XFL-Q: Communication-Efficient Federated Learning with Layer-wise Quantization for IoT

Rachid El Mokadem, Yann Ben Maissa
National Institute of Posts and Telecommunications (INPT), STRS Laboratory, Rabat, Morocco
r.elmokadem@gmail.com, benmaissa@inpt.ac.ma

***Abstract*—Federated Machine Learning (FL) enables collaborative model training across distributed Internet of Things (IoT) devices without exposing raw data, preserving privacy while leveraging collective intelligence. However, communication costs remain a critical bottleneck in FL deployments due to constrained bandwidth and energy limitations in edge computing environments. In our previous work, we introduced XFL (eXtreme Federated Learning), an FL algorithm that transmits a *single* model layer per node, per communication round, achieving linear reductions in data exchange volumes.**

In this paper, we propose XFL-Q, an extension of XFL that incorporates aggressive quantization techniques using int4 and int8 precisions to compress transmitted layers during federated training. By combining single layer updates with quantization, our approach achieves a *synergistic* effect where reduced transmission granularity enables more aggressive compression without severely impacting model convergence. The experimental evaluation on a 6-layer CNN trained on the CIFAR-10 dataset under IID data distribution shows that XFL-Q reduces communication volume by up to 97.9% compared to standard FedAvg while maintaining comparable accuracy. This very significant improvement in communication efficiency makes XFL-Q particularly suitable for energy-constrained IoT and edge computing applications.

***Index Terms*—Federated Learning, Communication Efficiency, Quantization, Model Compression, IoT, Edge Computing**

I. Introduction

Context. The proliferation of IoT devices enables distributed machine learning across domains like smart cities, industrial automation, and mobile health. However, IoT resource constraints (bandwidth, energy, computation) challenge traditional centralized machine learning paradigms that rely on centralized data collection and processing.

Federated Machine Learning (FL) [1] addresses these challenges by enabling collaborative model training across distributed devices without requiring raw data transmission. In FL, client devices train locally on private data, sharing only model updates with a central server, preserving privacy and reducing communication overhead.

Motivation. Despite these advantages, communication remains the primary bottleneck in FL deployments, particularly in IoT scenarios where devices operate under severe bandwidth constraints and energy limitations. Standard FL algorithms like FedAvg [2] require transmission of complete model parameters at each round, resulting in substantial data exchange volumes that can be prohibitive for resource-constrained devices. Communication cost scales linearly with model size as each device transmits d 32-bit floating-point parameters per round. For modern deep networks where d can exceed millions of parameters, this creates substantial bandwidth requirements that may be unsustainable for IoT devices. Wireless communication energy often dominates computational costs in IoT: transmitting 1 byte can equal thousands of arithmetic operations [3]. This energy-communication trade-off makes communication reduction critical.

Our previous contribution addressed this challenge through XFL (eXtreme Federated Learning) [4], where we build upon the idea of Block Coordinate gradient Descent (BCD) [5] to optimize single layer's variables, per node, per round while others are fixed. This approach reduces communication overhead by transmitting only that layer instead of the entire model. It achieves linear reduction in communication volume proportional to the number of layers ($1/L$ for L layers), while maintaining convergence properties through a round-robin or stochastic layer selection mechanism. While XFL demonstrates significant improvements over standard FL approaches, the absolute size of transmitted data remains substantial, particularly for layers with large parameter counts.

Contribution. In this paper, we propose XFL-Q, which extends XFL by incorporating aggressive quantization techniques to further compress the transmitted layer parameters. The key insight is that the reduced granularity (one layer) of transmission in XFL creates opportunities for more aggressive quantization without severely impacting overall model convergence. By quantizing layer parameters to int4/int8 precision before transmission, we achieve significant communication volume reductions, leveraging redundancy across rounds to compensate for quantization errors. Our contributions in this work include: (1) the design and implementation of XFL-Q for federated learning in edge/IoT environments; (2) extensive experimental evaluation demonstrating up to 97.9% reduction in communication volume with minimal accuracy degradation; (3) analysis of the complementary effects between layer selection and quantization in federated settings.

