Establishing the Parameters of a Decentralized Neural Machine Learning Model

Aline Ioste [1], Marcelo Finger[1]

[1]Department of Computer Science,
Institute of Mathematics and Statistics
University of São Paulo (IME-USP), Brazil

ioste@ime.usp.br,mfinger@ime.usp.br

Abstract. *The decentralized machine learning models face a bottleneck of high-cost communication. Trade-offs between communication and accuracy in decentralized learning have been addressed by theoretical approaches. Here we propose a new practical model that performs several local training operations before a communication round, choosing among several options. We show how to determine a configuration that dramatically reduces the communication burden between participant hosts, with a reduction in communication practice showing robust and accurate results both to IID and NON-IID data distributions.*

1. Introduction

Advances in deep learning in the last decade brought the need for more extensive datasets to solve real-world problems in machine learning [Paullada et al. 2021]. The restriction of centralized datasets for machine learning has been seen as the limiting factor for scientific progress [Sun et al. 2017]. Obtaining a dataset that can broadly generalize a problem remains a great challenge, which has promoted the need to join data from different owners without violating confidentiality constraints. In this setting, research for decentralized machine learning setup attracted a lot of attention, and several methods have been proposed for decentralized training. For example, there are proposals for decentralized stochastic gradient compression [Lian et al. 2017, Nedić et al. 2018, Koloskova et al. 2019] and for performing multiple local Stochastic Gradient Descent (SGD) steps before averaging [Zhang et al. 2016, McMahan et al. 2017, Stich 2018, Lin et al. 2018]. Perhaps the most widely studied decentralized models are those covered in the federated learning literature, with several alternative variations [McMahan et al. 2016, Konečnỳ et al. 2016, McMahan et al. 2017, Li et al. 2018, Zhao et al. 2018, Mohri et al. 2019, Yurochkin et al. 2019, Wang et al. 2020, Zhang et al. 2021].

A large number of important algorithmic questions remain open on the topic of real-world usability of decentralized schemes for machine learning [Kairouz et al. 2021]. The changes in the frequency of communication between distributed participants have been a primary bottleneck for federated learning. Trade-offs between communication and accuracy in federated learning have received a lot of interest from a modeling point of view [Han et al. 2021, Acharya et al. 2020, Tang et al. 2019]. Theoretical analysis shows that schemes that perform several local update steps before a communication round are significantly more challenging than those that use a single step. However, obtaining concrete insights from these theoretical works for communication reduction in practice is difficult because they do not account for the impact of the learning optimization algorithm. So it remains an open research challenge to propose practical training methods that address those problems [Kairouz et al. 2021].

1.1. Objective: A New Decentralized Training

The goal of this paper is to propose a departure from the traditional federated schema, in which all training steps are performed locally, and show how its parameters may be established. This new proposal aims to drastically mitigate the communication overload between the participants. As shown in Figure 1, for example, it assumes that:

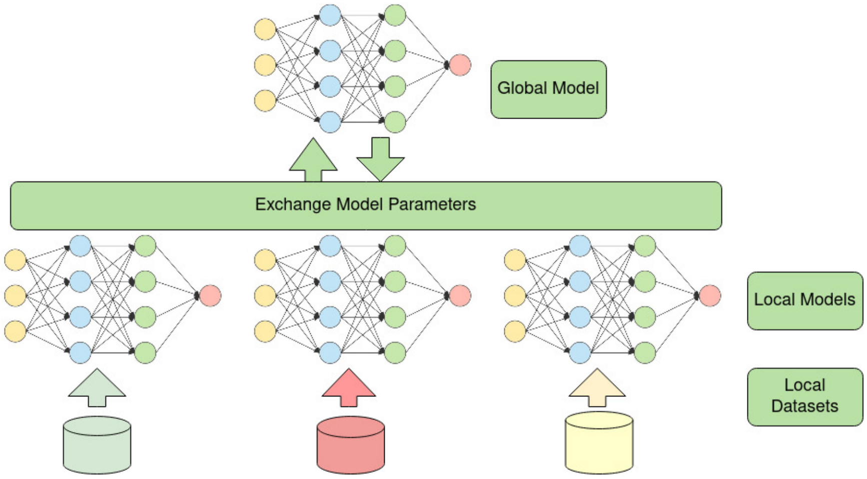

Figure 1. A distributed architecture in which models are fully trained locally before a communication round with the coordinator

