Federated Learning with Heterogeneous Quantization

Cong Shen
University of Virginia
Charlottesville, VA 22904, USA

Shengbo Chen
Henan University
Kaifeng, Henan 475001, China

2020 IEEE/ACM Symposium on Edge Computing (SEC) | 978-1-7281-5943-0/20/$31.00 ©2020 IEEE | DOI: 10.1109/SEC50012.2020.00060

***Abstract*—Quantization of local model updates before uploading to the parameter server is a primary solution to reduce the communication overhead in federated learning. However, prior literature always assumes homogeneous quantization for all clients, while in reality devices are heterogeneous and they support different levels of quantization precision. This heterogeneity of quantization poses a new challenge: fine-quantized model updates are more accurate than coarse-quantized ones, and how to optimally aggregate them at the server is an unsolved problem. In this paper, we propose FEDHQ: Federated Learning with Heterogeneous Quantization. In particular, FEDHQ allocates different weights to clients by minimizing the convergence rate upper bound, which is a function of quantization errors of all clients. We derive the convergence rate of FEDHQ under strongly convex loss functions. To further accelerate the convergence, the instantaneous quantization error is computed and piggybacked when each client uploads the local model update, and the server dynamically calculates the weight accordingly for the current round. Numerical experiments demonstrate the performance advantages of FEDHQ+ over conventional FEDAVG with standard equal weights and a heuristic scheme which assigns weights linearly proportional to the clients' quantization precision.**

I. INTRODUCTION

Federated learning (FL) has received tremendous attention since the seminal work [1]–[3]. It is well known that communication is the primary bottleneck for federated learning, mainly due to two key factors: 1) modern machine learning (ML) models are usually very large, e.g., millions of parameters for deep neural networks (DNN); 2) the number of clients may be large, especially when massive Internet-of-Things (IoT) devices participate in model training. Therefore, for FL, a communication-efficient approach is very important. If the cost of each communication is reduced, more devices might be able to participate, which will make federated learning more attractive.

It has been shown that quantization of each client's model update before communication to the parameter server is an efficient solution to alleviate the communication overhead in FL [4]–[8]. However, most prior works assume a homogeneous quantization setting for all clients. There has been no consideration of the heterogeneity of quantization precision levels across clients, which is in fact common in practice. For instance, mobile phones usually have higher quantization accuracy than low-cost IoT devices. This heterogeneity of quantization poses a new challenge: model updates from fine-quantized clients are more accurate than the updates from coarse-quantized ones, and how to optimally aggregate them at the server is a new problem.

In this paper, we address the aforementioned challenge, i.e., how to aggregate the updates from clients with heterogeneous quantization precision levels. In particular, we propose FEDHQ, an algorithm that judiciously assigns different weights to clients. By minimizing the upper bound of the convergence rate for strongly convex loss functions, the resulted weights can be precisely characterized as a function of the quantization errors of all clients. This function reveals that clients with smaller quantization error should be given higher weights, and all clients contribute to the model aggregation regardless of their quantization precision. We then proposed FEDHQ+, where the server dynamically allocates weights to clients using the *instantaneous* quantization errors reported from all clients. We show via simulations that FEDHQ+ outperforms both the baseline FEDAVG which assigns equal weights to all clients, and a heuristic aggregation scheme which assigns weights that are linearly proportional to the quantization precision.

The authors contribute equally to the paper.

II. THE FEDHQ ALGORITHM

In this section, we first define the FL framework with quantization, and then propose the FEDHQ algorithm.

A. Preliminaries

We consider the standard machine learning problem $\min_{x\in\mathbb{R}^d} f(x)$, where $f : \mathbb{R}^d \to \mathbb{R}$ is the differentiable objective function, and $x \in \mathbb{R}^d$ is the variable that we would like to optimize. We assume that there are n clients in the federated setting. The problem can be rewritten as

$$\min_{x\in\mathbb{R}^d} \sum_{i=1}^{n} \frac{m_i}{m} f_i(x), \tag{1}$$

where $f_i : \mathbb{R}^d \to \mathbb{R}$ is the local objective function for client i, and m_i is the number of samples for client i with $\sum_i m_i = m$. All the involved functions $f(\cdot)$ and $f_i(\cdot)$ can be convex or non-convex.

In [2], FEDAVG has been proposed to solve problem (1). The algorithm randomly selects a subset of clients, which run SGD locally for several epochs. The server periodically

978-1-7281-5943-0/20/$31.00 ©2020 IEEE
DOI 10.1109/SEC50012.2020.00060

updates the model by aggregating the local SGD updates from selected clients, and sends the updated global model to clients.

