Q-GADMM: QUANTIZED GROUP ADMM FOR COMMUNICATION EFFICIENT DECENTRALIZED MACHINE LEARNING

Anis Elgabli, Jihong Park, Amrit S. Bedi†, Mehdi Bennis, and Vaneet Aggarwal‡

ABSTRACT

In this paper, we propose a communication-efficient decentralized machine learning (ML) algorithm, coined quantized group ADMM (Q-GADMM). Every worker in Q-GADMM communicates only with two neighbors, and updates its model via the alternating direct method of multiplier (ADMM), thereby ensuring fast convergence while reducing the number of communication rounds. Furthermore, each worker quantizes its model updates before transmissions, thereby decreasing the communication payload sizes. We prove that Q-GADMM converges for convex loss functions, and numerically show that Q-GADMM yields 7x less communication cost while achieving almost the same accuracy and convergence speed compared to a baseline without quantization, group ADMM (GADMM).

Index Terms— Communication-efficient decentralized machine learning, GADMM, ADMM, quantization.

1. INTRODUCTION

Recently, distributed machine learning (ML) at the network edge has received significant attention [1-3]. In contrast to classical cloud-based ML, distributed ML hinges on wireless communication and network dynamics, whereby communication may hinder its performance. To mitigate this bottleneck, one can decrease the communication cost of distributed ML by reducing the number of communication rounds until convergence, communication links per round, and/or the payload size per link.

Specifically, to reduce the communication payload sizes, arithmetic precision of the model parameters can be decreased by for instance 1-bit gradient quantization [4], multi-bit gradient quantization [5], or weight quantization with random rotation [6]. Alternatively, instead of model parameters, model outputs can be exchanged for large models via knowledge distillation [7][8]. To reduce communication links, model updates can be sparsified by collecting the updates until a time deadline [9], upon the values sufficiently changed from the preceding updates [5][10], or based on channel conditions [11][13]. To reduce the number of communication rounds, convergence speed can be accelerated via collaboratively adjusting the training momentum [14][15]. However, these methods commonly postulate a central server that collects model update information, which may not be scalable.

In decentralized ML architectures without any central entity, communication payload sizes can be reduced by a quantized weight gossiping algorithm [16]. On the other hand, both communication links and rounds can be decreased using the group ADMM (GADMM) algorithm proposed in our prior work [17]. Spurred by these preceeding works, in this article we further integrate quantization into GADMM, and propose a communication-efficient decentralized ML algorithm, coined quantized group ADMM (Q-GADMM), thereby reducing communication rounds, links, and payload sizes altogether.

Quantized Group GADMM (Q-GADMM). As shown in Fig. 1, Q-GADMM divides workers into head and tail groups as done in GADMM. For less communication rounds via faster convergence, the workers in the same group update their models in parallel, whereas the workers in different groups update the models in an alternating way using ADMM [18]. Every alternation entails a single communication round, in which each worker communicates only with two neighboring workers in the opposite group, reducing communication links. Lastly, before every transmission, the workers quantize their model parameters, decreasing communication payload sizes.
Contributions. It is non-trivial to prove the convergence of Q-GADMM, in which quantization errors may lead to high unintended variance in model parameter updates. Its neighbor-based communications aggravate this problem, since the errors may easily propagate across iterations. To mitigate these problems, in this article we propose a novel stochastic quantization scheme that ensures unbiased model updates and non-increasing quantization step sizes, by adjusting the quantization levels and probabilities over iterations. We thereby prove that with exchanging the maximum range of quantization and the number of quantization levels, Q-GADMM achieves the convergence and the optimality of GADMM for convex, closed, and proper functions. Numerical results show that Q-GADMM converges as fast as GADMM with up to 7x less communication cost than GADMM.