1. The participants must completely execute the training of their local models, when each local model is assumed to be an instance of the same neural network model, differing only on the network weight values;
2. local models of participants must be able to reach a good generalization from their dataset;
3. local models generated by the participants may contain completely distinct generalizations;
4. after local training, each participant sends the learned model weights to a central coordinator;
5. the coordinator combines the weights received from the participants to compute a global model;
6. the coordinator sends to all the participants the combined model;
7. the participants may re-train their local models based on the global weights sent by the coordinator, sending them again to the coordinator;
8. thus the process of obtaining a global model is iterated, to approximate an idealized central model, i.e a model that would be trained with all the available data;
9. the coordinator decides when the process reaches a stable point;
10. all participants receive a copy of the combined model, thus acquiring the ability to process patterns unseen from their local data.

Note that no data travels on the communication infrastructure, only the models weights are sent over the network, from which no data is recoverable, thus maintaining data privacy restrictions.

This paper aims to demonstrate how such an architecture may be obtained in practice, and how the coordinator may combine the local weights, with the effect of drastically reducing communication overload.

The coordinator may choose to combine the weights in two ways. The first as it is done in most federated learning methods employs a convex combination. As another option, a non-convex combination may be preferable. We analyze both possibilities from a practical point of view Alternatively, the performance of the combined model by a practical process convex (current algorithms, average model) and a non-convex.

In the following we show how convex combination, in the form of average combination, leads to an unsatisfying global model; that is, we provide experimental evidence to support this claim. In the same way, we then show that using non-convex linear combination factors may achieve better results, providing evidence that there are forms of non-convex combination that succeed for a few iterations before decaying, and finally presenting a robust form of non-convex combination in which the combination of locally learned weights becomes a stable and accurate process of generating a global model even under highly unbalanced situations.

2. Convex Weight Combination

In decentralized learning, federated learning is the most popular approach [Konečnỳ et al. 2016, McMahan et al. 2017], which uses a convex scheme of immediate aggregation of local weights to obtain average weights in a federated model. Several variations of this technique were proposed, keeping the average computation to generate the federated weights [Li et al. 2018, Zhao et al. 2018, Mohri et al. 2019, Yurochkin et al. 2019, Wang et al. 2020].

There is a strong relationship between the convergence of the average model and the Independent and Identically Distributed (IID) hypothesis. Identically Distributed means that all items in the sample are taken from the same probability distribution, and Independent means that the sample items are all independent, they are not connected in any way.

Not independent and identically distributed (Non-IID) data in decentralized learning means differences exist between the training data of the local hosts. While various assumptions can be made on the datasets between distinct hosts being optimized, the most fundamental split is between assuming iid and non-iid data.

The federated averaging algorithm [McMahan et al. 2017] with dataset IID is the most common approach to optimization for federated learning, an adaption of local-update or parallel SGD. Each client runs some SGD steps locally, and then the updated local model weights $\overline{w}_i$ are averaged to form the updated global model on the coordinator, given by equation (1)

$$\overline{w}_i = \frac{1}{H}\sum_{h=1}^{H} w_i^h, \qquad 1 \leq i \leq W \tag{1}$$

where H is the total number of participant hosts, and W is the number of weights in the network.

We now test how the combination of weights using equation (1) for a simple feed-forward network performs the resulting global model, both for the balanced and unbalanced cases.

2.1. Balanced, iid Convex Combination Experiment

We tested the federated averaging algorithm on a scheme that executes complete training of local models before a communication round. We employ a three-layer feed-forward neural network with 1873 parameters (weights).

