Privacy-Preserving Hyperparameter Tuning in Federated Learning Setting

Simon Perriard
Cybersecurity Master semester 4, EPFL
12 ECTS Semester project, Spring 2021
Supervised by Sinem Sav
LDS (Prof. Jean-Pierre Hubaux)

***Abstract*—This project explores the possibility to deduce hyperparameters for FedAvg [1] based on a gridsearch done by clients in an individual fashion with different settings, i.e., iid and non-iid with several skews (quantity, label, feature). It also presents heuristics for each skew type to find the global hyperparameters for FedAvg.**

I. Introduction

In machine learning, supervised learning is a great way to solve complex problems such as medical images classification, text translation or intrusion detection based on user's behaviour for example. One of the pitfalls of this method is that it requires a lot of training data to build a robust and accurate model. In a real life scenario, multiple companies may have an interest in training a collaborative model, but without sharing their sensitive data. A way to train a privacy-preserving model that uses data from all the participants is to encrypt the data or the model and apply a federated optimization method such as FedAvg [9] in encrypted domain. However, this solution adds a consequent computation and communication time overhead. In machine learning, it is important to choose optimal hyperparameters to get the best possible results. Grid search is a robust way to perform hyperparameter tuning, but it is a costly process. Furthermore, moving this task in a federated and privacy-preserving setting adds more communication and computational overhead on top of an already heavy process, increasing the hyperparameter optimization time by orders of magnitude. The goal of this project is to find a way to tune hyperparameters in a privacy-preserving and federated setting without relying on high-communication or computation frameworks by finding a correlation between the hyperparameters found with FedAvg and the ones found by individual clients search.

The project was structured as follows:

1) IID setting: the dataset is evenly distributed among all the participants.
2) Non-IID setting: the dataset is distributed with a skew as described in [7]. We chose three different unbalanced setups, i.e., quantity, label, and feature skews with different magnitudes.

Moreover, we conducted several experiments and approximations to determine whether ADAM optimizer [2] could be efficiently used in a privacy-preserving setting by relying on homomorphic encryption. These experiments are detailed in Appendix A.

II. Experiments setting

A. Datasets

We used four datasets for the experiments: MNIST [3], extended MNIST or EMNIST [4], the cropped version of Street View House Number (SVHN_CROPPED) [5] and CIFAR-10 [6].

B. Models

The model for MNIST is Conv2D(filters=32, kernel=5)-activation-AvgPool(window_size=3)-Conv2D(32,5)-activation-AvgPool(2)-Flatten-Dense(200)-activation-Dense(10)-softmax.

We used conv2d architecture detailed in [7] for EMNIST and SVHN. The structure for the architecture is Conv2D(16,5)-activation-AvgPool(2)-Conv2D(6,5)-activation-AvgPool(2)-Flatten-Dense(120)-activation-Dense(84)-activation-Dense(62 for EMNIST, 10 for SVHN_CROPPED)-softmax.

We used a variant of the model proposed in a Machine Learning Mastery blog post[1] for CIFAR10. The structure for the architecture is built with three blocks Conv2D(64+32*i)-activation-Conv2D(64+32*i)-activation-AvgPool(2)-Dropout(0.2+0.1*i) for $i \in [0, 1, 2]$ followed by Flatten-Dense(128)-activation-Dropout(0.5)-Dense(10)-softmax.

All the experiments are conducted with standard stochastic gradient descent (SGD) as the optimizer.

III. IID Setting

Firstly, we will discuss the relevance of applying a federated optimization method, then we will see a simple method to efficiently train our models.

Since each client receives the same amount of uniformly distributed data, we can apply the following heuristic reasoning. We can argue that under IID setting, every client would converge towards the same set of hyperparameters, with a small variance in the results. The experiments that we have conducted support this heuristic.

[1] https://machinelearningmastery.com/how-to-develop-a-cnn-from-scratch-for-cifar-10-photo-classification/

It may not be relevant to apply a federated optimization technique such as FedAvg [1] when the data is IID since all the clients will individually converge towards the same hyperparameters. However, we could take advantage of the hyperparameters found by other participants to speedup or conduct a better hyperparameter tuning.

