Adaptive Ratio-Based-Threshold Gradient Sparsification Scheme for Federated Learning

Jeong Min Kong
Department of Electrical and Computer Engineering
University of Toronto
Toronto, Canada
jeong.kong@mail.utoronto.ca

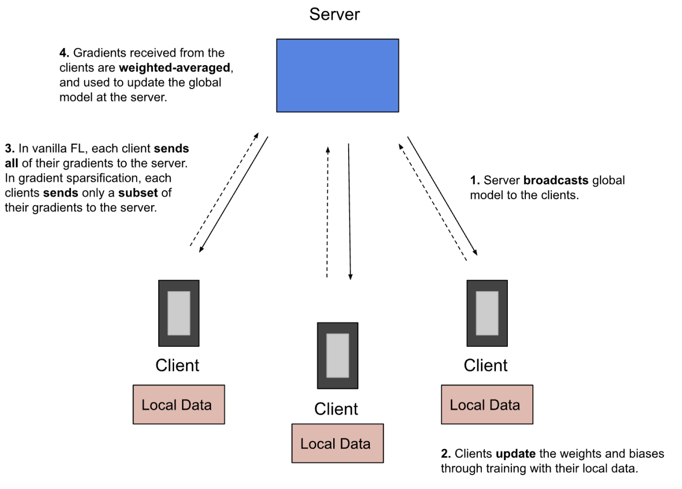
Elvino Sousa
Department of Electrical and Computer Engineering
University of Toronto
Toronto, Canada
es.sousa@utoronto.ca

***Abstract*—Federated learning (FL) is a distributed learning paradigm that has received great attention over the past several years due to its privacy-preserving property. As the models involved in FL are usually dense and overparameterized however, various studies are being conducted in gradient sparsification to reduce the high communication overhead. While many of the recently-presented schemes that are variations of top-k have shown competitive inference accuracy convergence to the baseline "vanilla" FL, they have a fixed sparsity rate throughout all of the communication rounds, which leads to an unnecessary excessive transmission of gradients as the global model converges. Furthermore, the constant-threshold gradient sparsification method called Threshold-ν, that is well-known for its dynamic rate, does not account for the ratio between the gradient and the pre-update parameter value, causing some gradients that are orders of magnitude larger than the pre-update parameter values to be neglected in the following aggregation process. In this paper, we introduce a new algorithm that addresses both of these issues, called *adaptive ratio-based-threshold gradient sparsification method*. Our main idea is introducing a custom gradient sparsity threshold for each local parameter based on their pre-update value and a hyperparameter denoted as ψ. We demonstrate through image classification experiments on MNIST and CIFAR-10 datasets in both independent-and-identically-distributed (IID) and non-IID settings that under optimal ψ, the gradient sparsity rates adapt & increase as the global model converges, while simultaneously producing inference accuracies that are competitive to vanilla FL.**

***Index Terms*—Distributed Deep Learning, Federated Learning, Gradient Sparsification**

I. Introduction

Federated learning (FL) is a privacy-preserving distributed learning framework that was first introduced in 2017 [1]. Unlike traditional distributed learning frameworks where raw data is exchanged between the clients and the server, the gradients of clients' local neural networks are shared in FL. As a consequence of this, the gradients of clients' local models are utilized to update the global model at the server, instead of the raw data. There are various ways to aggregate the local gradients, the most popular approach being Federated Averaging (FedAvg). FedAvg was first introduced in ”vanilla” FL, and it simply computes the weighted average of the local gradients to update the global model. In vanilla FL, which is the most popular FL framework being used today, all of the local gradients are sent to the server for aggregation. However, it is important to highlight that most of the models involved in FL are dense and overparameterized as discussed in [2], and such nature puts an extremely heavy burden on the communication systems. To reduce the high communication overhead in FL, researchers have started exploring gradient sparsification, which are methods that allow clients to transmit only a subset of their local gradients to the server, while still being able to produce global model inference accuracy close to that of vanilla FL.



4. Gradients received from the clients are **weighted-averaged**, and used to update the global model at the server.

3. In vanilla FL, each client **sends all** of their gradients to the server. In gradient sparsification, each clients **sends** only a **subset** of their gradients to the server.

1. Server **broadcasts** global model to the clients.

2. Clients **update** the weights and biases through training with their local data.

Fig. 1. FL system architecture.