B. Quantization

In our proposed algorithm, each client receives the model x_t from the server at the beginning of round t, and obtains the new model x_{t+1}^i by running SGD locally. A quantizer operator $Q_i(\cdot)$ is then applied on the difference between the new updated model and the previously received model, i.e., $x_{t+1}^i - x_t$. All clients report the quantized $Q_i(x_{t+1}^i - x_t)$ to the server, and the server updates its model by aggregating all clients' updates. The aggregation method when $\{Q_i\}_{i=1}^n$ are heterogeneous is the focus of this paper.

Recently, many sophisticated quantization methods have been proposed. We describe a quantization example used in [9], which is also adopted in our experiments. We emphasize that (i) the proposed algorithm applies to any quantizer; and (ii) the analytical result holds for any quantizer as long as Assumption 3 in the next section is satisfied.

Example: Block Floating Point (BFP) Quantization. We allocate W bits to represent each number, and maximal F bits to represent the exponent. Then the actual shared exponent $E(x)$ for a block of numbers w is given by

$$E(w) = \text{clip}(\lfloor \log_2 \max_i |w_i| \rfloor, -2^{F-1}, 2^{F-1} - 1), \quad (2)$$

where $\text{clip}(a, b, c) = \max(\min(a, b), c)$. The minimal quantization gap $\theta = 2^{E(x)+2-W}$. For $x \in w$, we have the quantization function as

$$Q(x) = \begin{cases} \text{clip}(\theta \lfloor \frac{x}{\theta} \rfloor, -2^{E(w)+1}, 2^{E(w)+1} - 2^{E(w)+2-W}) \\ \qquad \text{w.p.} \quad \lceil \frac{x}{\theta} \rceil - \frac{x}{\theta}, \\ \text{clip}(\theta \lceil \frac{x}{\theta} \rceil, -2^{E(w)+1}, 2^{E(w)+1} - 2^{E(w)+2-W}) \\ \qquad \text{w.p.} \quad \frac{x}{\theta} - \lfloor \frac{x}{\theta} \rfloor. \end{cases} \quad (3)$$

C. Algorithm Description

Prior literature always assumes clients with homogeneous quantization levels, while in reality clients are heterogeneous and they support different levels of quantization precision. Therefore, we consider the following problem

$$\min_{x \in \mathbb{R}^d} \sum_{i=1}^n p_i f_i(x), \quad (4)$$

where p_i is the assigned weight for client i, and $\sum_i p_i = 1$. Note that problem (4) is a generalization of (1).

We describe the proposed algorithm FEDHQ that aims at solving (4). We run a total T rounds of federated learning. At the beginning of each round $t = (1, 2, \cdots, T)$, the server randomly selects a subset of clients $\mathcal{S}_t$, and notifies each client $i \in \mathcal{S}_t$ the latest model $x_t^i = x_t$. Then each client updates its model by running SGD locally K times and obtains x_{t+1}^i. The client computes the model update and uploads the quantized update to the parameter server. The server then aggregates the received quantized local update, by taking a *weighted average*, that is,

$$x_{t+1} = x_t + \sum_{i=1}^n p_i Q_i(x_{t+1}^i - x_t)). \quad (5)$$

FEDHQ is formally described in Algorithm 1.

Algorithm 1: The FEDHQ Algorithm

Input: Server aggregation weight $\{p_i\}_{i=1}^n$;
for $t = 1$ **to** T **do**
  Server randomly selects $\mathcal{S}_t$, and notifies all clients x_t in $\mathcal{S}_t$;
  for *client* $i \in \mathcal{S}_t$ **do**
    $x_{t,0}^i \leftarrow x_t$;
    for $\tau = 0$ **to** $K - 1$ **do**
      $x_{t,\tau+1}^i = x_{t,\tau}^i - \eta_t \nabla f_i(x_{t,\tau}^i)$;
    end
    $x_{t+1}^i \leftarrow x_{t,K}^i$;
    Computes $Q_i(x_{t+1}^i - x_t)$ and sends to server;
  end
  Server updates $x_{t+1} = x_t + \sum_{i=1}^n p_i Q_i(x_{t+1}^i - x_t))$;
end

Notice that in the algorithm above, p_i is a tunable parameter that we can design. In Section III, we show that the optimal p_i is a function of quantization errors.

