# Optimizing Communication Efficiency in Federated Learning

Arman Ehsasi, Lara Orlandic, Justyna Czestochowska

November 20, 2022

*Abstract*— **Federated Learning is an emerging solution in distributed training that preserves data privacy, but strains network bandwidth by continuously passing model weights from servers to clients. In this work, we simulate a Federated Training scenario for three different distributions of data across clients. We then investigate the impacts of varying the number of local training epochs, as well as weight quantization methods, on the communication rounds and total transmitted model bits required for validation accuracy of the model to converge. We find that 16-bit floating point quantization increases the number of communication rounds in IID data but decreases it for non-IID or mixed data. This quantization method reduces the total number of sent model bits for most distributions. Furthermore, increasing the local training epochs per communication round decreases the number of communication rounds by up to 72.7%, 67.9%, and 27.7% for IID, mixed, and non-IID data, respectively. Our optimizations resulted in reductions from baseline in the total transferred bits by 55.8-89.0% depending on the data configuration.**

## I. Introduction

Federated Learning is a distributed machine learning paradigm in which models are trained across a network of client devices and aggregated by a central server, preserving data privacy by keeping the data stored locally on each client [1]–[3] However, its primary drawback is the significant bandwidth consumption of continuously passing model weights from the central server to the clients and back [4], [5]. Several solutions have been proposed to limit this bandwidth overhead by quantizing or sparsifying the gradients of the model before sending them [5], [6]. However, these approaches do not examine the effects of varying the number of local training epochs performed on the clients prior each communication round, which reduces the number of communication rounds until convergence and thereby decreases the total network bandwidth used in training [3]. Furthermore, these methods are not verified using both independent and identically distributed (IID) and non-IID data, whereas data distribution significantly impacts the number of communication rounds required for convergence [2].

In this work, we simulate a Federated Learning architecture and estimate the total bits transmitted in training various network configurations. We study the effects of varying the number of local training epochs on the number of communication rounds until convergence in the case of three different data distributions: IID, non-IID, and a mixed dataset. We then apply two weight quantization methods to reduce the size and precision of the models prior to transmission. Overall, we aim to investigate the effects of weight quantization, client training epochs, and data distribution on network bandwidth consumption.

## II. Methods

### A. Federated Averaging Simulation

For all experiments, we simulate a Federated Learning architecture with one central server and 5 clients. We modified the Federated Averaging algorithm developed by McMahan et al. to include weight quantization and model size measurements prior to the simulation of model transmission from the server to the clients and back [3]. The goal of the network is to correctly classify handwritten digits of the MNIST dataset [7]. Each client $k$ contains a partition $P_k$ of the data set, on which it aims to minimize its local loss function $f_k(w)$:

$$f_k(w_k) = \frac{1}{n_k} \sum_{i \epsilon P_k} f_i(w_k) \tag{1}$$

where $n_k = |P_k|$, and $f_i(w_k)$ is the Cross-Entropy Loss of sample $i$ given the local model weights $w_k$. Then, the client model weights are adjusted using Stochastic Gradient Descent with a learning rate $\gamma = 0.001$. Each client $k$ is trained for $E$ epochs on its dataset $P_k$. The model weights are quantized by a function $q$ to reduce their bit precision and size, and the total size in bits of all model weights is computed and stored. Next, the quantized model weights $q(w_k)$ are sent to the central server for averaging[1],

$$w = \frac{1}{k} \sum_{k} q(w_k), \tag{2}$$

to compute the final model weights $w$, which are then tested on a validation dataset stored on the server. Finally, these weights are quantized again and sent to the clients to use as starting weights for the next round of local training.

[1]The *FederatedAveraging* algorithm is displayed in Figure 4 in the Appendix.

Training is stopped when the validation accuracy converges to a set threshold, at which point the number of communication rounds required to reach convergence $R$, as well as the total number of bits transmitted to and from the server per client $B_{trans}$, are recorded. The network simulation is implemented using the PyTorch framework, and the same CNN architecture used in [3] is implemented.

### B. Data Distribution on Clients

The data privacy offered by Federated Learning often results in a high variability of sample distributions across training participants, as a client's dataset $P_k$ is not representative of the whole population [3]. To investigate the impact of data distributions on $R$, we prepare three different sample distribution methods with an increasing amount of IID data. The first scenario is fully non-IID where we assume that clients have non-overlapping sets of digits. This means that each of our 5 clients contains samples from only two classes. This approach maintains full label separation, as each client performs training only on their unique dataset. The second scenario tries to compromise privacy with convergence speed by interspersing a set percentage of IID data into the otherwise non-IID data of each client. After investigating the effects of the percentage of shared IID samples on convergence time[2], we use 5% of the whole training dataset for this purpose, which results in 600 IID samples per client. Finally, the last scenario distributes data across clients in a full IID manner, which should ensure that the stochastic gradient is an unbiased estimate of the entire gradient [2].

### C. Weight quantization

Typically, model weights are stored as 32-bit floating point values. We implement two weight quantization methods to reduce the bit precisions of the weights prior to transmission, thereby reducing the model size.

