ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-7605-5/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICASSP39728.2021.9413697

ADAPTIVE QUANTIZATION OF MODEL UPDATES FOR COMMUNICATION-EFFICIENT FEDERATED LEARNING

Divyansh Jhunjhunwala[⋆] *Advait Gadhikar*[⋆] *Gauri Joshi*[⋆] *Yonina C. Eldar*[†]

[⋆] Carnegie Mellon University, Pittsburgh, USA, {djhunjhu, agadhika, gaurij}@andrew.cmu.edu
[†] Weizmann Institute of Science, Rehovot, Israel, {yonina.eldar@weizmann.ac.il}

ABSTRACT

Communication of model updates between client nodes and the central aggregating server is a major bottleneck in federated learning, especially in bandwidth-limited settings and high-dimensional models. Gradient quantization is an effective way of reducing the number of bits required to communicate each model update, albeit at the cost of having a higher error floor due to the higher variance of the stochastic gradients. In this work, we propose an adaptive quantization strategy called AdaQuantFL that aims to achieve communication efficiency as well as a low error floor by changing the number of quantization levels during the course of training. Experiments on training deep neural networks show that our method can converge in much fewer communicated bits as compared to fixed quantization level setups, with little or no impact on training and test accuracy.

Index Terms— distributed optimization, federated learning, adaptive quantization

1. INTRODUCTION

Distributed machine learning training, which was typically done in the data center setting, is rapidly transitioning to the Federated Learning (FL) setting [1] [2], where data is spread across a large number of mobile client devices. Due to privacy concerns, the FL clients perform on-device training and only share model updates with a central server. A major challenge in FL is the communication bottleneck due to the limited uplink bandwidth available to the clients.

Recent work tackling this problem has taken two major directions. The first approach reduces the load on the communication channel by allowing each client to perform multiple local updates [1, 3, 4, 5, 6], thus reducing the communication frequency between clients and server. However, this optimization may not be enough due to the large size of model updates for high dimensional models, like neural networks. The second approach deals with this problem by using compression methods to reduce the size of the model update being communicated by the clients at an update step [7, 8, 9, 10, 11, 12, 13]. However, such compression methods usually add to the error floor of the training objective as they increase the variance of the updates. Thus, one needs to carefully choose the number of quantization levels in order to strike the best error-communication trade-off.

In this work we propose AdaQuantFL, a strategy to automatically adapt the number of quantization levels used to represent a model update and achieve a low error floor as well as communication efficiency. The key idea behind our approach is that we bound the convergence of training error in terms of the number of bits communicated, unlike traditional approaches which bound error with respect to number of training rounds (see Fig. 1). We use this convergence analysis to adapt the number of quantization levels during training based on the current training loss. Our approach can be considered orthogonal to other proposed methods of adaptive compression such as varying the spacing between quantization levels [14] and reusing outdated gradients [15]. In [16], the authors propose an adaptive method for tuning the number of local updates or the communication frequency. AdaQuantFL is a similar strategy, but for tuning the number of bits communicated per round. Our experiments on distributed training of deep neural networks verify that AdaQuantFL is able to achieve a given target training loss using much fewer bits compared to fixed quantization methods.

2. SYSTEM MODEL

Consider a system of n clients and a central aggregating server. Each client i has a dataset $\mathcal{D}_i$ of size m_i consisting of labeled samples $\xi_j^{(i)} = (\mathbf{x}_j^{(i)}, y_j^{(i)})$ for $j = 1, \ldots, m_i$. The goal is to train a common global model, represented by the parameter vector $\mathbf{w} \in \mathbb{R}^d$, by minimizing the following objective function:

$$\min_{\mathbf{w}\in\mathbb{R}^d} \left[f(\mathbf{w}) = \sum_{i=1}^{n} p_i f_i(\mathbf{w}) = \sum_{i=1}^{n} p_i \frac{1}{m_i} \sum_{j=1}^{m_i} \ell(\mathbf{w}; \xi_j^{(i)}) \right], \tag{1}$$

where $p_i = \frac{m_i}{\sum_{i=1}^{n} m_i}$ is the fraction of data held at the i-th client and $f_i(\mathbf{w})$ is the empirical risk at the i-th client for a possibly non-convex loss function $\ell(\mathbf{w}; \xi_j^{(i)})$.

