size is small.

D. Statistical Heterogeneity and Real-world data

The clients in the real world are smart devices like mobile phones. Naturally, because of differences in characteristics of devices, amount of usage and some other factors, data distribution is significantly different across the clients—several authors discuss non-identical client data partitions. Tzu-Ming 2019 measured the effect of distribution on visual classification [7]. They utilize concept of **Dirichlet** distribution to generate synthetic non-identical data. Authors add α **Coefficient** to data generator;which varying α from 0 to ∞ generate different class distribution. Furthermore, Sai in 2021 [8], shows **Client Drift** notion when data is heterogeneous(non-iid). Data drift results in more communication round, unstable and slow convergence. They developed the **SCAFFOLD** technique, which estimates the updated direction of the serer model and then updates ate direction for each client to correct the Drift.

E. System Heterogeneity and real-world clients

State-of-the-art federated learning models only evaluate the convergence rate on small CNNs or suppose that all clients have the same and enough computational power, which is

a wrong assumption in the real world. **FedGKT** [9] was a proposed model that split prominent CNN between edge and server; Concurrently, combined by FL routine.

F. Privacy

Yangsibo Huang, in 2021 [10], demonstrates that it is possible to reconstruct the clients' Private data using communications weights by **gradient inversion attacks**. They indicate there are many wrong assumptions about federated learning privacy. Hence, recovery of the private users' data is possible, even though we have a large batch size of images, many steps of local training, and a sizeable deep network like resnet-50.

III. Proposed approach

A. Centralized Model

Owing to evaluate the performance of the further Federated model, we need to define an upper bound. Generally, Federated algorithms are compared with their centralized situation. Hence, we choose *ResNet-50* to implement a centralized training procedure.We used modified ResNet-50 [4] based on **Deeper Bottleneck Architectures**. At this implementation instead of skipping 2 layers,each residual block skip output of *relu* activation function to 3 layers contain *1 x 1* convolutions layer as bottleneck. Because it increases depth while having fewer parameters. Besides, it has been proven *GN* have better handle non-iid data than BN in federated scenario [11]. To consider this comparison,our *resnet-50* has possibility to choose its normalizer with an argument.Meanwhile, the optimizer is stochastic gradient decent*(SGD)* proposed in state of the art.

B. Federated Baseline

second stage of our exploration is turning the centralised model into a federated client-server layout. There is two possible;

1) IID data: means distribution does not fluctuated and items in samples taken from same probability distributions. Technically,

- Each $x^i \longrightarrow D$ *(identically distributed)*
- $\forall i \neq p(x^i)p(x^j)$ *(independently distributed)*

In practice, due to having IID data, we uniformly drew samples with replacements from the entire training dataset and the same probability for each label. It should be considered that this situation is unrealistic in real-world federated scenarios.

2) Non-IID data: means we have to violates *Independence* and *Identicalness*. Most empirical works on generating synthetic non-iid data sets focused on unbalanced mini-batch sizes and identical distributions based on existing labels. Accordingly

- Data on each node should be generated in different distribution $x_t \sim P_t$
- The number of data samples on each client, k_t, can vary significantly

To address this matter, we applied *Dirichlet* Sampling to generate non-identical distribution at each node, which sample points with different label probability at each node. In addition, we assign the other numbers of data to clients considering the unbalanced mini-batch size.

Our federated baseline simulate main characteristics of *FedAvg* [1] by *dictionary* data structure. In particular, at each global training round E, we assign randomly selected clients k from all set of Clients K to *Values* of a dictionary; which the *key* is their S_t server. Meanwhile, we assign a randomly sampled set of data point b as *Values* of each client k *key* locally.