*1) Float16 Quantization:* The low-precision bits of the models are ignored and weights are converted to float16 values, which cuts their size in half.

*2) Int8 Quantization:* Float32 weights are transformed into 8-bit integers by taking the maximum weight and multiplying it by a constant $c$ to scale its magnitude to 127, the maximum 8-bit integer value. All weights are multiplied by $c$ and converted to integers. After transmission, the weights are divided by $c$ and transformed back to float32. This reduces the weight size by a factor of 4.

[2]See Section A in the Appendix Fig. (5)

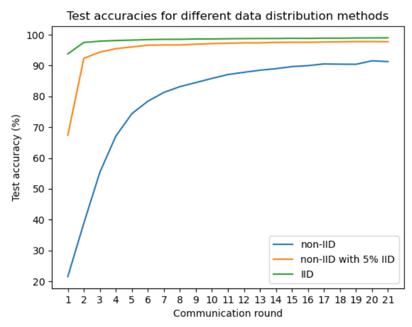

Fig. 1: Convergence for different data distributions without quantization, E=1

## III. Results

### A. Baseline

To obtain baseline convergence patterns of our three data distributions and set convergence thresholds, we plot the validation accuracies at each communication round using one local epoch per round ($E = 1$) and no weight quantization, as shown in Fig. 1. As previously reported [2], the non-IID data takes longer to converge than IID and mixed data distributions, and it converges to a lower accuracy. Consequently, we set the convergence threshold to 99% for IID, 98% for mixed, and 93% for non-IID distributions.

### B. Varying Number of Local Training Steps

Next, we vary $E$ to examine its effects on $R$, the communication rounds required for convergence. We can see in the first half of Fig. 2 that increasing $E$ generally results in a lower $R$ across all data distributions, meaning that there is a trade-off between local and global processsing. Increasing $E$ from 1 to 20 decreased $R$ for non-IID, mixed, and IID data by 27.8%, 67.9%, and 72.7%, respectively.

### C. Quantization

Along with varying E, we apply float16 weight quantization before transmitting the weights, as shown in Fig. 2. As expected, $R$ increases when quantization is applied to IID data, since the reduction in weight precision is compensated by additional communication rounds. However, in mixed and non-IID datasets,

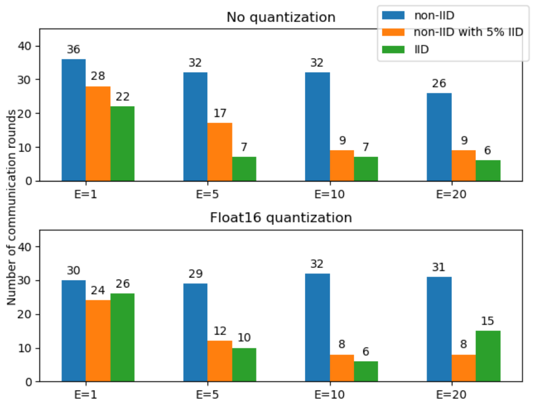

Fig. 2: The effects of varying E and adding float16 quantization on all data distributions

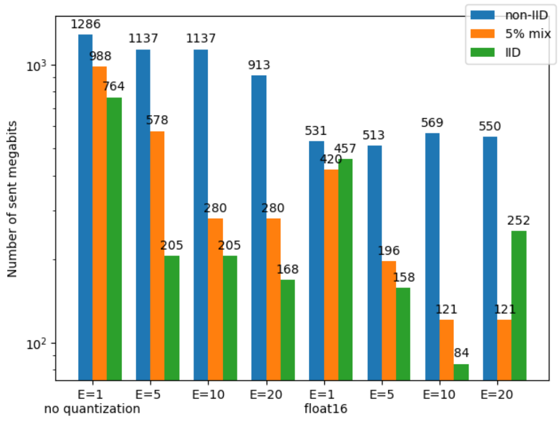

Fig. 3: $B_{trans}$ for various combinations of $E$ and quantization methods

quantization reduces $R$. For example, when $E = 5$, applying float16 quantization increased $R$ by 42.8% in the IID case, but decreased it by 29.4% and 9.38% in the mixed and non-IID cases, respectively. The results of the int8 quantization failed to converge to the desired thresholds of all models and are therefore omitted from the comparison.

### D. Bandwidth Estimation

To estimate the total number of transmitted bits required for convergence, $B_{trans}$, we add up all of the model sizes that were passed throughout the communication rounds $R$ and divide it by the number of clients, and the result is displayed in Fig. 3. We can see that although float16 quantization often results in a higher $R$, the smaller size of the models compensates for this and $B_{trans}$ is reduced in every scenario except for IID data with $E = 20$. For example, we see significant improvements from baseline when float16 quantization is applied and E is increased to 10: $B_{trans}$ is reduced by 55.8%, 87.7%, and 89.0% for non-IID, mixed, and IID data, respectively. Finally, we see that increasing $E$ tends to decrease $B_{trans}$, except in the case of non-IID data with float16 quantization.