III. CONVERGENCE ANALYSIS

In this section, we present the main theoretical results on the performance of FEDHQ under a strongly convex loss function f. For the purpose of better illustration of the key idea, in the following analysis we always assume *all* clients are selected and a *single* local SGD update occurs in each round[1].

We first claim some commonly adopted assumptions used in literature, e.g., [5], [10].

Assumption 1 *Each function f_i is L-smooth, that is, for any $x, y \in \mathbb{R}^d$, we have*

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|.$$

Assumption 2 *The function f is μ-strongly convex, that is, for any $x, y \in \mathbb{R}^d$, we have*

$$\langle \nabla f(x) - \nabla f(y), x - y \rangle \geq \mu \|x - y\|^2.$$

Assumption 3 *The quantizer Q_i is unbiased and the expected quantization error is bounded, that is,*

$$\mathbb{E}[Q_i(x)] = x,$$

and

$$\|Q_i(x) - x\|^2 \leq q_i \|x\|^2. \quad (6)$$

[1]In the experiment section, we simulate partial clients participation and multiple local SGD updates scenarios.

Assumption 4 *The second moment of stochastic gradient for all function f_i is bounded, that is,*

$$\mathbb{E}\|\nabla f_i(x)\|^2 \leq \sigma^2.$$

Assumption 1 means that the gradient of f_i is Lipschitz continuous. In this subsection, the loss function is assumed to be strongly convex as stated in Assumption 2, while this assumption no longer holds for the non-convex analysis in the next subsection. Assumption 3 is saying that the quantization is an unbiased estimation of the input, while the quantization error is bounded by some constant multiplying the norm of the input. Notice that there are many quantizers that satisfy this assumption. The BFP quantization method is an example. Assumption 4 implies that there is a uniform bound for the variance of stochastic gradients.

We present the main theorem on the convergence of FedHQ as follows. All proofs are omitted due to the space limitation.

Theorem 1 *Assume all assumptions hold. Denote x^* as the optimal solution to Problem* (4)*. If the stepsize $\eta_t = \frac{1}{\mu t}$, there exists a constant t_0, such that for any $t > t_0$, we have*

$$\mathbb{E}\|x_t - x^*\|^2 \leq \frac{t_0}{t}\mathbb{E}\|x_{t_0} - x^*\|^2 + \frac{C_0}{t},$$

where

$$C_0 \triangleq \frac{n\sigma^2(\sum_i p_i^2(1+q_i))}{\mu^2}.$$

Remark 1 Theorem 1 shows that FEDHQ converges to the optimal model at rate $O(1/T)$, which is of the same order as the convergence rate of conventional parallel SGD without quantization [11]. This implies that quantization does not fundamentally change the scaling behavior of convergence.

Remark 2 Notice that Theorem 1 still holds if we relax condition (6) in Assumption 3 to the average case:

$$\frac{\mathbb{E}\left[\|Q_i(x) - x\|^2 | x\right]}{\|x\|^2} \leq q_i. \tag{7}$$

Notice that the convergence rate upper bound includes a parameter C_0, which is a function of assigned weights p_i and quantization error q_i. We can optimize C_0 as presented in the following theorem.

Theorem 2 *Consider optimizing the upper bound of the convergence rate in Theorem 1 given the quantization error q_i, the optimal weight allocation scheme is given by*

$$p_i^{\mathsf{C}} = \frac{\frac{1}{(1+q_i)}}{\sum_i \frac{1}{(1+q_i)}}. \tag{8}$$

Proof Sketch: We aim to minimize the upper bound of the convergence rate, that is, to minimize C_0. It is equivalent to minimizing $\sum_i p_i^2(1+q_i)$. Thus, the optimization problem can be written as follows:

$$\begin{aligned} \min_{p_i} \ & \sum_i p_i^2(1+q_i) \\ s.t. \ & \sum_i p_i = 1, \quad p_i \geq 0. \end{aligned} \tag{9}$$

It can be seen that Eqn. (9) is a convex optimization problem, which we use KKT condition to solve.

Remark 3 The optimal weight assignment in Theorem 2 admits a *closed-form* expression in Eqn. (8), which is appealing both theoretically (as it allows for gaining insight into its structure) and practically (as it enables simple computations).

Remark 4 Theorem 2 shows that the optimal weight $p_{n_1} > p_{n_2}$ if $q_{n_1} < q_{n_2}$, that is, the clients with higher quantization precision should be assigned with higher weights, which coincides with the intuition.

Remark 5 Theorem 2 also shows that neither assigning equal weight to all clients nor assigning all weight to the clients with highest quantization precision is an optimal choice. There exists a fundamental tradeoff that balances the coverage and preference for clients with different quantization precision.