C. Statistical Heterogeneity

Investigate the *FedAvg* make it clear to us this algorithm converge slowly, specially when we have statistical heterogeneity(*non-iid* data). Our proposed solution to deal with this topic is using *SCAFFOLD* [8] algorithm. *SCAFFOLD* looking for slow convergence in concept called "client drift". It simply points out a computed global average of optima is not equivalent to optima of average f. The main idea is to control the variates which need additional correction terms at the step of local optimization update (SGD here):

- $y_i = y_i - \eta(g_i(y_i)) + C - c_i$
- C is guessed direction of server update
- c_i is guessed direction of client i update

where correction guess is exactly computed previous round update direction for each client or server. Technically, *scaffold* can choose larger learning rate η and achieve the optima faster in comparison to non-corrected *FedAvg*.

D. System Heterogeneity

System heterogeneity results from the difference between the computational power of physical devices. We implement *FedGKT* [9] as proposed method to reduce this effect. FedGKT works based on two main notions:

- *Split learning:* It is conceivable to cut a large deep network and train each part on different devices [12]. Here we divide our resnet-50 into a resnet-8 and resnet-49; which run on clients and the server respectively. At each client, we have a *feature extractor* that sends local trained weight values as input to the training server. Additionally, we have a classifier that can calculate predicted labels locally.
- *knowledge distillation:* The *softmax* layer of a deep NN generate one-hot encoding and soften our probabilities into 0 and 1, for which 1 is our prediction. On the other hand, The KD-loss (KD divergence) calculate a probabilistic output from softmax by changing a temperature T hyper-parameter, called soft label Z. [13] This method is used to transfer learned knowledge from one distilled network to the other one.*(teacher to student network).*

we reformulate the loss function of clients and server as below:

- $l_s = l_{CE} + \sum_{k=1}^{K} l_{KD}(Z_s, Z_{c^k})$ *(server loss)*
- $l_c^k = l_{CE} + l_{KD}(Z_s, Z_{c^k})$ *(client loss)*

KD loss can minimise the gap between grand truth and soft labels. After local training, the feature extractor sends the

output tensor to the server, and the server calculates the loss. Then the server sends its soft labels to the client, and the client calculates its loss. So clients and servers boost the performance of each other. The final model contains a feature extractor and a shared service model.

E. Attack simulation

Federated models without any extra measurement, such as encryption, have the potential for attack. At this step, we used a Gradient Inversion attack to reconstruct the user image batch by sniffing and inverting the gradient at each communication round. Thus, we emulate the attack by loading the model and getting the gradient using the difference between the original weights, and the weights are feeding the model with the image batch.

IV. Results

A. Data-set

The main experimented *data-set* in all missions of project is *CIFAR-10* which is an image classification problem with 10 classes. Since most federated learning real-world problems deal with text and images, this Multi-class source is a feasible choice to simulate a federated environment. After the standard augmentation pipeline, the data-loader sends 64 batch sizes of the image to our model, and all images are resized into $32, 32, 3$. Fig 1. represent a sample set of data.

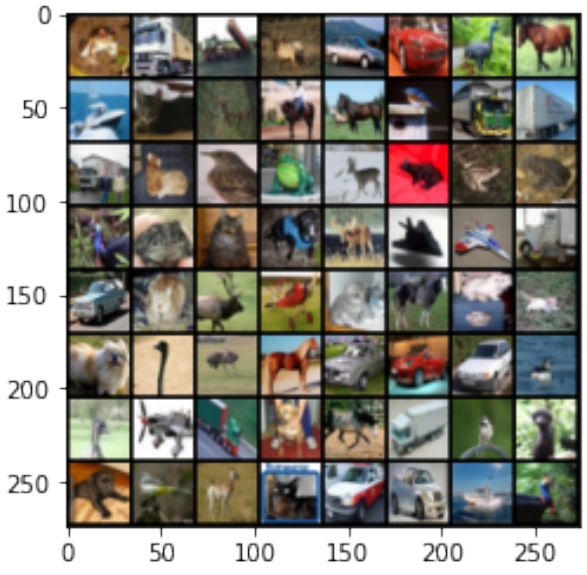

Fig. 1. Sample batch of CIFAR-10 dataset

B. Upper bound Model

We implemented centralised *resnet-50* model with BN and GN (*with 2 groups*) . The model optimizer is *SGD* and loss calculated by *cross entropy* function.The accuracy after 200 epochs, is visible in Table I and Fig 2. Predictably, model which uses group normalization performs better than the batch norm in case of accuracy and convergence. Concerning the massive amount of trainable parameters, the model has considerable test accuracy and shows how many residual blocks can be effective on the vanishing gradient.