The dataset used in this test is the Covid-19 dataset that consists of laboratory tests for SARS-CoV-2 patients admitted to the Albert Einstein Hospital of São Paulo in 2020 [Mello 2020]. The goal here is to see if the results from several laboratory tests may predict the result for a given patient.

For this experiment, we took a balanced dataset with the following characteristics. The total number of COVID-19 training patients contains 1064 samples, 532 positive and 532 negative. The test dataset contains 558 samples, 279 positive and 279 negative samples.

We split the data into two owners, representing two participant hosts. Training dataset1 has 568 training dataset samples, with 284 positive samples and 284 negative samples, and training dataset2 contains the remaining samples.

Local models have the same neural network architecture. They are trained locally until they achieve local convergence. These models were trained locally until they reached an initial local performance of more than 90%, and after the complete training these models are sent to the decentralized process using the average model to combine, equation (1). This process is repeated by 30 rounds, as shown in Figure (2) A.

Note that the average model reaches 95.699% in the first communication round from weights of local models, and in the following rounds, the average model experiences a plunging performance, which can be called "unlearning", as it decays to 50%, which is equivalent to just random guessing.

The red line represents the model with the same architecture trained with the complete training dataset centralized. It reaches 98.029% accuracy on the test dataset.

2.2. Unballanced, non-iid Convex Combination Experiment

In this assumption, the Covid-19 dataset consists of a training dataset of 811 samples, 279 positive and 532 negative. The test dataset contains 558 samples, being positive 279 and negative 279 samples.

The split is made into two participant hosts. In dataset 1, there are 486 training dataset samples, 162 positive samples, and 324 negative samples, and dataset 2 with 325 training samples, 117 positive samples, and 208 negative samples.

Local models have the same architecture neural network. They are trained locally until they reach a local convergence. Initially, Model1 obtains 98.387% accuracy, being 98.208% to positive and 96.992% to negative samples; and Model2 obtains 70.789% accuracy, being 42.294% to positive and 98.496% to negative samples. These models are sent to a decentralized process using an average model in equation (1) for 30 rounds as shown in Figure 2 B.

Note that an unbalanced, non-idd distribution in local datasets using the average model, equation (1) results in the inability of the average model to reach a generalization. The average model maintains a performance of 50.00% in 30 rounds, and it has not improved.

Again, the red line represents the model with the same architecture trained with the complete centralized training dataset. It reaches 98.387% accuracy on the test dataset.

The probability of there occurring a mixture of unbalanced, non-iid datasets partitioned between different owners is very high, and being resilient to these variations is crucial to solving real-world problems. Therefore, designing a model averaging policy that achieves fast and continuous convergence on decentralized machine robust in Non-iid distribution is the biggest challenge, being an open problem [Kairouz et al. 2021].

Decentralized learning is a practical approach to increase the probability of broad generalization about a machine learning problem. However, the average model of current decentralized learning protocols fails at materializing the advantages in non-iid data in convex combination strategy problems on a scheme that executes complete local training described in Section 1.1.

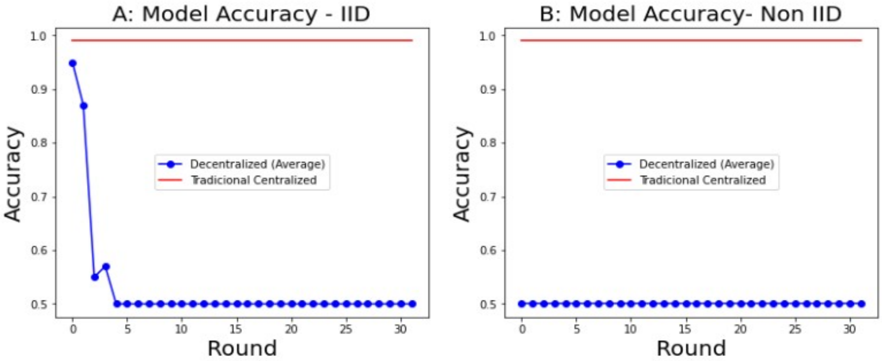

Figure 2. Assumption idd and non-idd with average model using a scheme that executes fully training of the local models before communication round with coordinator using combination convex

Another form of coordinating locally trained models using the linear but not the convex combination is analyzed.