IV. THE FEDHQ+ ALGORITHM

In FEDHQ, we assume that the weight p_i is a prefixed parameter whose optimization depends on the *statistics* of quantization errors. A further enhancement is to make the weight choice *adaptive*, i.e., allowing it to change over time. Doing so would allow the dynamic weight to better reflect different stages of the model training, and the instantaneous quantization error $q_i(t)$ may vary significantly over learning rounds, especially during the early stages. This inspires us to propose FEDHQ+ in Algorithm 2, which is an enhanced FEDHQ algorithm that enables dynamic weights $p_i(t)$ based on instantaneous $q_i(t)$.

The main advantages of the dynamic weight $p_i(t)$ design in FEDHQ+ are twofold.

1) This helps avoid a mismatched prefixed p_i. If the quantization errors are not well estimated ahead of time, a poorly selected p_i may lead to underperformance.
2) The weights for all clients can be properly adjusted based on their instantaneous quantization errors. Even for clients with the same quantization level, their instantaneous quantization errors can be very different. For example, a client A with coarse quantization level should be assigned with a larger weight than a client B with fine quantization level, if the instantaneous quantization error for A is smaller. This can avoid bias towards some certain clients, and thus improve the algorithm efficiency.

V. EXPERIMENT

We perform two image classification tasks on MNIST and CIFAR10, to verify the performance of FEDHQ+. For both

Algorithm 2: The FEDHQ+ Algorithm

for $t = 1$ **to** T **do**
  Server randomly selects $\mathcal{S}_t$, and notifies all clients x_t in $\mathcal{S}_t$;
  for *node* $i \in \mathcal{S}_t$ **do**
    $x^i_{t,0} \leftarrow x_t$;
    for $\tau = 0$ **to** $K-1$ **do**
      $x^i_{t,\tau+1} = x^i_{t,\tau} - \eta_t \nabla f_i(x^i_{t,\tau})$;
    end
    $x^i_{t+1} \leftarrow x^i_{t,K}$;
    Computes $Q_i(x^i_{t+1} - x_t)$;
    Computes the current quantization error $q_i(t) = \max \frac{\|Q_i(x^i_{t+1}-x_t)-(x^i_{t+1}-x_t)\|^2}{\|x^i_{t+1}-x_t\|^2}$;
    Sends both $Q_i(x^i_{t+1} - x_t)$ and $q_i(t)$ to the server;
  end
  Server computes $p_i(t)$ from $q_i(t)$;
  Server updates $x_{t+1} = x_t + \sum_{i=1}^n p_i(t) Q_i(x^i_{t+1} - x_t))$;
end

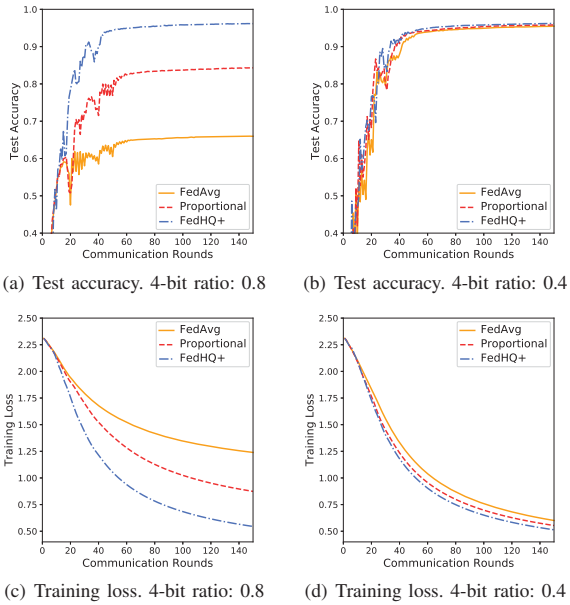
tasks, we assume that there are two categories of clients using the BFP quantization method: one has fine quantization level with $W = 8, F = 8$, and the other has coarse quantization level with $W = 4, F = 4$. Besides FEDHQ(+), we also implement two baseline schemes for comparison. The first is the well-known FEDAVG, which assigns the same weights for all clients. The second scheme is a heuristic extension of FEDAVG, which we call PROPORTIONAL, where each client is assigned a weight that is linearly proportional to its quantization bits W (e.g., in our setting, the weight for fine-quantized clients is twice as large as the one for the coarse-quantized one).