This experiment follows the following steps, for each dataset and for 1, 10, 20 and 50 clients:

1) Perform a gridsearch on each client and output the best hyperparameters.
2) Average the best hyperparameters, retrain each client and test their accuracy on the test dataset with the averaged parameters.
3) Determine the best activation function between `relu`, `tanh` and `sigmoid`.
4) Replace the best activation function by its least-squares approximated alternative and determine the best interval for this approximation through gridsearch. The reason for using approximated activation functions is that, when moving to a privacy-preserving setting, the underlying homomorphic encryption scheme does not support non-polynomial functions.

With this experiment, we show that averaging the hyperparameters of the clients is feasible under IID setting, although we can observe a loss of accuracy when the number of clients is growing and when we switch from "original" activation functions to their approximated version. Nonetheless, this loss of accuracy was expected because we used approximated activation functions and they are expected to perform less well than their non-polynomial counterparts.

Since we cannot do a simple average between the best intervals, the best one will be the most frequent best interval amongst the participants.

IV. Non-IID Setting

Due to a phenomenon called *client drift* (Fig. 2 [7]) in federated learning, considering non-IID data distribution amongst the clients is an important matter. In this context, we will certainly encounter different distribution skews depending on the participants. We target three skews in this project: quantity, label, feature.

- Quantity skew: Each participant has different amounts of training samples.
- Label skew: Some participants are highly specialized in some of the target classes if their environment is prone to it, e.g., there may be more lung cancers in a poor mining area than in a clean countryside.
- Feature skew: The distribution of the features of each participant is different. As an example, depending on the geographical location of each participant, one may have more black dogs, another one more white dogs and, a last one more brown dogs, assuming that they want to train a model that classifies pictures of animals based on their local fauna, there will be a feature skew in this situation.

The following experiments were run for each dataset and for 10 and 20 clients. We performed a gridsearch amongst individual clients (decentralized) and amongst clients with FedAvg (fedavg). The goal is to find a link between decentralized best hyperparameters and fedavg ones that best captures the correlation between them.

A. FedAvg

We used the FedAvg aggregation algorithm to federate the clients. The vanilla version is described in [1]. However, we used the FedJAX[2] library to train our models and it implements a generalized and better version described in [8].

Server side parameters are server learning rate, server momentum and number of rounds, which are denoted as server_lr, server_mom and rounds respectively.

Client side parameters are client learning rate, client momentum, batch size and epochs per round, which are denoted as client_lr, client_mom, bs and epr respectively.

B. Quantity skew

Due to time constraints number of epochs (decentralized), rounds, epr, server_lr and server_mom were fixed to 60, 30, 2, $\sqrt{0.001}$ and 0.9 respectively. Skews are from a Dirichlet[3] distribution with parameters $0.1, 0.4, 1.0, 2.0, 5.0$, a small value means a large skew.

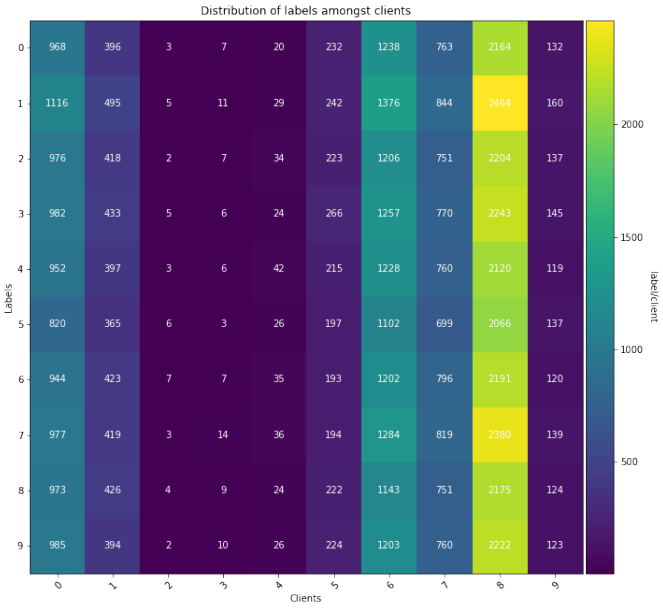

Fig. 1. MNIST samples distribution for 10 clients with a quantity skew of *Dir(0.4)*

The steps for the experiment are:

1) Perform gridsearch (client_lr, client_mom, bs) on both decentralized and FedAvg.
2) Output best hyperparameters and find a heuristic to link them.