3. Linear Non-Convex Combination

Using an average model to combine neural network models is not a very realistic assumption. Note that each layer in a feed-forward network is a non-linear process optimizing non-linear functions, but convex combinations define a linear process, and should not be expected to perform well in a combination of non-linear processes.

However, if the aim is to create a robust scheme for decentralized machine learning on neural networks, a non-convex combination is required. Thus equation (1) has to be substituted for something more adequate.

From the neural network literature, we also know that the problems of vanishing and exploding weights have to be avoided as well. However, the next simplest model available is a linear combination of parameters α^h, one for each participating host, whose sum is significantly different from 1, $\sum_{h=1}^{H} \alpha_h \neq 1$. Furthermore, the parameter α^h has to reflect the data balance of h with respect to the set of all data.

In this case, the general combination of weights respects (2).

$$\overline{w}_i = \sum_{h=1}^{H} \alpha^h w_i^h, \qquad 1 \leq i \leq W \tag{2}$$

where

- H, the number of participant hosts;
- w_i^h, the value of ith weight infor host model h;
- $\overline{w}_{i,\ell}$, the value of ith weight in the combined model.

To reflect the data balance of host h data with respect to the set of all data consider r_h, the relative size of the training corpus at h with respect to the total size, $r_h = \frac{T_t}{T}$, where T_h is the size of the corpus in h and $T = \sum_{h=1}^{T} T_h$.

We now examine how we can define the *linear combination factor* α^h.

3.1. A simple combination factor

The initial idea for the combination factor α^h is to achieve a fast convergence from local models by a scheme that executes complete local training, given by equation (3).

$$\alpha^h = (c + r_h) \tag{3}$$

The value r_h the relative size of h data with respct to the global corpus. The value c corresponds to a learning rate, and note that when c = 1 we are guaranteed that the sum of all α_h is considerably larger than 1.

3.1.1. Experiment under balanced, iid dataset

The neural network and training and testing datasets for local models are exactly the same used in Section 2.1. The local models are completely trained and they are submitted to the 30 rounds of communication. The coordinator combines the global model by equation (2) with the combination factor given by equation (3), with $c = 1$. The accuracy in the first rounds is 80.108%, 97.849%, 98.208%, 98.746%, 97.670%; at this point, the combination starts to decay.

Note in Figure 3 A (idd) that the coordinator is able to combine the local models in a decentralized way and improve the local results for the first iterations.

3.1.2. Experiment under unbalanced, non-iid dataset

Now consider the same neural network, with training and testing datasets for local models, are the same used in Section 2.2. The local models are trained with the number of epochs necessary to achieve the best local accuracy, and they are submitted to the 30 rounds of communication with the coordinator. The coordinator combines the global model by equation (2).

Note in Figure 3 B (non-idd) that the combination is robust for the first iterations, a behavior that is similar to the balance case when the coordinator combines the global model by equation (2) with combination factor given by equation (3), with $c = 1$. The combined model reaches 93.369%, 97.133%, 97.849%, 98.387%, 98.746%; at which point the combination starts to decay and "unlearn".

The value of α^h = 1 + r_h during the communication process led the combined model to a remarkable accuracy of 97$\%$ in the second round on both balanced and unbalanced cases, a considerable improvement from the convex combination, but it cannot guarantee a stably combined model after reaching the maximal precision.

This means that, after the combined model reaches the maximum generalization, the coordinator must stop the process. On the other hand, if the aim is obtaining a scheme that can reach a fast and continuous convergence on a decentralized machine on non-iid distribution, using the value of α^h = 1 + r_h is risky because it gives margin the possibility of exponential growth in the model parameters, where the model's weights may become too large during local training. As a result, the model loses the ability to generalize.