II. Related Work and Contribution

Various gradient sparsification algorithms have been presented for FL in recent years. One of the most well-known algorithms is called random-k, in which only the randomly-sampled k gradients of client's neural network are communicated to the aggregation server during each communication round [3]. Another popular algorithm is called top-k, and in this approach, the largest k gradients of client's neural network are communicated to the aggregation server [4]. A few variations of top-k have been introduced to yield improved fairness, reduced computational load, and better inference ac-

979-8-3503-3559-0/23/$31.00 ©2023 Crown

curacy and convergence time compared to the original method, such as FAB-top-k and rTop-k [5,6]. Furthermore, other novel gradient sparsification methods based on fixed sparsity rates have been proposed, most notably GradDrop [7] and adaptive-quantization-based algorithm in [8]. In addition to gradient sparsification, model pruning have also been extensively studied to reduce the communication bottleneck in FL, leading to innovative techniques such as Complement Sparsification, FedTiny, LotteryFL, and FL-PQSU [2,9,10,11].

While these algorithms have demonstrated competitive inference accuracy and convergence time compared to vanilla FL, most of them have a sparsity rate that is constant throughout all of the communication rounds. As a consequence, even when the global model begins to converge at higher communication rounds, and there are thus less drastic local parameter (weight/bias) updates compared to earlier communication rounds, the clients unnecessarily continue to transmit the same number of gradients to the aggregation server. There is a method well-known for having a dynamic sparsity rate, called Threshold-ν [12], where all of the local gradients that exceed a common threshold ν get transmitted to the aggregation server, and those that fall below get accumulated locally until they reach ν. However, this approach fails to take into account the ratio between the gradient and the pre-update parameter value, causing some gradients that are orders of magnitude larger than the pre-update parameter values to be neglected in the following aggregation process.

In this paper, we present a new gradient sparsification algorithm that addresses both of the issues aforementioned, while still maintaining competitive inference accuracy and convergence time to the baseline vanilla FL. Unlike Threshold-ν, our method introduces a custom gradient sparsity threshold for each parameter based on their pre-update value. Furthermore, we demonstrate through various experimentation that the gradient sparsity rate in our method is *adaptive*, and increases as the global model becomes more convergent, unlike the fixed sparsity rate schemes that were previously discussed.

III. Methodology

The algorithmic description of our proposed scheme is shown in Algorithm 1. First, the aggregation server randomly (uniformly) samples a subset of available clients that will participate in FL. The aggregation server then broadcasts the global model to the selected clients, and these clients train the received model with their own, local data. After the local training, each weight and bias of the local model is examined for gradient sparsification. The gradient sparsification procedure is as follows: if the absolute gradient, ie. the absolute difference between the updated local weight/bias and the pre-update local weight/bias, is greater than ψ% of the pre-update local weight/bias, then transmit the gradient to the aggregation server; otherwise, do not transmit the gradient to the aggregation server. ψ is a hyperparameter, and we demonstrate in the experimentation section that the value of ψ has a significant impact on both the gradient sparsity and convergence of the global model. After collecting the sparse gradients from its selected clients, the aggregation server employs FedAvg to update its global model. This entire process is repeated until the global model converges. Note that we are only considering gradient sparsification in the uplink (from clients to aggregation server), as most bandwidth consumption in FL arises in the uplink; however, this algorithm can also be easily adapted for the downlink (aggregation server to clients).

Algorithm 1 Adaptive Ratio-Based-Threshold Gradient Sparsification Scheme

```
Definitions: $\psi$ = hyperparameter, $N$ = # of communication rounds (based on the estimate of when the global model will converge), $M$ = # of randomly selected clients participating in FL, $J$ = # of parameters (weights and biases) in global/local model, $w^t$, $t = 0, ..., N-1$, = global parameters at communication round $t$, $w_j^t$, $j$ = $1, ..., J$, = $j$th element of $w^t$, $D_i$, $i$ = $1, ..., M$, = local dataset at client $i$, $g_i^t$ = local gradient of client $i$ at communication round $t$, $g_{i,j}^t$ = $j$th element of $g_i^t$
Output: $w^N$
  Broadcast randomly initialized $w^0$ to the clients
  for $t = 0, ..., N-1$ do
    At the clients:
    Receive $w^t$ from aggregation server
    for $i = 1, ..., M$ do
      $g_i^t \leftarrow \nabla Loss(w^t, D_i)$
      for $j = 1, ..., J$ do
        if $|g_{i,j}^t| > \frac{\psi}{100} \cdot w_j^t$ then
          Keep $g_{i,j}^t$
        else
          $g_{i,j}^t \leftarrow 0$
        end if
      end for
      Transmit $g_i^t$ to aggregation server
    end for
    At the aggregation server:
    $\hat{g}^t \leftarrow \frac{1}{M} \sum_{i=1}^{M} g_i^t$
    $w^{t+1} \leftarrow w^t - \hat{g}^t$
  end for
  return $w^N$
```