TABLE I
CENTRALIZED RESNET-50 ACCURACY

Normalization layer	Centralized Accuracy	Number of Parameters
BN	0.66	23,520,842
GN	0.74	23,520,842

TABLE II
FEDAVG WITH DIFFERENT NORMALIZATION LAYER AND DISTRIBUTION

Distribution	Normalization layer	Accuracy
IID	BN	0.62
Non-IID	BN	0.25
IID	GN	0.50
Non-IID	GN	0.49

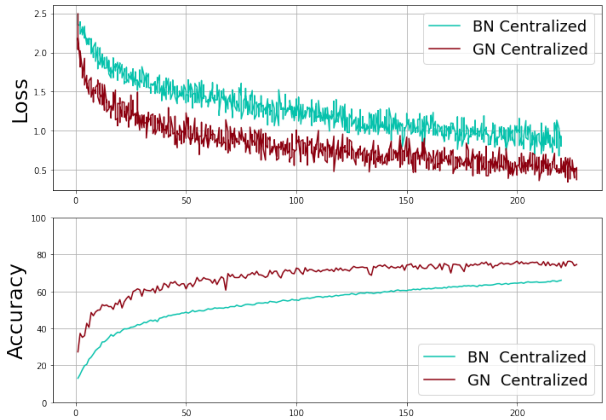

Fig. 2. Comparing BN and GN (*with 2 group*) in centralized model

C. FedAVG

We have implemented a federated baseline the same as centralized model configurations. We defined 100 clients that, in each round randomly picked to contribute to calculations. Each client in the IID setting has 128 images, and in the non-iid setting, it is associated with a random number of images.The experimental result are shown in Table II and Figure 3 compared four result of FedAvg with its different settings. Based on 3 we see IID distribution has better accuracy than the non-iid setting and shows us the effect of data heterogeneity in federated learning. There are two remarkable results in the non-IID setting:

- Group Normalization is substantially more accurate that batch normalization because of small batch sizes in clients.
- With GN we have smaller variance in error over multiple training process.

D. SCAFFOLD

In the previous section, FedAvg shows that it can perform well on non-iid data. We should attend that our experiment