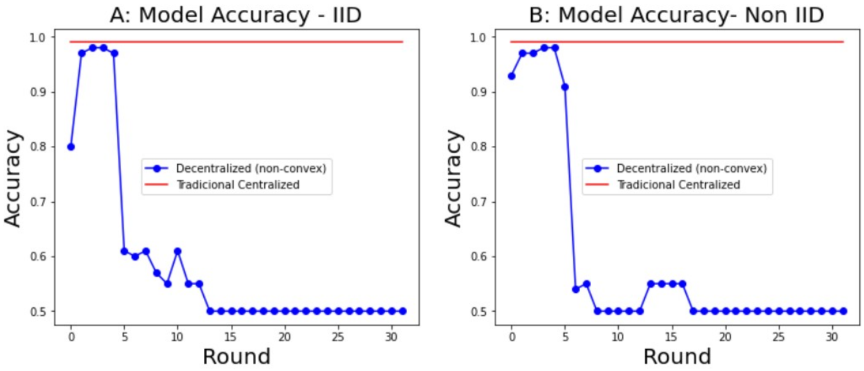

Figure 3. Assumption idd and non-idd using a scheme that executes fully training of the local models before communication round with coordinator using combination non-convex $\alpha^h = 1 + r_h$

At this point, we had two alternatives. To determine a stopped method before the avalanche learning or to design a regularization factor that does not suffer from this decay so fast. We explore here the second alternative, as it brings a more robust result.

3.2. Exponential combination factors

Here we explore linear combination factors of the exponential form. In traditional gradient descent backpropagation, the learning rate allows one to incorporate a fraction of the gradient at every training epoch. Then when the learning rate is significant, the learning process may diverge, and small learning rate values are used for better results.

Generally, the cases of exploding gradients can be avoided by carefully configuring the model with a small learning rate, scaling the target variables, and using a standard loss function. In analogy to this case, we use c as a learning rate, obtaining the combination factor of the equation (4).

$$\alpha^h = \mathrm{e}^{c \cdot r_h} \tag{4}$$

Note that previous combination factor (3) corresponds to the first two terms of a Taylor expansion of (4) when $c = 1$ in both equations; in fact, this was the motivation to try exponential factors, initially for $c = 1$ and then, to avoid exploding weights and instability after a few iterations, $c \lll= 1$. We now proceed to experiment with that form of combination.

Figure 4 shows the variation of c, and note that even if the number of hosts (client) increases, there is a point of stability. The combination of the weights happens in a non-convex way, and the c determines how fast or slows they will move to the optimal global. In this way, we can adjust the weights of the combined model concerning the descent of the loss gradient.

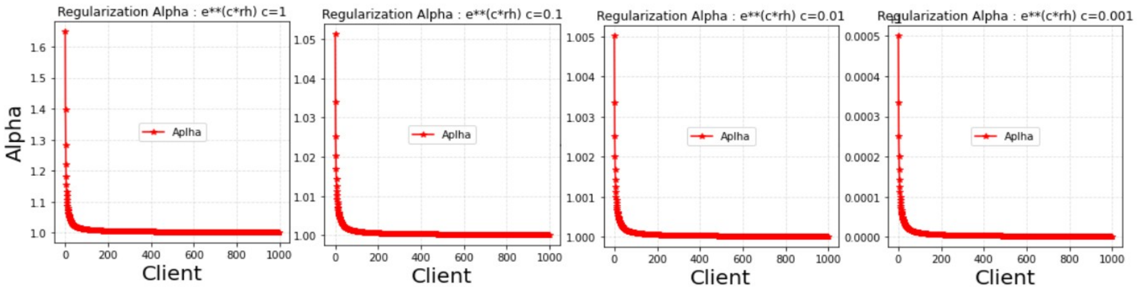

Figure 4. Regularization of Alpha to combination non-convex $\alpha^h = \mathrm{e}^{c \cdot r_h}$

3.2.1. Experiment under balanced, iid dataset

Again, the neural network and training and testing datasets for local models are exactly the same used in Section 2.1. The local models are completely trained and they are submitted to the 30 rounds of communication. The coordinator combines the global model by equation (2) with combination factor given by (4); however, this time the value of $c = 10^{-3}$ to provide with a small factor.