2. PROBLEM FORMULATION

We consider a set of \(N\) workers storing their local batch of input samples. The \(n\)-th worker has its model vector \(\theta_n \in \mathbb{R}^d\), and aims to learn a global model vector \(\theta\) that solves the following decentralized learning problem:

\[
\begin{align*}
\text{Minimize} & \quad \sum_{n=1}^{N} f_n(\theta_n) \\
\text{subject to} & \quad \theta_n = \theta_{n+1}, \quad \forall n = 1, \ldots, N - 1.
\end{align*}
\]

(1)

The constraint in (1) dictates that \(\theta_{n-1} = \theta_n \) and \(\theta_n = \theta_{n+1}, \forall n\), making worker \(n\) have joint constraints with only two neighbors. In our prior work [17], GADMM was proposed to solve (1) in a decentralized manner. To this end, workers are split into two groups, heads and tails, such that each worker in the head (or tail) group is communicating with two tail (or head) workers. As Fig. 1 illustrates, the primal variables, i.e., model vectors, of head workers are updated in parallel, and then downloaded by their neighboring tail workers. Likewise, the primal variables of tail workers are updated in parallel, and downloaded by their neighboring head workers. Lastly, the dual variables are updated locally at each worker. To improve communication efficiency, quantization will be introduced to GADMM in the next section.

3. PROPOSED ALGORITHM: Q-GADMM

In this section, we propose a stochastic quantization scheme that guarantees the convergence of Q-GADMM, and then describe the overall operational procedures of Q-GADMM. Following [5], worker \(n\) in Q-GADMM at iteration \(k\) quantizes its model vector \(\hat{\theta}_n\) as \(\theta_n = Q_n(\hat{\theta}_n, \hat{\theta}_n^{k-1})\), based on its previously quantized model vector \(\hat{\theta}_n^{k-1}\). The function \(Q_n(\cdot)\) is a stochastic quantization operator that depends on the quantization probability \(p_{n,i}^k\) for each model vector’s dimension \(i \in \{1, 2, \cdots, d\}\), and on \(b_n^k\) bits used for representing each model vector dimension. To ensure the convergence of Q-GADMM, \(p_{n,i}^k\) and \(b_n^k\) should be properly chosen as detailed next.

Stochastic Quantization. As Fig. 1 shows, the \(i\)-th dimensional element \([\hat{\theta}_n^{k-1}]_i\) of the previously quantized model vector is centered at the quantization range \(2R_n^k\) that are equally divided into \(2^{b_n^k} - 1\) quantization levels, yielding the quantization step size \(\Delta_n^k = (2^{b_n^k} - 1)/(2R_n^k)\). In this coordinate, the difference between the \(i\)-th dimensional element \([\theta_n^k]_i\) of the current model vector and \([\hat{\theta}_n^{k-1}]_i\) is drawn at

\[
[c_n(\theta_n^k)]_i = \Delta_n^k (\hat{\theta}_n^k)_i - [\hat{\theta}_n^{k-1}]_i + R_n^k.
\]

(2)

Here, adding \(R_n^k\) ensures the non-negativity of the quantized value. Then, \([c_n(\theta_n^k)]_i\) is mapped to:

\[
[q_n(\theta_n^k)]_i = \begin{cases} 
[c_n(\theta_n^k)]_i & \text{with probability } p_{n,i}^k, \\
[c_n(\theta_n^k)]_i & \text{otherwise},
\end{cases}
\]

(3)

where \([\cdot]^+\) and \([\cdot]_0\) are ceiling and floor functions, respectively.

Next, to maintain the model parameter unbiased after quantization, \(p_{n,i}^k\) is chosen such that the expected quantization error \(\mathbb{E}\{\epsilon_n^k\} \) becomes zero, i.e.,

\[
p_{n,i}^k ([c_n(\theta_n^k)]_i - [c_n(\theta_n^k)]_i) + (1 - p_{n,i}^k) ([c_n(\theta_n^k)]_i - [c_n(\theta_n^k)]_i) = 0.
\]

(4)

Consequently, \(p_{n,i}^k\) is obtained as:

\[
p_{n,i}^k = \frac{[c_n(\theta_n^k)]_i - [c_n(\theta_n^k)]_i}{\Delta_n^k},
\]

(5)

where the denominator follows from \(\Delta_n^k = |[c_n(\theta_n^k)]_i - [c_n(\theta_n^k)]_i|\). With \(p_{n,i}^k\) in (5), quantization error variance is bounded, i.e., \(\mathbb{E}\{\epsilon_n^k\} \leq (\Delta_n^k)^2/4\).