[2]https://github.com/google/fedjax
[3]https://en.wikipedia.org/wiki/Dirichlet_distribution

One heuristic for deriving FedAvg hyperparameters from decentralized ones relying on our experimental results, is given in Eq. 1, Eq. 2 and Eq. 3. From the decentralized experiment, $ratio_i$ is the ratio of samples client i received to train on, lr_i is the best learning rate of client i, mom_i is the best momentum of client i, bs_i is the best batch size of client i, val_acc_i is the best validation accuracy of client i and $best\_acc$ is $\max_i(val\_acc_i)$. We also chose the gridsearch parameters such that lr_i and $mom_i < 1$

$$global\_lr = \sum_{i \in clients} \frac{ratio_i * lr_i * (1 - lr_i)}{10} * (1 - (best\_acc - val\_acc_i)) \quad (1)$$

$$global\_mom = \sum_{i \in clients} \min(\frac{1.0}{\#parties}, ratio_i * mom_i * (1 - mom_i) * (1 - (best\_acc - val\_acc_i)) * 10) \quad (2)$$

$$global\_bs = \sum_{i \in clients} ratio_i * bs_i * val\_acc_i \quad (3)$$

As we observe no trend or relation for approximated activation functions interval search, we suggest a seperate gridsearch for the intervals to be run in the federated setting, or take the most common interval.

Then, we compared the performance of the model using FedAvg with gridsearch-found best hyperparameters and with the ones found with the method described above. Results are encouraging since we could observe a slight loss in accuracy of around 3% on less complex datasets (MNIST, EMNIST, SVHN), and around 10% for a more complex CIFAR-10 dataset, see Fig. 2 and Fig. 3.

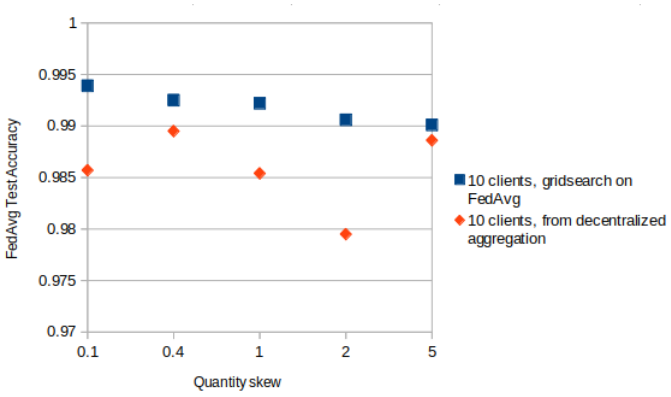

Fig. 2. Test accuracies on MNIST with FedAvg. Blue shows the test accuracy with the hyperparameters found by FedAvg gridsearch and orange shows the test accuracy found by aggregation of decentralized gridsearch results under the method described for quantity skew.

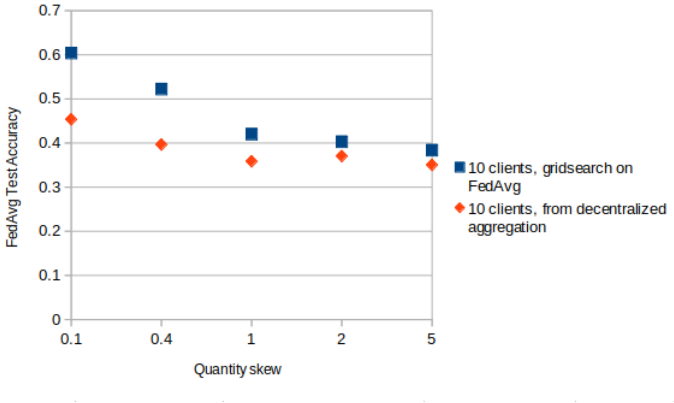

Fig. 3. Test accuracies on CIFAR-10 with FedAvg. Blue shows the test accuracy with the hyperparameters found by FedAvg gridsearch and orange shows the test accuracy found by aggregation of decentralized gridsearch results under the method described for quantity skew.