Note in Figure 5 A (idd) that the coordinator is able to combine local models in a non-convex, decentralized way in a totally robust process. In the first round, the combined model reaches 88.849%, 95.925%, 96.305%, 98.462%, 96.305%, 98.462% maintaining stability in the following rounds.

3.2.2. Experiment under unbalanced, non-iid dataset

Again, consider the same neural network, with training and testing datasets for local models, are exactly the same used in Section 2.2. The local models are trained with the number of epochs necessary to achieve the best local accuracy, and they are submitted to the

30 rounds of communication with the coordinator. The coordinator combines the global model by equation (2) with the combination factor given by equation (4) for $c = 10^{-3}$.

Note in Figure 5 B1 (non-idd) that the process remains robust even with unbalanced, non-iid local datasets. In the first round, the combined model reaches 50.000%, 85.484%, 97.746%, and 98.642%, maintaining stability in the following rounds.

3.2.3. Highly skewed non-iid data

To be sure that the result for this kind of combination is really robust, we submit the model to a stress test consisting of highly skewed non-iid data.

From the same Covid-19 dataset, a new training dataset was formed with 811 samples, of which 279 were positive and 532 negative. The test dataset contains 558 samples, 279 positive and 279 negative samples.

The split is made into two hosts. Dataset 1, with 330 samples, of which 223 positive samples and 107 negative samples; and Dataset 2, with 481 samples, of which 56 positive samples and 425 negative samples.

Local models: Model 1 reaches 50.000% total accuracy during the local training, being positive 0.000% and 98.00% negative. Model 2 reaches 50.000% total accuracy, being positive 98.00% and 0.000% negative.

In Figure 5 B2 (non-idd) we see the results of this experiment. The combination of unbalanced and non-iid models also achieves stability. The accuracy of the combined model reaches 75.986%, 97.849%, 97.849%, 98.462%, and 98.462%, maintaining stability in the following rounds.

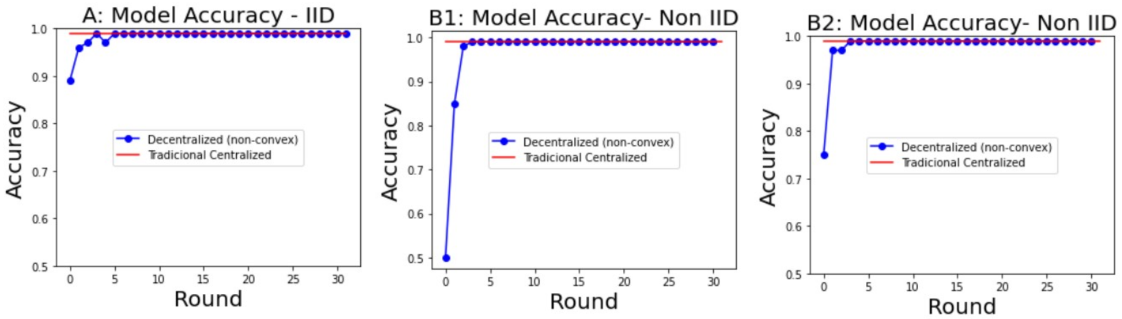

Figure 5. Assumption idd and non-idd using a scheme that executes fully training of the local models before communication round with coordinator using combination non-convex $\alpha^h = e^{c \cdot r_h}$

3.2.4. Comparison with Benckmark

The MNIST is a large database of handwritten digits that is widely used for training and testing in the field of machine learning. The MNIST database contains 60,000 training images and 10,000 testing images [LeCun 1998]. The training dataset was split non-iid across hosts and combined via scheme described in Section 1.1. We compare approaches combined convex (federated Learning, equation 1) and non-convex (Combined learning using the equations 2 and 4, where c=1.