In addition to the above, the convergence of Q-GADMM requires non-increasing quantization step sizes over iterations, i.e., \(\Delta_n^k \leq \Delta_n^{k-1} \forall k\). To satisfy this condition, \(b_n^k\) is chosen as:

\[
b_n^k = \left\lfloor \log_2 \left(1 + (2^{b_n^{k-1}} - 1)R_n^k/R_n^{k-1} \right) \right\rfloor.
\]

(6)

Given \(p_{n,i}^k\) in (5) and \(b_n^k\) in (6), the convergence of Q-GADMM is to be proven in Sec. 4. Note that in our numerical simulations in Sec. 5, we observe that \(R_n^k\) decreases over iterations, and thus \(\Delta_n^k \leq \Delta_n^{k-1}\) holds even for \(b_n^k\) fixed as a constant.

With the aforementioned stochastic quantization procedure, \(b_n^k, R_n^k,\) and \(q_n(\theta_n^k) = ([q_n(\theta_n^k)]_1, [q_n(\theta_n^k)]_2, \cdots, [q_n(\theta_n^k)]_d)\) suffice to represent \(\hat{\theta}_n\), which are transmitted to neighbors. After receiving these values, \(\hat{\theta}_n\) can be reconstructed as follows:

\[
\hat{\theta}_n = \theta_n^k + \Delta_n^k q_n(\theta_n^k) - R_n^k 1.
\]

(7)

Consequently, when the full arithmetic precision uses 32bit, every transmission payload size of Q-GADMM is \(b_n^k d + (b_R + b_b)\) bits, where \(b_R \leq 32\) and \(b_b \leq 32\) are the required bits to represent \(R_n^k\) and \(b_n^k\), respectively. Compared to GADMM whose payload size is \(32d\) bits, Q-GADMM can achieve a huge reduction in communication overhead by setting \(b_n^k \ll 32\), particularly for large models, i.e., large \(d\).
Algorithm 1 Quantized Group ADMM (Q-GADMM)

1: Input: \( N, f_n(\theta_n)\forall n, \rho, K \). Output: \( \theta_n, \forall n \)
2: Initialization: \( \theta_0^{(0)} = 0, \lambda_0^{(0)} = 0, \forall n \)
3: \( \mathcal{N}_h = \{\theta_1, \theta_3, \ldots, \theta_{N-1}\}, \mathcal{N}_t = \{\theta_2, \theta_4, \ldots, \theta_N\} \)
4: while \( k \leq K \) do
5:  head workers (\( n \in \mathcal{N}_h \)): in Parallel
6:  Reconstruct \( \theta_{n-1}^{(k)} \) and \( \theta_{n+1}^{(k)} \) via (7), and update \( \theta_n \) via (12)
7:  Choose \( p_{n,i} \) via (5) and \( b_n^{(k)} \) via (6)
8:  Quantize \( \theta_n \) via (9)
9:  Transmit \( b_n^{(k)}, R_n, q_n(\theta_n) \) to its two tail neighbors
10: tail workers (\( n \in \mathcal{N}_t \)): in Parallel
11: Reconstruct \( \theta_{n-1}^{(k)} \) and \( \theta_{n+1}^{(k)} \) via (7), and update \( \theta_n \) via (13)
12: Choose \( p_{n,i} \) via (5) and \( b_n^{(k)} \) via (6)
13: Quantize \( \theta_n \) via (9)
14: Transmit \( b_n^{(k)}, R_n, q_n(\theta_n) \) to its two head neighbors
15: all workers (\( n \in \{1, \ldots, N\} \)): in Parallel
16: Update \( \lambda_{n-1} \) and \( \lambda_n^{(k)} \) locally via (14)
17: \( k \leftarrow k + 1 \)
18: end while