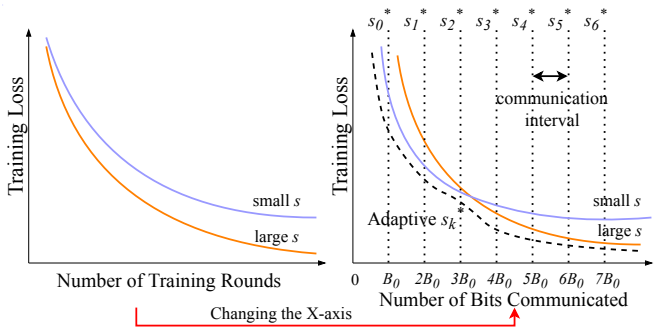


Fig. 1: Viewing training in terms of bits communicated.

Quantized Local SGD. The model is trained iteratively using the local stochastic gradient descent (local SGD) algorithm, proposed in [6, 3]. In local SGD, the entire training process is divided into rounds consisting of τ local updates at each client. At the beginning of the k-th round, each client reads the current global model $\mathbf{w}_k$ from the central server and updates it by performing τ local SGD steps for $t = 0, \cdots, \tau-1$ as follows:

$$\mathbf{w}_{k,t+1}^{(i)} = \mathbf{w}_{k,t}^{(i)} - \eta g_i(\mathbf{w}_{k,t}^{(i)}, \xi^{(i)}), \tag{2}$$

where $\mathbf{w}_{k,0}^{(i)} = \mathbf{w}_k$ and $g_i(\mathbf{w}_{k,t}^{(i)}, \xi^{(i)})$ is the stochastic gradient computed using a mini-batch $\xi^{(i)}$ sampled uniformly at random from the i-th client local dataset $\mathcal{D}_i$. After completing τ steps of local SGD, each client sends its update for the k-th round denoted by $\Delta\mathbf{w}_k^{(i)} = \mathbf{w}_{k,\tau}^{(i)} - \mathbf{w}_{k,0}^{(i)}$, to the central server. In order to save on bits communicated over the bandwidth-limited uplink channel, each client only sends a quantized update $Q(\Delta\mathbf{w}_k^{(i)})$, where $Q(\cdot)$ represents a stochastic quantization operator over $\mathbb{R}^d$. Once the server has received the quantized updates from all the clients, the global model is updated as follows.

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \sum_{i=1}^{n} p_i Q(\Delta\mathbf{w}_k^{(i)}). \tag{3}$$

Stochastic Uniform Quantizer. In this work we consider the commonly used [7, 17, 18] stochastic uniform quantization operator $Q_s(\mathbf{w})$, which is parameterized by the number of quantization levels $s \in \mathbb{N} = \{1, 2, \dots\}$. For each dimension of a d-dimensional parameter vector $\mathbf{w} = [w_1, \dots, w_d]$,

$$Q_s(w_i) = \|\mathbf{w}\|_2 \text{sign}(w_i)\zeta_i(\mathbf{w}, s), \tag{4}$$

where $\zeta_i(\mathbf{w}, s)$ is a random variable given as,

$$\zeta_i(\mathbf{w}, s) = \begin{cases} \frac{l+1}{s} & \text{with probability } \frac{|w_i|}{\|\mathbf{w}\|_2} s - l \\ \frac{l}{s} & \text{otherwise.} \end{cases} \tag{5}$$

Here, $l \in \{0, 1, 2, ..s-1\}$ is an integer such that $\frac{|w_i|}{\|\mathbf{w}\|_2} \in [\frac{l}{s}, \frac{l+1}{s})$. For $\mathbf{w} = \mathbf{0}$, we define $Q_s(\mathbf{w}) = \mathbf{0}$.