A. MNIST

For MNIST, we train the same CNN model as in [2]. A total of 100 clients participate the FL. At each round, all clients are selected and they run SGD locally with batch size $B = 600$ and epoch $K = 1$. We evaluate both IID and non-IID partitioning of the total MNIST data over clients in a way that is the same as [2].

IID partition. Figure 1 gives a detailed view of the convergence by plotting the test accuracy and training loss versus the number of communication rounds for two scenarios when the ratio of 4-bit quantization clients is 0.8 and 0.4. We can see that FEDAVG has the worst performance, which means aggregating heterogeneous clients' updates using the same weight leads to high performance loss. Furthermore, FEDHQ+ outperforms the other two heuristic schemes under all scenarios, as it adjusts the allocated weights judiciously.

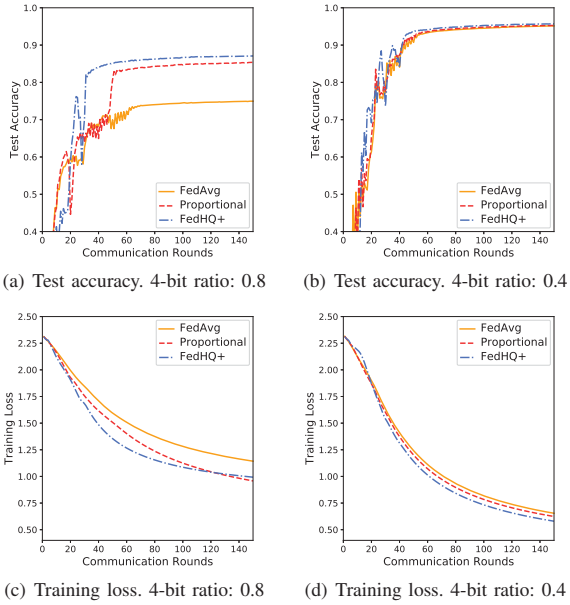
Non-IID partition. Figure 2 displays the test accuracy and training loss versus the number of communication rounds for two scenarios when the ratio of 4-bit quantization clients is 0.8 and 0.4, respectively. Similarly, FEDHQ+ has the best performance among these three schemes under all scenarios.



(a) Test accuracy. 4-bit ratio: 0.8 (b) Test accuracy. 4-bit ratio: 0.4

(c) Training loss. 4-bit ratio: 0.8 (d) Training loss. 4-bit ratio: 0.4

Fig. 1. Test accuracy and training loss versus communication rounds. IID partition on MNIST.



(a) Test accuracy. 4-bit ratio: 0.8 (b) Test accuracy. 4-bit ratio: 0.4

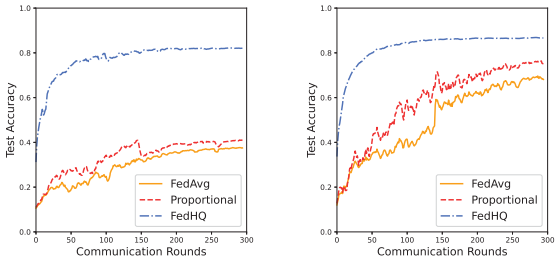
(c) Training loss. 4-bit ratio: 0.8 (d) Training loss. 4-bit ratio: 0.4

Fig. 2. Test accuracy and training loss versus communication rounds. Non-IID partition on MNIST.

Similarly, we can observe that FEDHQ+ performs universally better than the other two aggregation schemes under the non-IID partition.

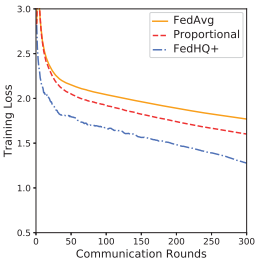
B. CIFAR10

To check whether the previous observations are generalizable, we also verify the algorithms on CIFAR10. At each

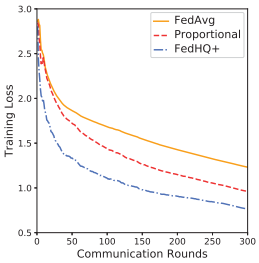


(a) Test accuracy. 4-bit:0.6

(b) Test accuracy. 4-bit:0.3

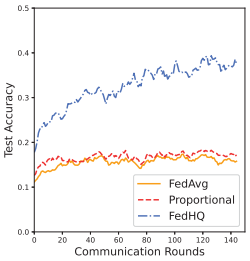


(c) Training loss. 4-bit:0.6

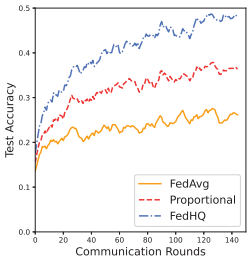


(d) Training loss. 4-bit:0.3

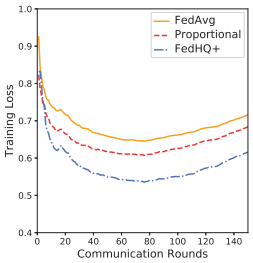
Fig. 3. Test accuracy and training loss versus communication rounds. IID partition on CIFAR10.