Q-GADMM Operations. For the sake of explanation, we consider an even \( N \) number of workers. We start by writing the augmented Lagrangian as:

\[
\mathcal{L}_p = \sum_{n=1}^{N} f_n(\theta_n) + \sum_{n=1}^{N-1} \langle \lambda_{n-1}, \theta_n - \theta_{n+1} \rangle + \frac{\rho}{2} \sum_{n=1}^{N} \| \theta_n - \theta_{n+1} \|^2_2. \tag{11}
\]

Let \( \mathcal{N}_h = \{\theta_1, \theta_3, \ldots, \theta_{N-1}\} \), and \( \mathcal{N}_t = \{\theta_2, \theta_4, \ldots, \theta_N\} \) denotes the sets of head and tail workers, respectively. At iteration \( k + 1 \), the workers’ primal and dual variables are updated as follows. First, head worker’s primal variables are updated as:

\[
\begin{align*}
\theta_{n}^{(k+1)} &= \arg\min_{\theta_n} \left\{ f_n(\theta_n) + \langle \lambda_{n-1}^{(k)}, \theta_n - \theta_{n-1} \rangle \right. \\
&\quad + \langle \lambda_{n}^{(k)}, \theta_n - \theta_{n+1}^{(k)} \rangle + \frac{\rho}{2} \left\| \theta_n - \theta_{n-1} \right\|^2_2 + \frac{\rho}{2} \left\| \theta_n - \theta_{n+1}^{(k)} \right\|^2_2. \tag{12}
\end{align*}
\]

Next, each head worker transmits its quantized model to its two tail neighbors. Then, the tail workers’ primal variables are update as:

\[
\begin{align*}
\theta_{n}^{(k+1)} &= \arg\min_{\theta_n} \left\{ f_n(\theta_n) + \langle \lambda_{n-1}^{(k)}, \theta_n - \theta_{n-1} \rangle \right. \\
&\quad + \langle \lambda_{n}^{(k)}, \theta_n - \theta_{n+1}^{(k)} \rangle + \frac{\rho}{2} \left\| \theta_n - \theta_{n-1} \right\|^2_2 + \frac{\rho}{2} \left\| \theta_n - \theta_{n+1}^{(k)} \right\|^2_2. \tag{13}
\end{align*}
\]

Finally, every worker locally updates the dual variables \( \lambda_{n-1} \) and \( \lambda_n \) as follows:

\[
\lambda_{n+1} = \lambda_{n-1} + \rho(\theta_{n-1}^{(k)} - \theta_{n}^{(k)}), \tag{14}
\]

The aforementioned operations are summarized in Algorithm 1.

4. CONVERGENCE ANALYSIS

In this section, we prove the optimality and convergence of Q-GADMM for convex functions. The necessary and sufficient optimality conditions are the primal and dual feasibility which are defined by the following two equations:

\[
\theta_n^{\star} = \theta_n^{(k-1)}, \forall n > 1, \quad 0 \in \partial f_n(\theta_n^{\star}) - \lambda_n^{\star} - \lambda_n^{(k-1)}, \forall n. \tag{15}
\]

First, at iteration \( k + 1 \), every \( \theta_n^{(k+1)} \) such that \( n \in \mathcal{N}_t \) minimizes (13), which implies that

\[
0 \in \partial f_n(\theta_n^{(k+1)}) - \lambda_n^{(k-1)} + \rho(\theta_n^{(k+1)} - \theta_n^{(k-1)}) + \rho(\theta_n^{(k+1)} - \theta_n^{(k+1)}). \tag{16}
\]

Let \( \varepsilon_n^{(k+1)} \) be the quantization error at iteration \( k + 1 \). Hence, \( \varepsilon_n^{(k+1)} = \theta_n^{(k+1)} - \theta_n^{(k-1)} \). Moreover, given that \( \lambda_n^{(k+1)} = \lambda_n^{(k)} + \rho(\theta_n^{(k+1)} - \theta_n^{(k-1)}) \), (16) can be re-written as

\[
0 \in \partial f_n(\theta_n^{(k+1)}) - \lambda_n^{(k-1)} + \lambda_n^{(k)} + 2\rho\varepsilon_n^{(k+1)}. \tag{17}
\]