C. Label skew

Due to time constraints number of epochs (decentralized), rounds, epr, server_lr and server_mom were fixed to 60, 30, 2, $\sqrt{0.001}$ and 0.9 respectively. Skews are from a Dirichlet distribution with parameters $0.1, 1.0, 5.0$, a small value means a large skew. The steps for this experiment are the same as in IV-B.

One heuristic for deriving FedAvg hyperparameters from decentralized ones relying on our experimental results, is given in Eq. 4, Eq. 5 and Eq. 6.

$$global\_lr = \sum_{i \in clients} \frac{lr_i * (1 - lr_i)}{\#parties * 10} * (1 - (best\_acc - val\_acc_i)) \quad (4)$$

$$global\_mom = \sum_{i \in clients} \frac{mom_i}{\#parties} \quad (5)$$

$$global\_bs = \sum_{i \in clients} \frac{bs_i * val\_acc_i}{\#parties} \quad (6)$$

We did not find any good heuristic to deduce the best interval. The best we found is to take the majority of the best intervals found by the decentralized search, but it only works half of the time. We may also need to run a gridsearch in the federated setting to tune the interval.

After comparing the performance of the model using FedAvg with gridsearch-found best hyperparameters with the ones found with the method described above, results are encouraging for MNIST and CIFAR10 datasets (around 2% loss of accuracy). The method has still room for improvement as we can see in Fig. 7 and Fig. 6 (up to 40% loss of accuracy), but this first approach lets us think that future research on label skew could develop a method to get an overall better performance.

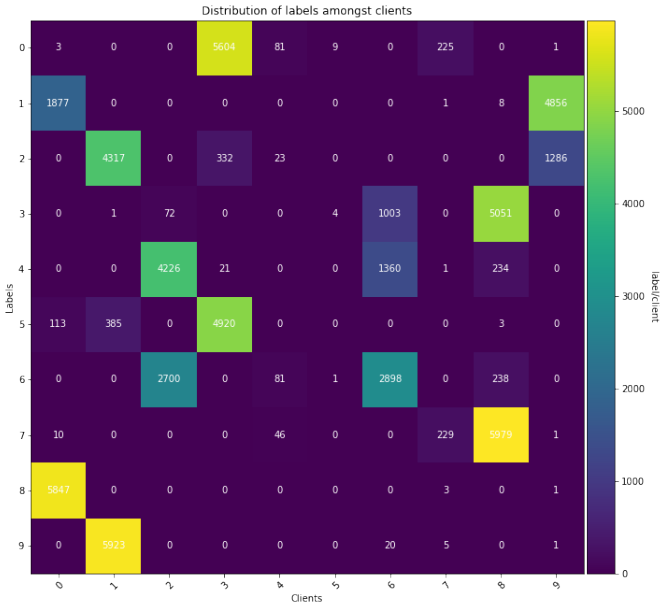

Fig. 4. MNIST samples distribution for 10 clients with a label skew of *Dir(0.1)*

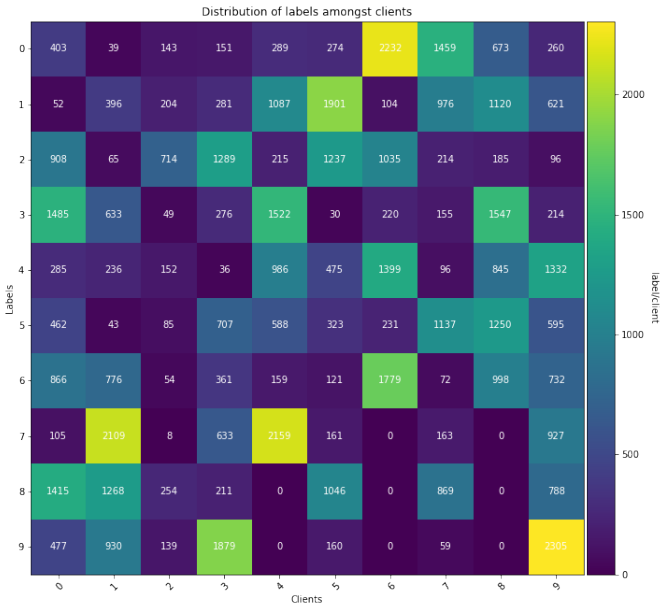

Fig. 5. MNIST samples distribution for 10 clients with a label skew of *Dir(1.0)*