IV. Experimentation and Discussion

A. Experimentation Setting

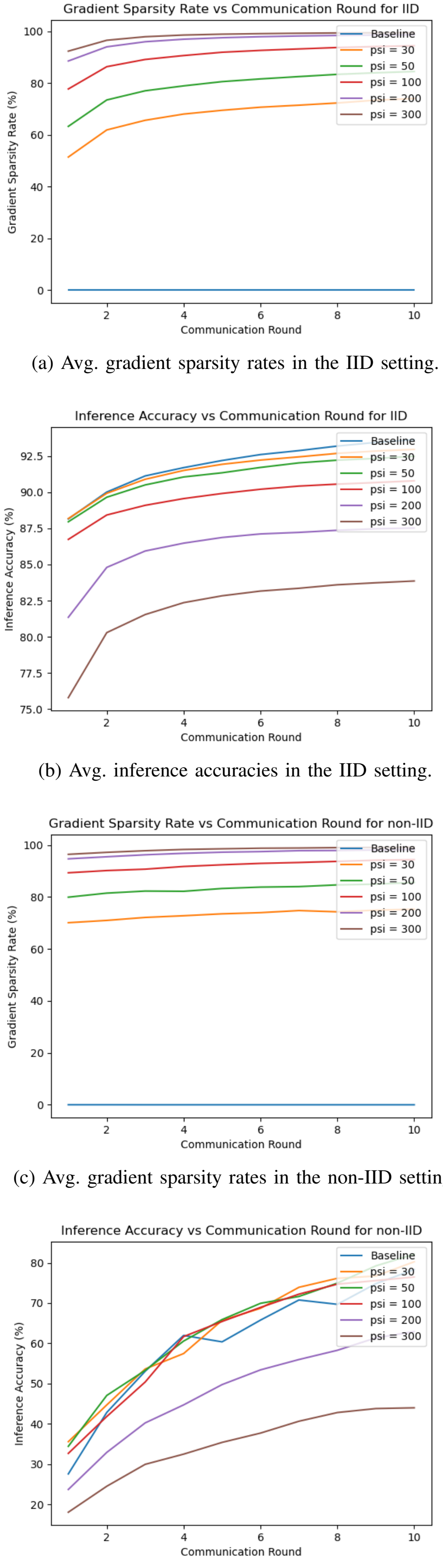
We perform our experiments on image classifications tasks using two of the most popular datasets, MNIST [13] and CIFAR-10 [14]. Both MNIST and CIFAR-10 have 10 output classes, MNIST containing grayscale images of handwritten numbers between 0 and 9, and CIFAR-10 containing color images of animals and transportation vehicles.

For all of our experiments, there is a total of 100 clients, in which 10 of them are randomly (uniformly) selected to participate in FL. We examine both independent-and-identically-distributed (IID) and non-IID settings. In both settings, the training dataset is evenly partitioned among the participating clients. For the IID case, each client is randomly assigned a uniform distribution over 10 classes. For the non-IID case, each client is assigned images strictly from 1 or 2 classes, that are non-overlapping with the images assigned to other clients. Per communication round, 10 epochs of local training is completed at each client.

We use two neural network architectures, multi-layer perceptron (MLP) and convolutional neural network (CNN), for each dataset & IID/non-IID setting described above. For both MNIST and CIFAR-10, MLP has one hidden layer of 64 neurons with ReLU activation, and an output layer with Softmax activation. For MNIST, CNN consists of two convolutional layers (10 and 20 output channels, respectively, with 5x5 filter, stride of 1, and ReLU activation), max pooling (with 2x2 filter and stride of 1) after each convolutional layer, one fully-connected (FC) layer of 50 neurons with ReLU activation, and an output layer with LogSoftmax activation. For CIFAR-10, CNN consists of two convolutional layers (6 and 16 output channels, respectively, with 5x5 filter, stride of 1, and ReLU activation), max pooling (with 2x2 filter and stride of 1) after each convolutional layer, two FC layers (120 and 84 neurons, respectively, with ReLU activation), and an output layer with LogSoftmax activation. For all, we use negative log likelihood (NLL) as the loss function, stochastic gradient descent (SGD) with momentum of 0.5 and learning rate of 0.01 as the optimizer, and 10 as the batch size.