Second, every \( \theta_n^{(k+1)} \) such that \( n \in \mathcal{N}_h \) minimizes (12) at iteration \( k + 1 \). Therefore

\[
0 \in \partial f_n(\theta_n^{(k+1)}) - \lambda_n^{(k-1)} + \lambda_n^{(k)} + 2\rho\varepsilon_n^{(k+1)} + \rho(\theta_n^{(k+1)} - \theta_n^{(k-1)}). \tag{18}
\]

Let \( \lambda_n^{(k+1)} = \theta_n^{(k+1)} - \theta_n^{(k-1)} \) and \( \varepsilon_n^{(k+1)} = \theta_n^{(k+1)} - \theta_n^{(k+1)} \) be the primal residual of each agent \( n \), and \( r_n^{(k+1)} \) be the dual residual of worker \( n \in \mathcal{N}_h \) at iteration \( k + 1 \). To prove the convergence of the proposed algorithms, we first provides the upper and lower bounds on the optimality gap in Lemma 1. Then we prove the decrement of the Lyapunov function in Lemma 2 and finally present the convergence result in Theorem 1.

**Lemma 1** At \( k + 1 \) iteration of Q-GADMM, the optimality gap satisfies \( LB_1 \leq \mathbb{E} \{ \sum_{n=1}^{N} f_n(\theta_n^{(k+1)}) - \sum_{n=1}^{N} f_n(\theta_n^{(k)}) \} \leq UB_1 \), where \( LB_1 \) and \( UB_1 \) are given in (20) and (21), respectively, at the bottom of this page.

Before we introduce the next lemma, we define a Lyapunov function \( V_k = \sum_{n=1}^{N} (1/\rho)\| \lambda_n^{(k)} - \lambda_n^{(k-1)} \|^2_2 + \sum_{n \in \mathcal{N}_h} \rho\| \theta_n^{(k)} - \theta_n^{(k-1)} \|^2_2 + \rho\| \theta_n^{(k)} - \theta_n^{(k-1)} \|^2_2 \). Next lemma shows that \( \mathbb{E} \{ V_k - V_{k+1} \} \) is bounded above. This property is then used in Theorem 1 to prove that \( V_k \) is monotonically decreasing at each iteration \( k \) of the proposed algorithm and the primal residual go to zero as \( k \to \infty \) when the quantization step size is reduced as \( k \) increases.

\[
LB_1 = \mathbb{E} \left\{ \sum_{n=1}^{N-1} \langle \lambda_n^{(k-1)}, \theta_{n+1}^{(k+1)} - \theta_n^{(k+1)} \rangle \right\} \tag{20}
\]

\[
UB_1 = \mathbb{E} \left\{ - \sum_{n=1}^{N-1} \langle \lambda_n^{(k+1)}, \theta_{n+1}^{(k+1)} - \theta_n^{(k+1)} \rangle + \sum_{n=1}^{N} \langle \theta_n^{(k+1)}, \theta_n^{(k+1)} - \theta_n^{(k+1)} \rangle \right\} \tag{21}
\]

\[
H_k = \mathbb{E} \left\{ \rho\| \theta_n^{(k+1)} - \theta_n^{(k)} \|^2_2 - \rho\| \theta_n^{(k+1)} - \theta_n^{(k)} \|^2_2 - \rho\| \theta_n^{(k+1)} - \theta_n^{(k)} \|^2_2 \right\} \tag{22}
\]
Lemma 2 When $f_n(\theta_n)$ is closed, proper, and convex, and the Lagrangian $\mathcal{L}_0$ has a saddle point, then the following inequality holds true at the $(k+1)$-th iteration of Q-GADMM:

$$
\mathbb{E}\{V_k - V_{k+1}\} \geq \rho \sum_n \mathbb{E}\{\|c_n^{(k+1)} - c_n^{(k+1)}\|^2\} - \frac{\rho d}{2} \sum_n \Delta_n^{(k+1)} + H_v,
$$

where $H_v$ is given in (22) at the bottom of the previous page.