Theoretical analysis shows that schemes that perform a single step in the decentralized machine learning achieve convergence [Kairouz et al. 2021] more easily, then we used a scenario where the averaging model works best to compare the approaches. The architecture used was Convolutional Neural Network (CNN) with 1,020,554 parameters, with 1 epoch before the communication round.

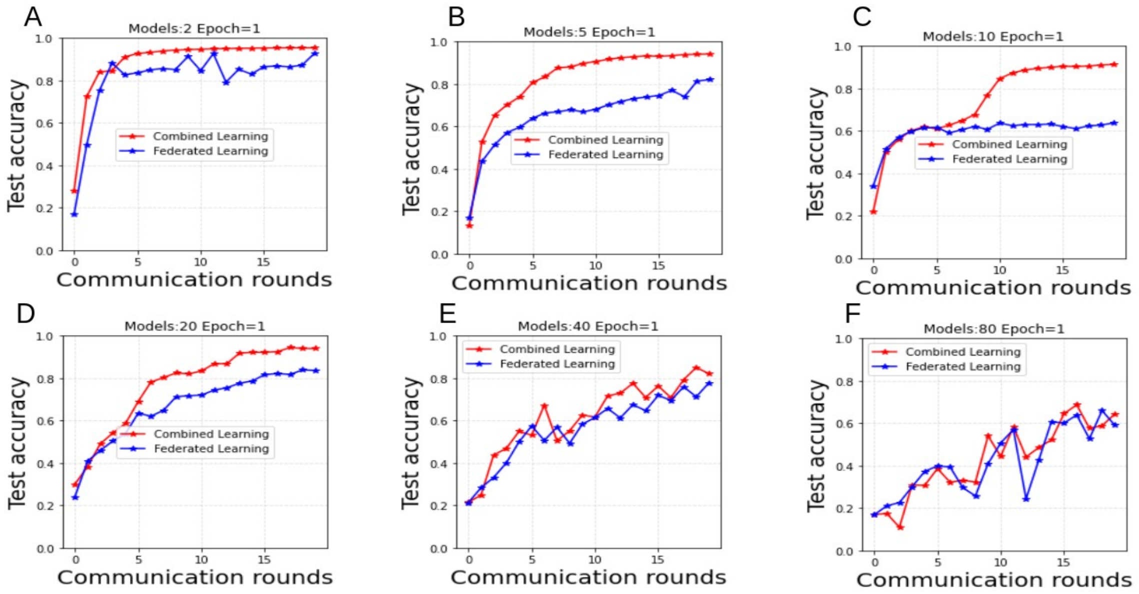

Figure 6. Compare non-convex and convex average model. A) 2 models, B) 5 models, C) 10 models, D) 20 models, E) 40 models, and F) 80 models.

Note that even in a more favorable communication scenario for the averaging model, the non-convex combination model performs better with non-idd data distributions as shown in figure 6. In all the experiments shown in figure 6 the training dataset (60,000 samples) was split non-iid across local hosts and combined via decentralized Private Preservation machine learning. The performance of models combined federated learning and combined learning was validated with a test dataset containing 10,000 testing images, having 1,000 samples of each label.

4. Conclusion

The experiments showed that non-convex linear Combined Learning with exponential combination factors is a new horizon to combine neural network models with a local training scheme. Furthermore, combining models with a non-convex process opens a new research direction.

The non-convex combination compares favorably to average models, exceeding their performance on a distribution non-idd. Furthermore, it improves the state-of-the-art using the scheme that performs several local update steps before a communication round on decentralized machine learning. Finally, the combined neural network model, non-convex, demonstrates remarkable robustness to the non-idd distribution.