Given $Q_s(w_i)$, we need 1 bit to represent sign(w_i) and $\lceil \log_2(s+1) \rceil$ bits to represent $\zeta_i(\mathbf{w}, s)$. The scalar $\|\mathbf{w}\|_2$ is usually represented with full precision, which we assume to be 32 bits. Thus, the number of bits communicated by a client to the central server per round, which we denote by C_s, is given by

$$C_s = d\lceil \log_2(s+1) \rceil + d + 32. \tag{6}$$

It can be shown from the work of [7, 18] that while $Q_s(\mathbf{w})$ remains unbiased for all s, i.e., $\mathbb{E}[Q_s(\mathbf{w})|\mathbf{w}] = \mathbf{w}$, the variance of $Q_s(\mathbf{w})$ decreases with s because of the following variance upper bound:

$$\mathbb{E}[\|Q_s(\mathbf{w}) - \mathbf{w}\|_2^2 | \mathbf{w}] \leq \frac{d}{s^2} \|\mathbf{w}\|_2^2. \tag{7}$$

From (6) and (7), we see that varying s results in a trade-off between the total number of bits communicated C_s and the variance upper bound – C_s increases with s while the variance upper bound in (7) decreases with s. Building on this observation, in the next section, we analyze the effect of s on the error convergence speed and use it to design a strategy to adapt s during the course of training.

3. TRADE-OFF BETWEEN ERROR AND THE NUMBER OF BITS COMMUNICATED

The motivation behind adapting the number of quantization levels s during training can be understood through the illustration in Fig. 1. In the left plot, we see that a smaller s, that is, coarser quantization, results in worse convergence of training loss versus the number of training rounds. However, a smaller s reduces the number of bits C_s communicated per round. To account for this communication reduction, we change the x-axis to the number of bits communicated in the right plot of Fig. 1. This plot reveals that smaller s enables us to perform more rounds for the same number of bits communicated, leading to a faster initial drop in training loss. The intuition behind our adaptive algorithm is to start with a small s and then gradually increase s as training progresses to reach a lower error floor. To formalize this, we provide below a convergence bound on the training loss versus the number of bits communicated for any given s.

Convergence Bound in terms of Error versus Number of Bits Communicated. For a non-convex objective function $f(\mathbf{w})$, it is common to look at the expected squared norm of the gradient of the objective function as the error metric we want to bound [19]. We analyze this quantity under the following standard assumptions.

Assumption 1. *The stochastic quantization operator $Q(.)$ is unbiased and its variance is at most some positive constant q times the squared ℓ_2 norm of its argument, i.e. $\forall\, \mathbf{w} \in \mathbb{R}^d$, $\mathbb{E}[Q(\mathbf{w})|\mathbf{w}] = \mathbf{w}$ and $\mathbb{E}[\|Q(\mathbf{w}) - \mathbf{w}\|_2^2 | \mathbf{w}] \leq q\|\mathbf{w}\|_2^2$.*

Assumption 2. *The local objective functions f_i are $L-$smooth, i.e.* $\forall\, \mathbf{w}, \mathbf{w}' \in \mathbb{R}^d$, $\|\nabla f_i(\mathbf{w}) - \nabla f_i(\mathbf{w}')\|_2 \leq L\|\mathbf{w} - \mathbf{w}'\|_2$.

Assumption 3. *The stochastic gradients computed at the clients are unbiased and their variance is bounded, that is, for all* $\mathbf{w} \in \mathbb{R}^d$, $\mathbb{E}[g_i(\mathbf{w}, \xi^{(i)})] = \nabla f_i(\mathbf{w})$ *and* $\mathbb{E}[\|g_i(\mathbf{w}, \xi^{(i)}) - \nabla f_i(\mathbf{w})\|_2^2] \leq \sigma^2$.