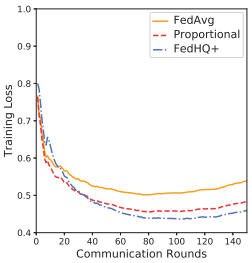
(a) Test accuracy. 4-bit:0.6



(b) Test accuracy. 4-bit:0.3



(c) Training loss. 4-bit:0.6



(d) Training loss. 4-bit:0.3

Fig. 4. Test accuracy and training loss versus communication rounds, Non-IID partition on CIFAR10.

round, $C = 10\%$ of the total clients are randomly selected to participate FL. Each selected client runs SGD locally for 5 epochs (i.e., $K = 5$), and then uploads the update to the server.

IID partition. We adopt ResNet18 in [12] with a batch size $B = 128$. Figure 3 (a) and (b) show the test accuracy versus the number of communication rounds for 4-bit quantization clients ratios of 0.6 and 0.3, respectively. Figure 3 (c) and (d) plot the corresponding training loss versus the number of communication rounds for these two scenarios. We can see that FEDHQ+ outperforms the other two heuristic schemes.

Non-IID partition. We use VGG11 in [13] with a batch size $B = 64$. Figure 4 shows the test accuracy and training loss versus the communication rounds for 4-bit quantization clients ratios of 0.6 and 0.3, respectively. Similarly, FEDHQ+ has the best performance among all methods.

VI. CONCLUSION

In the paper, we have investigated how to aggregate client updates with heterogeneous quantization precision levels in federated learning. We proposed FEDHQ/FEDHQ+, which computes the weights by optimizing the convergence rate upper bound as a function of the statistical/instantaneous quantization errors. The result suggested that clients with small quantization errors should receive larger weights, and all clients should contribute to the model updating regardless of their quantization errors. The simulation results showed that FEDHQ+ outperforms the well-known FEDAVG which allocates same weights to all clients, and a heuristic scheme that assigns weights linearly proportional to the clients' quantization bits.

REFERENCES

[1] J. Konecný *et al.*, "Federated learning: Strategies for improving communication efficiency," in *NIPS Workshop on Private Multi-Party Machine Learning*, 2016.

[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of AISTATS*, Apr. 2017, pp. 1273–1282.

[3] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-IID data," in *ICLR*, 2020.

[4] J. Konecný, "Stochastic, distributed and federated optimization for machine learning," *CoRR*, vol. abs/1707.01155, 2017.

[5] A. Reisizadeh, A. Mokhtari, H. Hassani, and R. Pedarsani, "An exact quantized decentralized gradient descent algorithm," *IEEE Trans. Signal Processing*, vol. 67, no. 19, pp. 4934–4947, 2019.

[6] X. Dai, X. Yan, K. Zhou, H. Yang, K. K. W. Ng, J. Cheng, and Y. Fan, "Hyper-sphere quantization: Communication-efficient sgd for federated learning," *ArXiv*, vol. abs/1911.04655, 2019.

[7] Y. Yu, J. Wu, and L. Huang, "Double quantization for communication-efficient distributed optimization," in *Advances in Neural Information Processing Systems*, 2019, pp. 4440–4451.

[8] A. K. Sahu, T. Li, M. Sanjabi, M. Zaheer, A. Talwalkar, and V. Smith, "On the convergence of federated optimization in heterogeneous networks," *CoRR*, vol. abs/1812.06127, 2018.

[9] G. Yang, T. Zhang, P. Kirichenko, J. Bai, A. G. Wilson, and C. D. Sa, "SWALP : Stochastic weight averaging in low-precision training," *CoRR*, vol. abs/1904.11943, 2019.

[10] P. Jiang and G. Agrawal, "A linear speedup analysis of distributed deep learning with sparse and quantized communication," in *Advances in Neural Information Processing Systems*, 2018, pp. 2525–2536.

[11] S. U. Stich, "Local SGD converges fast and communicates little," in *ICLR*, 2018.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[13] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2015.