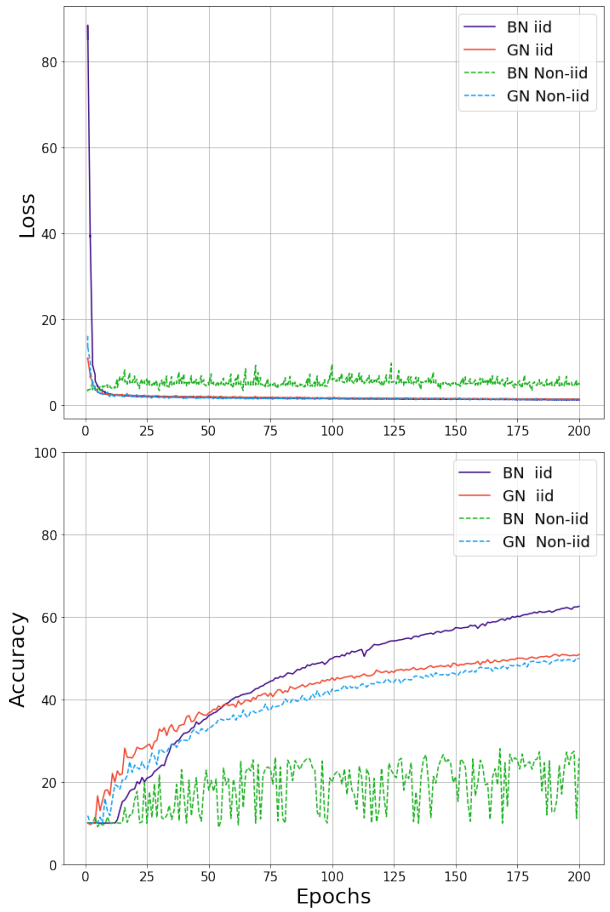

Fig. 3. Comparing BN and GN (*with 2 group*) in FedAvg which is our federated baseline

has performed with only one local training step and Not very heterogeneous data. Accordingly, we provide a situation in that each client train $E = 10$ times locally. After, we change the a Dirichlet's parameter to have a rather heterogeneous distribution among clients. Fig 4 imagine data distribution before and after increasing a from $(1, 100)$ to $(1, 2)$. With these changes on non-iid data, SCAFFOLD shows its power. Due to the RAM-demanding training process, this algorithm was trained by 50 clients, 10 local epochs and 20 global epochs. (totally 200). At this setting, FedAvg gets stoke in local minimums because of the effect of client drift, even with different learning rates, whereas SCAFFOLD normally converges during the training. In both BN and GN normalization, SCAFFOLD experienced a stabler, faster and more precise. Surprisingly, We had better accuracy with BN; However, I believe the low number of epochs caused this issue.

E. FedGKT

According to our computational resources, we set the global epochs to 10 times which simulate the communication between resnet-8 of clients and resnet-49 of the server. Communication rounds were exactly similar to FedAvg in the third section. Table IV and Fig 6 have summarised the final output. As we

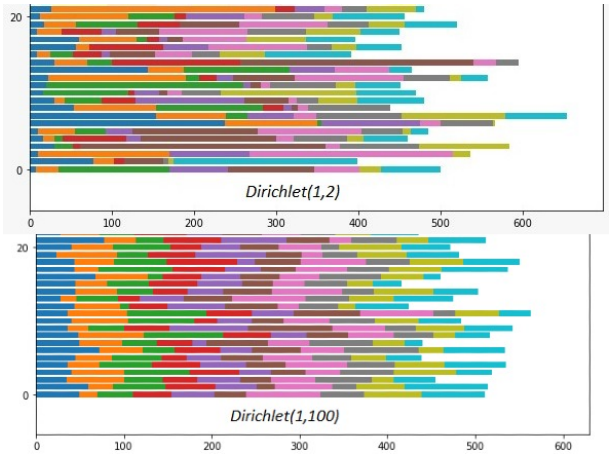

Fig. 4. Effect of increasing heterogeneity on data distribution. (1,2) has higher amount of difference among clients

TABLE III
COMPARISON BETWEEN FEDAVG AND SCAFFOLD ON NON-IID HIGH HETEROGENEOUS CLIENTS

Method	Normalization layer	Accuracy
SCAFFOLD	BN	0.172
FedAvg	BN	0.111
SCAFFOLD	GN	0.132
FedAvg	GN	0.101

expect, the iid setting,is rare in real-world experiments, gets the highest BN and GN scores. FedGKT brings us some pros:

- It is worth noting that this high output accuracy was achieved in only 10 iterations compared to 200 global communications of FedAvg. Lower communication rounds mean The users' devices involve fewer racecourses during the training process and invade less time for computations. Furthermore, it's more secure; with more occasional communications, the possibility of the Gradient Inversiont attack reduces.
- Instead of running a vast resnet-50 on clients, we run a small resnet-8 for each user. This small network is trainable on most IoT devices, even with a less powerful processor.