Assumption 4. *Each client i has a dataset $\mathcal{D}_i$ of m samples drawn independently from the same distribution (i.i.d data).*

Under these assumptions, the authors in [17] recently derived a convergence bound for the FL setup described in Section 2 for non-convex $\ell(\cdot;\cdot)$. We use this result for AdaQuantFL, however in practice our algorithm can also be successfully applied without Assumption 4 (non-i.i.d data) as seen in our experiments Section 5. Also while the existing result [17] studies the error convergence with respect to the number of training rounds, we bound the same error in terms of number of bits communicated, defined as follows.

Definition 1 (Number of Bits Communicated, B)**.** *The total number of bits that have been communicated by a client to the central server until a given time instant is denoted by B.*

Since all clients participate in a training round and follow the same quantization protocol, B is same for all clients at any instant. We also note that the stochastic uniform quantizer having s quantization levels, satisfies Assumption 1 with $q = \frac{d}{s^2}$ [7, 18]. Now using this definition of B and our earlier definition of C_s in (6) we get the following theorem:

Theorem 1. *Under Assumptions 1-4, take $Q(.)$ to be the stochastic uniform quantizer with s quantization levels. If the learning rate satisfies $1 - \eta L(1 + \frac{d\tau}{s^2 n}) - 2\eta^2 L^2 \tau(\tau - 1) \geq 0$, then we have the following error upper bound in terms of B:*

$$\frac{C_s}{B\tau} \sum_{k=0}^{(B/C_s)-1} \sum_{t=0}^{\tau-1} \mathbb{E}[\|f(\bar{\mathbf{w}}_{k,t})\|_2^2] \leq A_1 \log_2(4s) + \frac{A_2}{s^2} + A_3. \tag{8}$$

Here, $\bar{\mathbf{w}}_{k,t} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{w}_{k,t}^{(i)}$ denotes the averaged model across all clients at each step, and

$$A_1 = \frac{2(f(\mathbf{w}_0) - f^*)d}{\eta B \tau}, \qquad A_2 = \frac{\eta L d \sigma^2}{n},$$
$$A_3 = \frac{\eta^2 \sigma^2 (\tau - 1) L^2 (n+1)}{n} + \frac{\eta L \sigma^2}{n} + A_1 \frac{d + 32}{d}, \tag{9}$$

and $\mathbf{w}_0$ is a random point of initialization and f^* is the minimum value of our objective.

The proof of Theorem 1 can be found in Appendix A of the full version of our paper [20]. This error bound allows us to see the trade-off between coarse and aggressive quantization seen in Section 3, for different values of s. As we decrease s, the value of the first term in our error bound ($A_1 \log_2(4s)$) decreases but it also adds to the variance of our quantized updates which increases the second term (A_2/s^2).

4. PROPOSED ADAQUANTFL STRATEGY

Our proposed algorithm aims at adaptively changing the number of quantization levels s in the stochastic uniform quantizer such that the error upper bound in Theorem 1 is minimized at every value B. To do so, we discretize the entire training process into uniform communication intervals, where in each interval we communicate B_0 bits (see Fig. 1). We now discuss how to find the optimal s for each such interval.

Finding optimal s for each communication interval. We propose selecting an s at any B (assuming $\mathbf{w}_0$ as the point of initialization) by setting the derivative of our error upper bound in (8) to zero. Doing so, we get a closed form solution of an optimal s as:

$$s^* = \sqrt{\frac{\eta^2 L \sigma^2 \tau B \log_e(2)}{n(f(\mathbf{w_0}) - f^*)}}. \tag{10}$$

Now at the beginning of the k-th communication interval clients can be viewed as restarting training at a new initialization point $\mathbf{w}_0 = \mathbf{w}_k$. Using (10) we see that the optimal s for communicating the next B_0 bits is given by,

$$s_k^* = \sqrt{\frac{\eta^2 L \sigma^2 \tau B_0 \log_e(2)}{n(f(\mathbf{w}_k) - f^*)}} \tag{11}$$