D. Feature skew

Due to time constraints, EMNIST dataset was left aside and number of epochs (decentralized), rounds, epr, server_lr and server_mom were fixed to 60, 30, 2, $\sqrt{0.001}$ and 0.9 respectively. Skews are from a centered Gaussian distribution

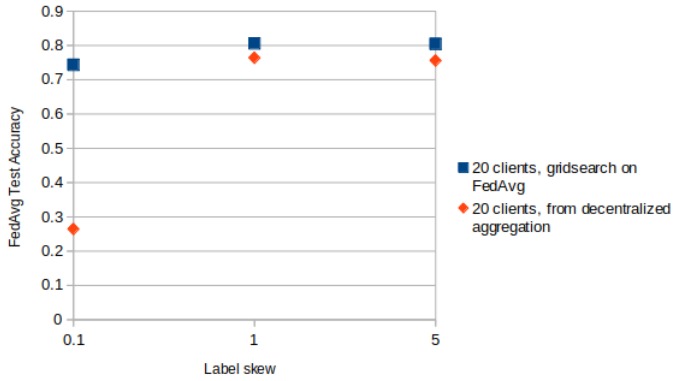

Fig. 6. Test accuracies on SVHN with FedAvg. Blue shows the test accuracy with the hyperparameters found by FedAvg gridsearch and orange shows the test accuracy found by aggregation of decentralized gridsearch results under the method described for label skew.

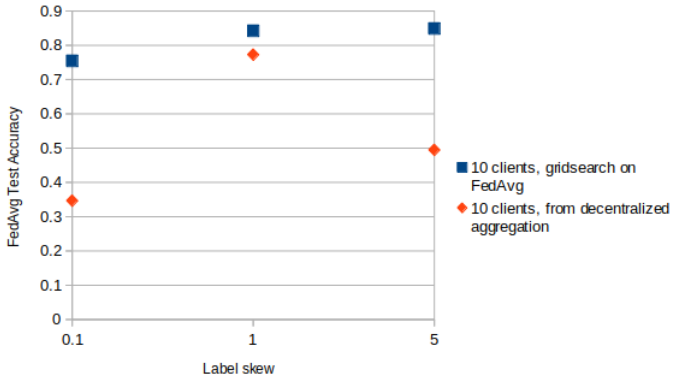

Fig. 7. Test accuracies on EMNIST with FedAvg. Blue shows the test accuracy with the hyperparameters found by FedAvg gridsearch and orange shows the test accuracy found by aggregation of decentralized gridsearch results under the method described for label skew.

of standard deviation $0.02 * \frac{i}{\#clients-1}$ and $0.1 * \frac{i}{\#clients-1}$ for each client i.

One heuristic for deriving FedAvg hyperparameters from decentralized ones relying on our experimental results, is given in Eq. 7, Eq. 8 and Eq. 9.

$$global\_lr = \sum_{i \in clients} \frac{lr_i * (1 - lr_i)}{\#parties * 10} * (1 - (best\_acc - val\_acc_i)) \quad (7)$$

$$global\_mom = \sum_{i \in clients} \min(\frac{1.0}{\#parties}, \frac{mom_i * (1 - mom_i)}{\#parties} * (1 - (best\_acc - val\_acc_i)) * 10) \quad (8)$$

$$global\_bs = \sum_{i \in clients} \frac{bs_i * val\_acc_i}{\#parties} \quad (9)$$

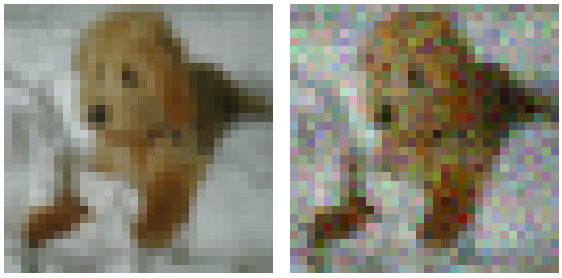

Fig. 8. Example of a feature skew on a CIFAR10 sample

As we observe no trend or relation for approximated activation functions interval search, we suggest a seperate gridsearch for the intervals to be run in the federated setting, or take the most common interval found by decentralized search.