Using Lemmas 1 and 2 we derive our main theorem stating the convergence and the optimality of Q-GADMM in solving (1).

Theorem 1 For non-increasing quantization step sizes, i.e., $\Delta_n^k \leq \Delta_n^{(k-1)} \forall k$, and under the assumption of Lemma 2 as $k \to \infty$, the primal and dual residual converges to 0 with probability 1, i.e., $\lim_{k \to \infty} r_{n, n+1}^k = 0$ and $\lim_{k \to \infty} s_{n, n}^k = 0$. Furthermore, the optimality gap converges to 0 with probability 1, i.e., $\lim_{k \to \infty} \sum_{n=1}^N \mathbb{E}\{f_n(\theta_n^k)\} = \sum_{n=1}^N f_n(\theta^*)$.

Intuitively, when $\Delta_n^k$ is non-increasing, the RHS of (23) is always positive. For such positive $\mathbb{E}\{V_k - V_{k+1}\}$, iteration $k + 1$ is one more step towards the optimal solution. Therefore, following the same proof of Theorem 1 for GADMM [17], as $k \to \infty$, Q-GADMM converges to the optimal solution. The detailed proofs of lemmas and the theorem are omitted due to the lack of space.

5. NUMERICAL RESULTS

To validate the performance of Q-GADMM and to verify its communication efficiency compared to GADMM, we conducted experiments for decentralized machine learning tasks. In particular, we evaluated the performance of Q-GADMM for decentralized linear regression using the California Housing dataset [19].

We used 4000 samples for training, and we uniformly distributed them across 10 workers. To benchmark Q-GADMM, we compare it with GADMM [17], GD, and quantized GD (QGD) [5]. In GD, each worker computes its gradient locally and then sends it to a parameter server. The server updates the global model using aggregated gradient descent and broadcasts it to all workers. In QGD, each worker sends a quantized version of its local model. Therefore, in GD and QGD, each iteration involves $N+1$ communication rounds. On the other hand, for GADMM and QGADMM, each iteration involves $N$ communication rounds since each worker needs to broadcast its updated model. Finally, for quantized versions of GADMM and GD, we assume that $b_n^k = 2$ for all $n$ and $k$, so the number of quantization levels is 4, and it remains constant over iterations and across workers. Finally, we choose $\rho = 1$ for all experiments.

Figure 2(a) shows the convergence speed in the objective value for the linear regression task. Clearly, Figure 2(a) verifies Theorem 1 and shows the convergence of Q-GADMM for convex loss function with the same speed as GADMM. Moreover, Figure 2(b), shows that both GADMM and Q-GADMM-2bits achieves the loss of $10^{-5}$ with almost the same number of communication rounds (32600 for Q-GADMM-2bits vs 32560 for GADMM) which is around 50% less compared to GD and QGD-2bits. However, as shown in Figure 2(c), Q-GADMM-2bits requires significantly less number of bits as compared to GADMM (7x less number of bits). Moreover, compared to GD and QGD-2bits, Q-GADMM-2bits requires around 15x and 3x less number of transmitted bits to achieve the loss of $10^{-5}$. Thanks to the fast convergence inherited from GADMM, and our proposed stochastic quantization, the number of transmitted bits at every iteration are significantly reduced while ensuring unbiased and zero mean quantization error. Finally, it worth mentioning that in real scenarios, the communication cost to the parameter server is significantly higher than communicating with neighbors; hence savings in communication overhead by utilizing GADMM/Q-GADMM are more significant compared to GD/QGD.

6. CONCLUSIONS

This article proposed a communication-efficient decentralized ML algorithm, Q-GADMM. Compared to the original GADMM without quantization, Q-GADMM not only enjoys the same convergence rate as GADMM, but also is more communication-efficient. Numerical tests in a convex regularized linear regression task corroborate the advantages of Q-GADMM over GADMM, as well as GD and QGD.
7. REFERENCES

[1] J. Park, S. Samarakoon, M. Bennis, and M. Debbah, “Wireless network intelligence at the edge,” to appear in Proceedings of the IEEE [Online]. Early access is available at: https://ieeexplore.ieee.org/document/8865093, November 2019.