As $f(\mathbf{w}_k)$ becomes smaller the value of s_k^* increases which supports our intuition that we should increase s as training progresses. However, in practice, parameters such as L, σ^2 and f^* are unknown. Hence, in order to obtain a practically usable schedule for s_k^*, we assume $f^* = 0$ and divide s_k^* by s_0^* to get the approximate adaptive rule:

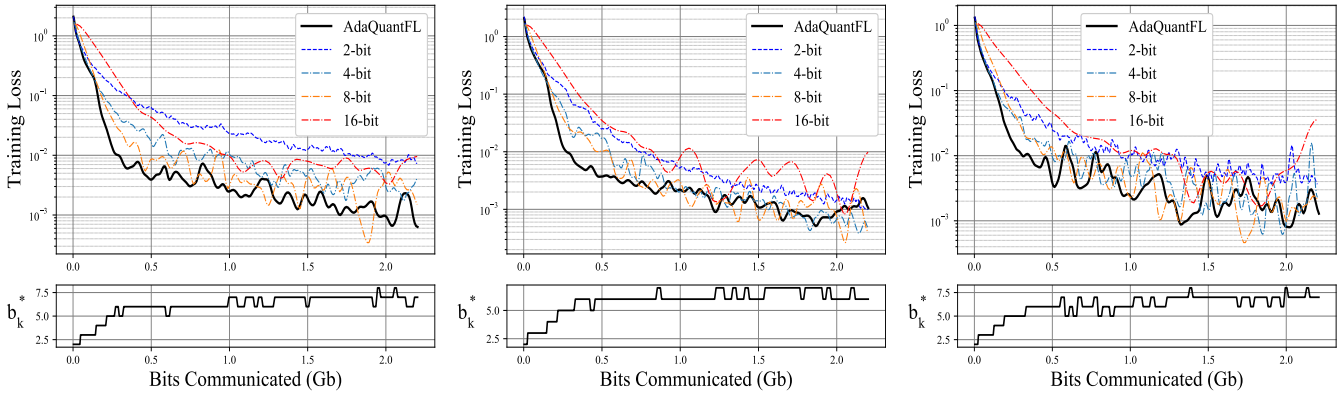
$$s_k^* \approx \sqrt{\frac{f(\mathbf{w}_0)}{f(\mathbf{w}_k)}} s_0^*. \tag{12}$$

The value of s_0^* can be found via grid search (we found $s_0^* = 2$ to be a good choice in our experiments).

Variable Learning Rate. Our analysis so far assumed the existence of a fixed learning rate η. In practice, we may want to decrease the learning rate as training progresses for better convergence. By extending the above analysis, we get an adaptive schedule of s for a given learning rate schedule:

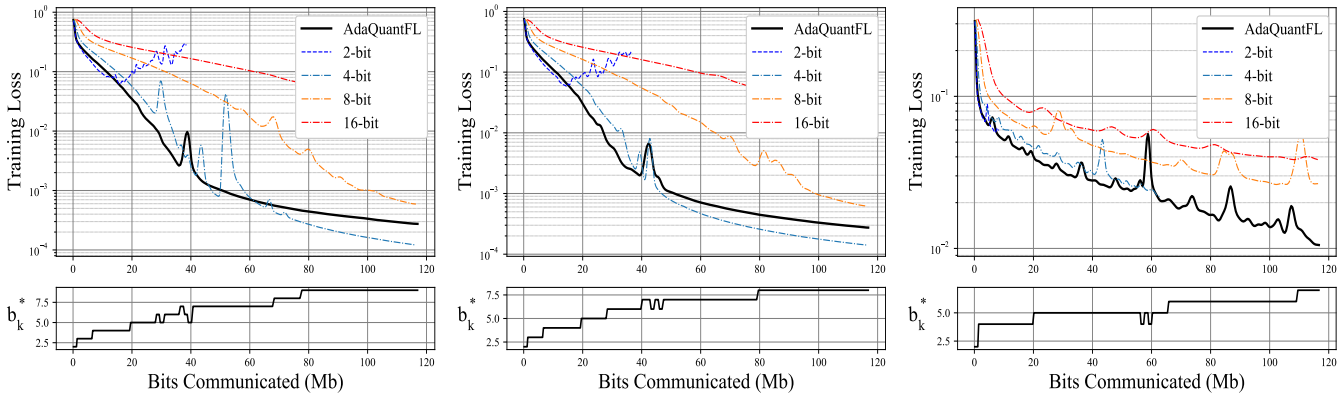
$$\boxed{\text{AdaQuantFL: } s_k^* \approx \sqrt{\frac{\eta_k^2 f(\mathbf{w}_0)}{\eta_0^2 f(\mathbf{w}_k)}} s_0^*.} \tag{13}$$