B. Results

Figure 2 and 3 illustrate the gradient sparsity rates & global model test inference accuracies at each communication round for various values of ψ. Figure 2 is the outcome of training MLP on the MNIST dataset, and Figure 3 is the outcome of training CNN on the MNIST dataset. These plots show that under optimal ψ in both IID and non-IID settings, as the communication round progresses, the gradient sparsity rate increases while simultaneously producing an inference accuracy competitive to vanilla FL. In both of these examples, the approximate optimal value of ψ is 100. This is because when ψ is below 100, even though the inference accuracy convergence is also similar to that of vanilla FL, the gradient sparsity rate is always substantially lower than when ψ is 100. On the other hand, when ψ is greater than 100, even though the gradient sparsity rate is always higher than when ψ is 100, the inference accuracy convergence is significantly worse compared to that of vanilla FL. Mathematically, we define optimal ψ as ψ that maximizes the average gradient sparsity rate across all communication rounds, under the constraint that global model inference accuracies do not lie more than 5% below that of vanilla FL for all communication rounds. At the approximate optimal value of 100 in these MNIST experiments, as the global model progressively converges through 10 communications rounds, the gradient sparsity rate constantly adapts & climbs from already-high 77.68% to 94.38% for MLP and from 74.24% to 92.72% for CNN in the IID setting,



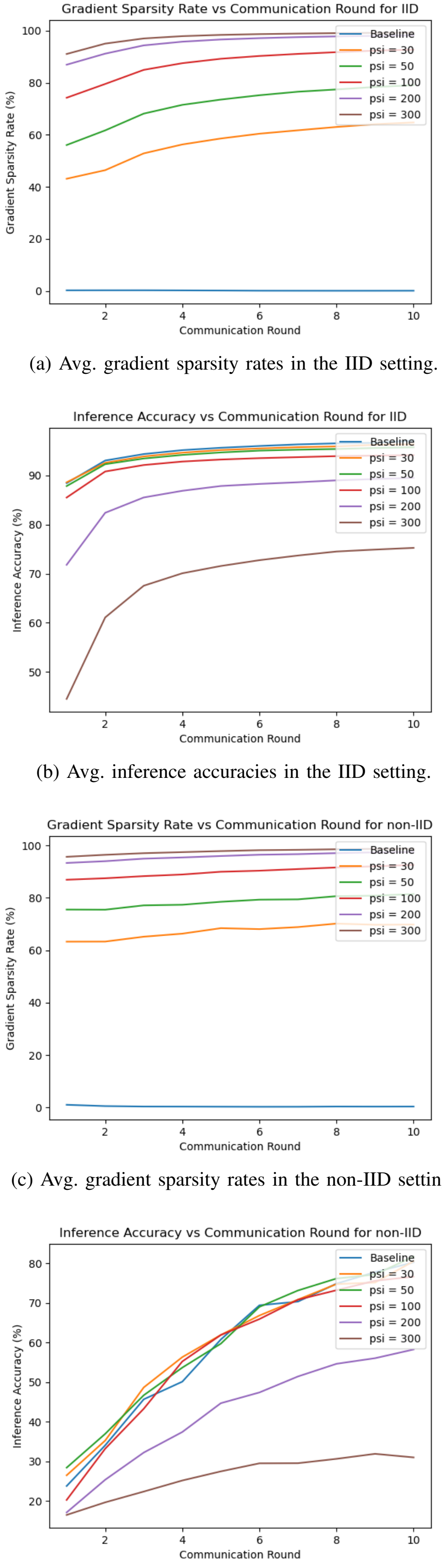
(a) Avg. gradient sparsity rates in the IID setting.

(b) Avg. inference accuracies in the IID setting.

(c) Avg. gradient sparsity rates in the non-IID setting.

(d) Avg. inference accuracies in the non-IID setting.

Fig. 2. Avg. gradient sparsity rates and inference accuracies for MLP-MNIST.



(a) Avg. gradient sparsity rates in the IID setting.