F. Grad Attack

To simulate the GI attack, we used our FedAvg with resnet-50 pre-trained in the second step. Behalf, we observe decreasing batch size can increase our recovery quality. Likewise, with

TABLE IV
FEDGKT WITH DIFFERENT NORMALIZATION LAYER AND DISTRIBUTIONS ON IID AND NON-IID DATA

Distribution	Normalization layer	Accuracy
IID	BN	0.699
Non-IID	BN	0.530
IID	GN	0.726
Non-IID	GN	0.360

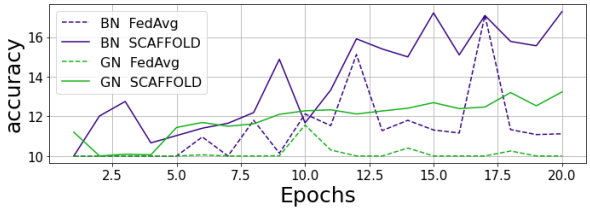

Fig. 5. Compare accuracy of FedAvg and SCAFFOLD in 20 epochs

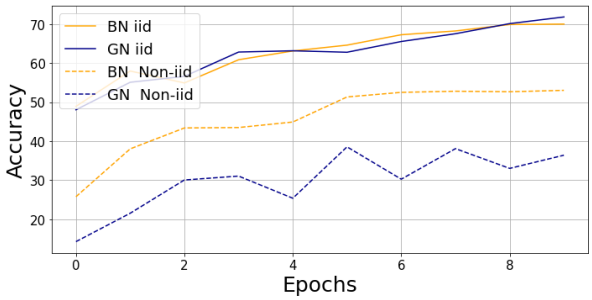

Fig. 6. Compare accuracy of FedGKT with different setting

more iterations Chance of recovery rise periodically. Fig **??** shows our original input batch and the recovered output. The quality of the recovered image is dramatically lower than the original one. However, considerable knowledge and general concept can still be extracted from this, and it's enough for a User privacy violation. Our model uses 32*32 images, and in actual cases, users have higher quality images, and with higher quality images, we can extract more accurate knowledge from the sniffed image.

V. Conclusions and Discussions

This work proposes a gradual process of constructing a federated model from a centralized one in different steps. Such that, the final developed Federated model performs noticeable functionality contrast by similar typical central training. We discuss the main challenges of federated learning, such as statistical and system heterogeneity, performance and privacy. Overall, obtained conclusions point out that Federated learning can be one of the excellent alternatives for data processors to overcome users' privacy policies like GDPR. Finally, there are some perspectives which are noticeable in this project:

- FedAvg method has good performance on iid data, which is not realistic, so we have to tune it with algorithms like FedGKT and Scaffold to perform well in real-world cases.
- , indeed, group normalization usually performs better than batch normalization, but it is not extensible to all client-server settings.
- Scaffold has major accuracy when we have a huge amount of client drift compared to pure FedAvg. Especially in real cases in which cultural and personal aspects mirror data.