Here, η_0 is the initial learning rate and η_k is the learning rate in the k-th interval. In terms of the number of bits used to represent each element in the model update, in the k-th interval, AdaQuantFL uses $b_k^* = \lceil \log_2(s_k^* + 1) \rceil$ bits (excluding the sign bit).



(a) ResNet-18 with fixed LR, i.i.d data (b) ResNet-18 variable LR, i.i.d data (c) ResNet-18 with fixed LR, non-i.i.d data

Fig. 2: AdaQuantFL on ResNet-18 requires a fewer bits to reach a lower loss threshold, in (a) AdaQuantFL reaches a loss of 0.02 in 0.3Gb while the 2-bit method takes 1.8Gb. Here $b_k^* = \lceil \log_2(s_k^* + 1) \rceil$ (defined in Section 4).



(a) CNN with fixed LR, i.i.d data (b) CNN with variable LR, i.i.d data (c) CNN with fixed LR, non-i.i.d data

Fig. 3: For the Vanilla CNN, AdaQuantFL is able to achieve the lowest error floor of 0.02 for the non-i.i.d data distribution, while other methods converge at a higher error floor. Here $b_k^* = \lceil \log_2(s_k^* + 1) \rceil$ (defined in Section 4).

5. EXPERIMENTAL RESULTS

We evaluate the performance of AdaQuantFL against fixed quantization schemes using $b = \{2, 4, 8, 16\}$ bits respectively to represent each element of the model update (excluding the sign bit) using the stochastic uniform quantizer. The performance is measured on classification of the CIFAR-10 [21] and Fashion MNIST [22] datasets using ResNet-18 [23] and a Vanilla CNN architecture [1] (referred as CNN here on) respectively. For all our experiments we set the number of local updates to be $\tau = 10$, $\eta = 0.1$ and train our algorithm over 4 clients for the ResNet-18 and 8 clients for the CNN. For the variable learning rate setting, we reduce the learning rate by a factor of 0.9 every 100 training rounds. We run our experiments on both i.i.d and non-i.i.d distributions of data over clients. Our experimental results verify that AdaQuantFL is able to reach an error floor using much fewer bits in most cases as seen in Fig. 2 and Fig. 3. Additional details and figures, including test accuracy plots can be found in Appendix D of [20].

6. CONCLUSION

In this paper we present AdaQuantFL, a strategy to adapt the number of quantization levels used to represent compressed model updates in federated learning. AdaQuantFL is based on a rigorous error vs bits convergence analysis. Our experiments show that AdaQuantFL requires fewer bits to converge during training. A natural extension of AdaQuantFL would be using other quantizers such as the stochastic rotated quantizer [18] and the universal vector quantizer [9].

7. REFERENCES

[1] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agøura y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," *International Conference on Artificial Intelligenece and Statistics (AISTATS)*, Apr. 2017.

[2] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al., "Advances and open problems in federated learning," *arXiv preprint arXiv:1912.04977*, 2019.