## IV. Discussion and Conclusion

In this work, we examined the effects of varying the local training steps $E$, weight quantization methods, and data distributions among clients on the total number of communication rounds $R$ and transferred model bits $B_{trans}$ required to perform Federated Training. First, we noted that sharing only 5% of IID data across clients results in a significant reduction in $R$ from the non-IID case, and enables the validation accuracy to rise within 1% of the validation accuracy of the completely IID scenario.

Next, we conclude that increasing $E$ generally results in a reduction of $R$, but less so for non-IID data than mixed and IID data. This could be because more local steps can cause the local models to overfit on their data, and therefore more communication rounds are required for the central model to converge. Consequently, increasing $E$ also decreases $B_{trans}$ when no quantization is applied.

We tested two weight quantization methods, float16 and int8. Int8 quantization prevented most models from converging to their target accuracies and was therefore not precise enough to be useful. Float16 quantization increased $R$ for IID and mixed data due to the loss of weight precision, but decreased it for non-IID data. This is perhaps because the quantization constitutes a sort of dropout mechanism to prevent overfitting on local data. Furthermore, this increase in $R$ caused by quantization is compensated by the smaller memory consumption of the models and reduced $B_{trans}$ for nearly every data distribution.

Overall, by applying float16 quantization and increasing $E$ to 10, we achieved reductions in $B_{trans}$ by over 55% for all data distributions. These results may enable designers of Federated Learning networks to choose the number of local training epochs and quantization schemes that best fit their intended data distribution.

## References

[1] B. K. Beaulieu-Jones, W. Yuan, S. G. Finlayson, and Z. S. Wu, “Privacy-Preserving Distributed Deep Learning for Clinical Data,” 12 2018. [Online]. Available: http://arxiv.org/abs/1812.01484

[2] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, “Federated Learning with Non-IID Data,” 6 2018. [Online]. Available: http://arxiv.org/abs/1806.00582

[3] H. Brendan McMahan Eider Moore Daniel Ramage Seth Hampson Blaise AgüeraAg and A. Arcas, “Communication-Efficient Learning of Deep Networks from Decentralized Data,” Tech. Rep., 2017.

[4] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally, “DEEP GRADIENT COMPRESSION: REDUCING THE COMMUNICATION BANDWIDTH FOR DISTRIBUTED TRAINING,” Tech. Rep.

[5] S. U. Stich, J.-B. Cordonnier, and M. Jaggi, “Sparsified SGD with Memory,” Tech. Rep.

[6] D. Alistarh, D. Grubic ETH Zurich, J. Z. Li, R. Tomioka, and M. Vojnovic, “QSGD: Communication-Efficient SGD via Gradient Quantization and Encoding,” Tech. Rep.

[7] “MNIST handwritten digit database, Yann LeCun, Corinna Cortes and Chris Burges.” [Online]. Available: http://yann.lecun.com/exdb/mnist/

## V. Appendix

**Algorithm 1** `FederatedAveraging`. The $K$ clients are indexed by $k$; $B$ is the local minibatch size, $E$ is the number of local epochs, and $\eta$ is the learning rate.

```
Server executes:
  initialize $w_0$
  for each round $t = 1, 2, \dots$ do
    $m \leftarrow \max(C \cdot K, 1)$
    $S_t \leftarrow$ (random set of $m$ clients)
    for each client $k \in S_t$ in parallel do
      $w_{t+1}^k \leftarrow$ ClientUpdate$(k, w_t)$
    $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$

ClientUpdate$(k, w)$:   // Run on client $k$
  $\mathcal{B} \leftarrow$ (split $\mathcal{P}_k$ into batches of size $B$)
  for each local epoch $i$ from 1 to $E$ do
    for batch $b \in \mathcal{B}$ do
      $w \leftarrow w - \eta \nabla \ell(w; b)$
  return $w$ to server
```

Fig. 4: *FederatedAveraging* Algorithm as Presented in [3]

### A. Mixed Non-IID Data

[2] explain that the degree of accuracy may be correlated with the amount of skewed data in a distribution. Indeed, we have observed that with the 5% IID-data mixed into the non-IID set, our results did perform better as opposed to only having non-IID data. We observed that with increasing the amount of the share of IID-data, we move closer to the original IID data set. This behaviour can be observed in Figure 5.

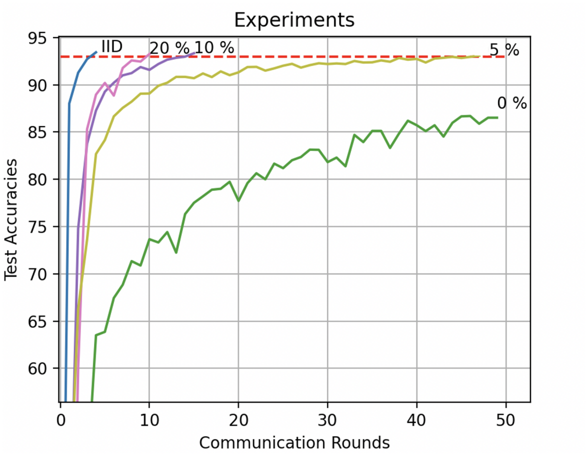

Fig. 5: Performance with varying IID-mixes (Percentages refer to the share of IID data per client)