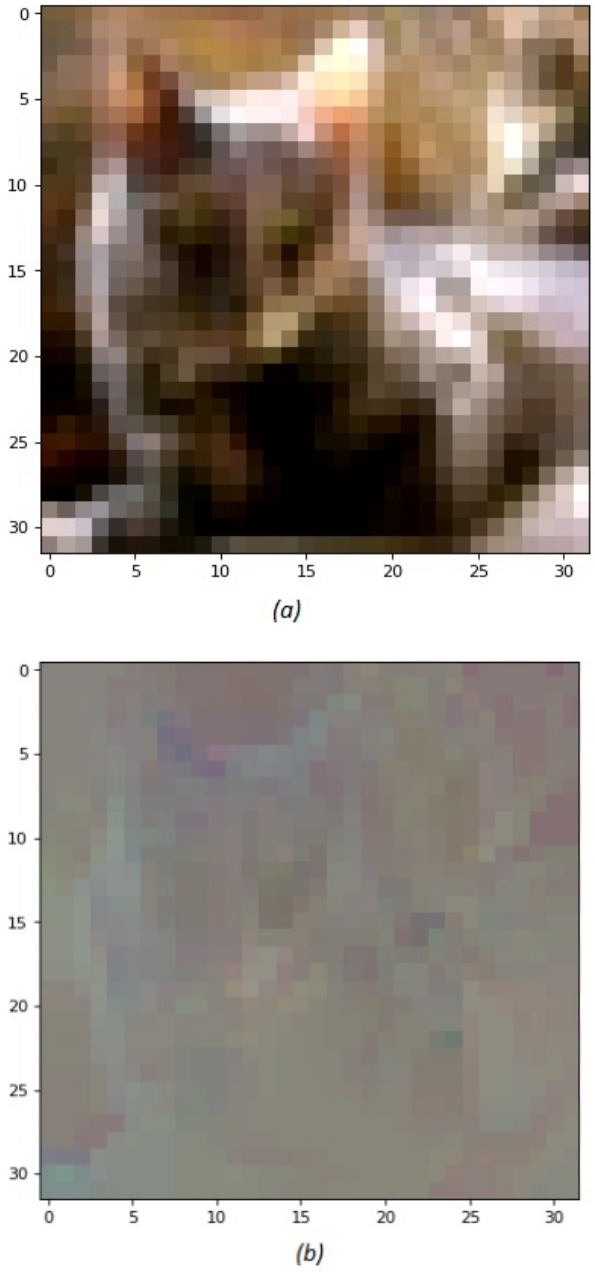

Fig. 7. (a) is user data and (b) is reconstructed image by GI attack

- We have to take care of data sniffers through privacy measures. without them, our communications are not safe enough.

Although we obtained acceptable results, we are aware that most of the simulated challenges were simplified cases, and the results can change slightly on a larger scale. Furthermore, there are a lot of novel works proposed recently in this field that solve other challenges we never discuss here.

References

[1] R. S. H. Brendan McMahan, Eider Moore Daniel, *Communication-Efficient Learning of Deep Networks from Decentralized Data*, vol. 1. Google Inc, 2017.

[2] P. Kairouz, *Federated Learning: Challenges, Methods, and Future Directions*, vol. 1. IEEE, 2019.

[3] T. Li, *Federated Learning: Challenges, Methods, and Future Directions*, vol. 1. IEEE, 2019.

[4] X. Z. Kaiming He, *Deep Residual Learning for Image Recognition*, vol. 1. ICCV, 2015.

[5] X. L, *FEDBN: FEDERATED LEARNING ON NON-IID FEATURES VIA LOCAL BATCH NORMALIZATION*, vol. 1. ICLR, 2021.

[6] K. H. Yuxin Wu, *Group Normalization*, vol. 1. ECCV, 2018.

[7] T.-M. H. Hsu, *Measuring the Effects of Non-Identical Data Distribution for Federated Visual Classification*, vol. 1. Archive, 2019.

[8] S. P. Karimireddy, *SCAFFOLD: Stochastic Controlled Averaging for Federated Learning*, vol. 1. Archive, 2021.

[9] M. A. He, Chaoyang and S. Avestimehr., *"Group knowledge transfer: Federated learning of large CNNs at the edge."*, vol. 1. Arxive, 2020.

[10] Z. S. K. L. S. A. Yangsibo Huang, Samyak Gupta, *Evaluating Gradient Inversion Attacks and Defenses in Federated Learning*, vol. 1. Arxive, 2021.

[11] K. Hsieh, *al."The non-iid data quagmire of decentralized machine learning."*, vol. 1. ICML, 2021.

[12] O. G. T. S. e. a. Vepakomma, P., *Split learning for health: Distributed deep learning without sharing raw patient data."*, vol. 1. arxive, 2018.

[13] O. V. J. D. Hinton, G., *Distilling the knowledge in a neural network*, vol. 1. ICML, 2015.