[3] Jianyu Wang and Gauri Joshi, "Cooperative SGD: Unifying Temporal and Spatial Strategies for Communication-Efficient Distributed SGD," *arXiv preprint arXiv:1808.07576*, 2018.

[4] Jianyu Wang, Hao Liang, and Gauri Joshi, "Overlap local-SGD: An algorithmic approach to hide communication delays in distributed SGD," *arXiv preprint arXiv:2002.09539*, 2020.

[5] Farzin Haddadpour, Mohammad Mahdi Kamani, Mehrdad Mahdavi, and Viveck Cadambe, "Local sgd with periodic averaging: Tighter analysis and adaptive synchronization," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, Eds., pp. 11082–11094. Curran Associates, Inc., 2019.

[6] Sebastian U Stich, "Local sgd converges fast and communicates little," *arXiv preprint arXiv:1805.09767*, 2018.

[7] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic, "Qsgd: Communication-efficient sgd via gradient quantization and encoding," in *Advances in Neural Information Processing Systems*, 2017, pp. 1709–1720.

[8] Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," *arXiv preprint arXiv:1610.02527*, 2016.

[9] Nir Shlezinger, Mingzhe Chen, Yonina C Eldar, H Vincent Poor, and Shuguang Cui, "Uveqfed: Universal vector quantization for federated learning," *arXiv preprint arXiv:2006.03262*, 2020.

[10] Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li, "Terngrad: Ternary gradients to reduce communication in distributed deep learning," *arXiv preprint arXiv:1705.07878*, May 2017.

[11] Venkata Gandikota, Daniel Kane, Raj Kumar Maity, and Arya Mazumdar, "vqsgd: Vector quantized stochastic gradient descent," 2019.

[12] Hongyi Wang, Scott Sievert, Shengchao Liu, Zachary Charles, Dimitris Papailiopoulos, and Stephen Wright, "Atomo: Communication-efficient learning via atomic sparsification," in *Advances in Neural Information Processing Systems*, 2018, pp. 9850–9861.

[13] Thijs Vogels, Sai Praneeth Karimireddy, and Martin Jaggi, "Powersgd: Practical low-rank gradient compression for distributed optimization," in *Advances in Neural Information Processing Systems*, 2019, pp. 14259–14268.

[14] Fartash Faghri, Iman Tabrizian, Ilia Markov, Dan Alistarh, Daniel Roy, and Ali Ramezani-Kebrya, "Adaptive gradient quantization for data-parallel sgd," *arXiv preprint arXiv:2010.12460*, 2020.

[15] Jun Sun, Tianyi Chen, Georgios B Giannakis, and Zaiyue Yang, "Communication-efficient distributed learning via lazily aggregated quantized gradients," *arXiv preprint arXiv:1909.07588*, 2019.

[16] Jianyu Wang and Gauri Joshi, "Adaptive Communication Strategies for Best Error-Runtime Trade-offs in Communication-Efficient Distributed SGD," in *Proceedings of the SysML Conference*, Apr. 2019.

[17] Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani, "Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization," in *International Conference on Artificial Intelligence and Statistics*, 2020, pp. 2021–2031.

[18] Ananda Theertha Suresh, X Yu Felix, Sanjiv Kumar, and H Brendan McMahan, "Distributed mean estimation with limited communication," in *International Conference on Machine Learning*, 2017, pp. 3329–3337.

[19] Léon Bottou, Frank E Curtis, and Jorge Nocedal, "Optimization methods for large-scale machine learning," *arXiv preprint arXiv:1606.04838*, Feb. 2018.

[20] Divyansh Jhunjhunwala, Advait Gadhikar, Gauri Joshi, and Yonina C. Eldar, "Adaptive quantization of model updates for communication-efficient federated learning," *arXiv preprint arXiv:2102.04487*, 2021.

[21] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton, "Cifar-10 (canadian institute for advanced research)," .

[22] Han Xiao, Kashif Rasul, and Roland Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.