Future work should look into different network architectures and different kinds of datasets, exploring non-convex combinations in practical machine learning problems. We should now investigate how this combination performs in another problem of CNN, Recurrent Architectures, such as Long Short Term Memory (LSTM) or Gated Recurrent Unit (GRU), with aim of investigating the limitations of this novel approach.

Acknowledgements

This work was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001. This work was carried out at the Center for Artificial Intelligence (C4AI-USP), with support by the São Paulo Research Foundation (FAPESP) (grant 2019/07665-4) and by the IBM Corporation. Aline Ioste was partly supported by CAPES. Marcelo Finger was partly supported by the São Paulo Research Foundation (FAPESP) (grants 2020/06443-5, 2014/12236-1); and the National Council for Scientific and Technological Development (CNPq) (grant PQ 303609/2018-4).

References

Acharya, J., Canonne, C. L., and Tyagi, H. (2020). Inference under information constraints i: Lower bounds from chi-square contraction. *IEEE Transactions on Information Theory*, 66(12):7835–7855.

Han, Y., Özgür, A., and Weissman, T. (2021). Geometric lower bounds for distributed parameter estimation under communication constraints. *IEEE Transactions on Information Theory*, 67(12):8248–8263.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. (2021). Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210.

Koloskova, A., Stich, S., and Jaggi, M. (2019). Decentralized stochastic optimization and gossip algorithms with compressed communication. In *International Conference on Machine Learning*, pages 3478–3487. PMLR.

Konečnỳ, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D. (2016). Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*.

LeCun, Y. (1998). The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. (2018). Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*.

Lian, X., Zhang, C., Zhang, H., Hsieh, C.-J., Zhang, W., and Liu, J. (2017). Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. *Advances in Neural Information Processing Systems*, 30.

Lin, T., Stich, S. U., Patel, K. K., and Jaggi, M. (2018). Don't use large mini-batches, use local sgd. *arXiv preprint arXiv:1808.07217*.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR.

McMahan, H. B., Moore, E., Ramage, D., and y Arcas, B. A. (2016). Federated learning of deep networks using model averaging. *arXiv preprint arXiv:1602.05629*, 2.

Mello, Luiz E, E. a. (2020). Opening Brazilian COVID-19 patient data to support world research on pandemics. Preprint available at `https://doi.org/10.5281/zenodo.3966427`.

Mohri, M., Sivek, G., and Suresh, A. T. (2019). Agnostic federated learning. *arXiv preprint arXiv:1902.00146*.

Nedić, A., Olshevsky, A., and Rabbat, M. G. (2018). Network topology and communication-computation tradeoffs in decentralized optimization. *Proceedings of the IEEE*, 106(5):953–976.

Paullada, A., Raji, I. D., Bender, E. M., Denton, E., and Hanna, A. (2021). Data and its (dis) contents: A survey of dataset development and use in machine learning research. *Patterns*, 2(11):100336.

Stich, S. U. (2018). Local sgd converges fast and communicates little. *arXiv preprint arXiv:1805.09767*.

Sun, C., Shrivastava, A., Singh, S., and Gupta, A. (2017). Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852.

Tang, H., Lian, X., Qiu, S., Yuan, L., Zhang, C., Zhang, T., and Liu, J. (2019). Deepsqueeze: Parallel stochastic gradient descent with double-pass error-compensated compression. *arXiv preprint arXiv:1907.07346*.

Wang, H., Yurochkin, M., Sun, Y., Papailiopoulos, D., and Khazaeni, Y. (2020). Federated learning with matched averaging. *arXiv preprint arXiv:2002.06440*.

Yurochkin, M., Agarwal, M., Ghosh, S., Greenewald, K., Hoang, T. N., and Khazaeni, Y. (2019). Bayesian nonparametric federated learning of neural networks. *arXiv preprint arXiv:1905.12022*.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2021). Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115.

Zhang, J., De Sa, C., Mitliagkas, I., and Ré, C. (2016). Parallel sgd: When does averaging help? *arXiv preprint arXiv:1606.07365*.

Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., and Chandra, V. (2018). Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*.