[2] J. Park, S. Wang, A. Elgabli, S. Oh, E. Jeong, H. Cha, H. Kim, S.-L. Kim, and M. Bennis, “Distilling on-device intelligence at the network edge,” Arxiv preprint, vol. abs/1908.05895, August 2019.

[3] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, “Federated learning: Challenges, methods, and future directions,” [Online]. ArXiv preprint: https://arxiv.org/abs/1908.07873, August 2019.

[4] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, “SignSGD: Compressed optimisation for non-convex problems,” In Proc. Intl. Conf. Machine Learn., Stockholm, Sweden, July 2018.

[5] J. Sun, T. Chen, G. B. Giannakis, and Z. Yang, “Communication-efficient distributed learning via lazily aggregated quantized gradients,” arXiv preprint arXiv:1909.07588 2019.

[6] J. Konecny, H. B. McMahan, F. X. Yu, P. Richtarik, A. T. Suresh, and D. Bacon, “Federated learning: strategies for improving communication efficiency,” in Proc. of NIPS Wksp. PMPML, Barcelona, Spain, December 2016. [Online]. Available: https://arxiv.org/abs/1610.05492

[7] E. Jeong, S. Oh, H. Kim, J. Park, M. Bennis, and S.-L. Kim, “Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data,” presented at Neural Information Processing Systems Workshop on Machine Learning on the Phone and other Consumer Devices (MLPCD), Montréal, Canada, 2018. [Online]. Available: http://arxiv.org/abs/1811.11479

[8] J. H. Ahn, O. Simeone, and J. Kang, “Wireless federated distillation for distributed edge learning with heterogeneous data,” [Online]. Arxiv preprint: http://arxiv.org/abs/1907.02745 May 2019.

[9] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, “Adaptive federated learning in resource constrained edge computing systems,” IEEE Journal on Selected Areas in Communications, vol. 37, no. 6, pp. 1205–1221, June 2019.

[10] T. Chen, G. Giannakis, T. Sun, and W. Yin, “Lag: Lazily aggregated gradient for communication-efficient distributed learning,” in Advances in Neural Information Processing Systems, 2018, pp. 5055–5065.

[11] T. Nishio and R. Yonetani, “Client selection for federated learning with heterogeneous resources in mobile edge,” In Proc. Int’l Conf. Commun. (ICC), Shanghai, China, May 2019. [Online]. Available: http://arxiv.org/abs/1804.08333

[12] H. H. Yang, Z. Liu, T. Q. S. Quek, and H. V. Poor, “Scheduling policies for federated learning in wireless networks,” arXiv preprint arXiv: 1908.06287, 2019.

[13] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, “A joint learning and communications framework for federated learning over wireless networks,” arXiv preprint arXiv: 1909.07972, 2019.

[14] W. Liu, L. Chen, Y. Chen, and W. Zhang, “Accelerating federated learning via momentum gradient descent,” arXiv preprint arXiv: 1910.03197, 2019.

[15] H. Yu, R. Jin, and S. Yang, “On the linear speedup analysis of communication efficient momentum SGD for distributed non-convex optimization,” In Proc. Intl. Conf. Machine Learn., Long Beach, CA, USA, June 2019.

[16] A. Koloskova, S. U. Stich, and M. Jaggi, “Decentralized stochastic optimization and gossip algorithms with compressed communication,” Proc. International Conference on Machine Learning (ICML), Long Beach, CA, USA, July 2019.

[17] A. Elgabli, J. Park, A. S. Bedi, M. Bennis, and V. Aggarwal, “Gadmm: Fast and communication efficient framework for distributed machine learning,” arXiv preprint arXiv:1909.00047 2019.

[18] R. Glowinski and A. Marroco, “Sur l’approximation, par éléments finis d’ordre un, et la résolution, par pénalisation-dualité d’une classe de problèmes de dirichlet non linéaires,” ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique, vol. 9, no. R2, pp. 41–76, 1975.

[19] L. Torgo, “Regression datasets,” 2014. [Online]. Available: https://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html