After comparing the performance of the model using FedAvg with gridsearch-found best hyperparameters with the ones found with the method described above, we observe good response from the models. We found out that the loss of accuracy was globally around 3%, with an outlier at 20% accuracy loss on SVHN with 20 clients and a skew of parameter 0.02. Those results are also encouraging as we already get a performance comparable to FedAvg gridsearch performance from a simple heuristic for the feature skew.

E. Discussion

To improve the overall performance and find a better method to aggregate the decentralized results, we could perform the gridsearch on number of epochs, rounds, epr, server_lr and server_mom as well and with a bigger grid. With the current framework and libraries, however, this type of experiments would take weeks to run, which brings a challenge for federated learning settings. Another challenge concerns the approximated activation functions. We observe that the accuracy may suffer from a heavy loss when switching to the approximated activation functions (up to 60% less accuracy). A way to cope with this issue could be to run the intervals search with a different approximation or bigger intervals.

V. Conclusion

From our experimental results, we observe that decentralized hyperparameters tuning is around two times less time consuming than gridsearch in the federated setting while preserving the privacy against federated learning attacks as the gridsearch is done in local premises.

We showe that it is possible to find heuristics to infer FedAvg hyperparameters from decentralized gridsearch results. Inital heuristics give encouraging results without being astonishing since accuracy could decrease heavily depending on the skew. As such, we can balance the accuracy loss and privacy protection by relying on homomorphic encryption for the future applications, to perform the given heuristics across participants.

Since the training time gain at stake may be consequent, it would be interesting to conduct an extended research based on the doors opened by this project to build a general and reliable method to infer the FedAvg hyperparameters from decentralized ones.

Appendix A
ADAM Experiment

The ADAM experiment was conducted aside the IID Setting and its goal was to determine whether it is possible to decrease the bit precision of the floating point coefficient [9] in the update rule[4] without affecting the accuracy.

The final goal would be to allow ADAM to be put to work in a privacy-preserving setting with a lower precision in the coefficient mentioned above, thus reducing the computation time.

$$\frac{1}{\sqrt{\hat{v}_t + \epsilon}}$$

Fig. 9. Coefficient used in the ADAM update rule

We tried different precision levels that are $0, 1, 2, 4, 6, 8, 10$ and 12 allowed decimals on the coefficient 9 and full precision.

We could observe some small accuracy variations, around 1%, probably due to the stochastic nature of the optimization algorithm. This result means that we could use ADAM in a privacy-preserving setting with low precision approximation with a negligible accuracy loss.

References

[1] H. Brendan McMahan, Eider Moore, Daniel Ramage Seth Hampson, Blaise Agüera y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data", https://arxiv.org/pdf/1602.05629.pdf, February 2017.

[2] Diederik P. Kingma, Jimmy lei Ba, "ADAM: A Method for Stochastic Optimization", ICLR 2015, https://arxiv.org/pdf/1412.6980v9.pdf, revised January 2017.

[3] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition", Proceedings of the IEEE, vol. 86,no. 11, pp. 2278–2323, 1998

[4] Cohen, G., Afshar, S., Tapson, J., and Van Schaik, A. "Emnist: Extending mnist to handwritten letters". https://arxiv.org/pdf/1702.05373v2.pdf, In 2017 International Joint Conference on Neural Networks (IJCNN), pp. 2921–2926. IEEE, 2017

[5] Y. Netzer and T. Wang, "Reading digits in natural images with unsupervised feature learning", http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf, Nips, pp. 1–9, 2011

[6] A. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images", https://www.cs.toronto.edu/ kriz/learning-features-2009-TR.pdf, Ph.D. dissertation, University of Toronto, 2009

[7] Q. Li, Y. Diao, Q. Chen, B. He, "Federated Learning on Non-IID Data Silos: An Experimental Study", https://arxiv.org/pdf/2102.02079.pdf, 2021

[8] Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, H. Brendan McMahan, "Adaptive Federated Optimization", https://arxiv.org/pdf/2003.00295v3.pdf, 2020

[9] S. Sav, A. Pyrgelis, JR. Troncoso-Pastoriza, D. Froelicher, JP Bossuat, J. Sousa, and JP. Hubaux, "POSEIDON: Privacy-Preserving Federated Neural Network Learning", NDSS, 2021, https://arxiv.org/abs/2009.00349, 2020

[4]https://ruder.io/optimizing-gradient-descent/index.html#adam