(b) Avg. inference accuracies in the IID setting.

(c) Avg. gradient sparsity rates in the non-IID setting.

(d) Avg. inference accuracies in the non-IID setting.

Fig. 3. Avg. gradient sparsity rates and inference accuracies for CNN-MNIST.

and from 89.32% to 94.39% for MLP and from 86.88% to 92.56% for CNN in the non-IID setting. In experiments with CIFAR-10, we observed widely-varying optimal ψ values with different neural network architectures and IID/non-IID settings, which is contrasting to MNIST. More specifically, we determined that the optimal ψ is 200 for MLP and 50 for CNN in the IID setting, and 300 for MLP and 60 for CNN in the non-IID setting. Despite this difference in behavior compared to MNIST, substantial jumps in gradient sparsity rates were similarly made at the optimal ψ values. For instance, at their respective optimal value, the gradient sparsity rate ascended from already-high 87.56% to 96.01% in the IID setting and from 94.39% to 98.14% in the non-IID setting, for MLP.

V. Conclusion and Future Work

This paper presented a novel FL gradient sparsification scheme that selects sparse gradients based on local parameters' pre-update values and hyperparameter ψ. Through image classification experiments on MNIST and CIFAR-10 in both IID and non-IID settings, we have shown that under optimal ψ, the gradient sparsity rates adapt & increase as the global model converges, while concurrently outputting inference accuracies close to that of baseline vanilla FL. In the future, we plan to better understand the relationship between optimal ψ and different types of datasets and neural network architectures in both IID and non-IID settings. Through these studies, we aim to develop more efficient generalized techniques for finding optimal ψ, that can replace the current brute-force approach.

References

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Aguera y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artificial Intelligence and Statistics*, 2017, pp. 1273–1282.

[2] X. Jiang, and C. Borcea, "Complement sparsification: Low-overhead model pruning for federated learning," *arXiv preprint arXiv:2303.06237*, 2023.

[3] L. Song et al., "Communication efficient SGD via gradient sampling with Bayes prior," *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nashville, TN, USA, 2021, pp. 12060-12069.

[4] D. Alistarh, T. Hoefler, M. Johansson, N. Konstantinov, S. Khirirat, and C. Renggli, "The convergence of sparsified gradient methods," in *Proc. Neural Inf. Process. Syst. (NeurIPS)*, 2018, pp. 5973–5983.

[5] P. Han, S. Wang, and K. K. Leung. "Adaptive gradient sparsification for efficient federated learning: An online learning approach," in *Proc. IEEE 40th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Singapore, 2020, pp. 300–310.

[6] L. P. Barnes, H. A. Inan, B. Isik, and A. Özgür, "rTop-k: A statistical estimation approach to distributed SGD," *IEEE J. Sel. Areas Inf. Theory*, vol. 1, no. 3, pp. 897–907, Nov. 2020.

[7] A. F. Aji and K. Heafield, "Sparse communication for distributed gradient descent," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Copenhagen, Denmark, Sep. 2017.

[8] N. Dryden, T. Moon, S. A. Jacobs, and B. Van Essen, "Communication quantization for data-parallel training of deep neural networks," *2016 2nd Workshop on Machine Learning in HPC Environments (MLHPC)*, Salt Lake City, UT, USA, 2016, pp. 1-8.

[9] H. Huang, L. Zhang, C. Sun, R. Fang, X. Yuan, and D. Wu, "FedTiny: Pruned federated learning towards specialized tiny models," *arXiv preprint arXiv:2212.01977*, 2022.

[10] A. Li, J. Sun, B. Wang, L. Duan, S. Li, Y. Chen, and H. Li, "LotteryFL: Personalized and communication-efficient federated learning with Lottery Ticket Hypothesis on non-IID datasets," *arXiv preprint arxiv:2008.03371*, 2020.

[11] W. Xu, W. Fang, Y. Ding, M. Zou and N. Xiong, “Accelerating federated learning for IoT in big data analytics with pruning, quantization and selective updating,” in *IEEE Access*, vol. 9, pp. 38457-38466, 2021.

[12] N. Strom, “Scalable distributed DNN training using commodity GPU cloud computing,” in *Proc. of INTERSPEECH*, 2015.

[13] 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. 2278-2324, 1998.

[14] A. Krizhevsky and G. Hinton, “Learning multiple layers of features from tiny images,” 2009.