Contents. The remainder of this paper is organized as follows. Section II reviews related work in communication-efficient federated learning and model compression techniques. In Section III, we propose our XFL-Q approach, including the layer selection mechanism and quantization process. Section IV describes our experimental setup and evaluation metrics. Section V presents results and discussion. Section VI concludes the paper.

979-8-3503-9281-4/25/$31.00 ©2025 IEEE

II. Related Work

Communication efficiency in federated learning has attracted significant research attention, with approaches broadly categorized into Gradient/Model Compression and Communication Protocol Optimization.

A. Gradient and Model Compression

Gradient sparsification techniques aim to reduce communication by transmitting only the most significant gradient components. SignSGD [6] reduces gradients to binary representations, achieving substantial compression at the cost of convergence speed. Top-k sparsification [7] methods select only the largest k gradient components for transmission, and adaptive schemes dynamically adjusting k based on communication budgets or convergence requirements in [8]. QSGD [9] introduces stochastic quantization for gradient compression, providing theoretical guarantees on convergence while achieving practical compression ratios. Quantization-based approaches have shown particular promise for federated learning scenarios. PowerSGD [10] applies low-rank approximation combined with quantization to compress gradient updates. Recent work has explored extreme quantization schemes, including ternary [11] and binary neural networks [12], though these typically require architectural modifications or specialized training procedures. Model compression techniques, including pruning and knowledge distillation, offer complementary approaches to communication reduction. Structured pruning methods remove entire channels or layers, reducing both computation and communication overhead.

However, these approaches typically require careful retraining and may not be suitable for heterogeneous federated environments where different devices may benefit from different compression strategies.

B. Communication Protocol Optimization

Standard federated learning algorithms like FedAvg [1] and FedPAQ [13] focus on reducing communication rounds through local computation, but still require transmission of complete model parameters. FedAvg performs multiple local SGD steps before communication, trading computation for communication efficiency. FedPAQ combines periodic local averaging with quantized updates to reduce both communication frequency and message size while preserving convergence.

Layer-wise update schemes represent a less explored but promising direction for extreme communication reduction in federated learning. They perform optimization at the layer level: layer-wise pruning in FedLP [14] and variable update frequency per layer in FedVF [15]. These techniques address the variation in sensitivity and importance across different layers, to accelerate model convergence and reduce data exchanged at each round. While these works offer important results, they often involve partial or selective updates of multiple layers per round, and may still rely on transmitting substantial amounts of data.

C. XFL and remaining Communication Challenges

Compared to most existing approaches, our previous algorithm [4] (XFL), achieves greater communication efficiency by significantly minimizing the size of exchanged updates. Sending a single layer per communication round per client can achieve very important data volume reductions while maintaining convergence properties.

XFL's approach is based on Block Coordinate Descent (BCD) optimization, where each iteration optimizes one block of variables while keeping others fixed. For us, each layer is a block: in every round, clients update and transmit one layer's parameters, which the server then aggregates. This leads to a linear reduction in communication volume by a factor of $1/L$, where L is the total number of layers, without compromising convergence guarantees. By design, it offers structural communication advantages over traditional weight-compression methods.

However, individual layers can still contain a substantial number of parameters, despite layer-wise transmission. For instance, neural network's layers in modern architectures may have hundreds of thousands or even millions of parameters. Additionally, the standard 32-bit floating-point format may exceed the precision needed in federated learning, especially when updates are aggregated across a large number of devices over noisy communication channels.

These considerations motivate enhancing the XFL framework with an additional layer-wise compression mechanism to push communication efficiency further. Combining the layer updates with value-based quantization enables excellent compression ratios. It makes this approach very good for resource-constrained IoT environments, where communication directly impacts battery life and deployment feasibility.

III. Proposed Approach: XFL-Q

In this section, we present XFL-Q, our approach which performs aggressive quantization on the XFL transmitted layer. In contrast to typical quantization methods, which often necessitate uniform compression throughout entire models, our proposed approach can apply more aggressive compression to individual layers as the round-robin transmission pattern can potentially distribute quantization errors across the training process. This represents a new *synergy* between structural sparsity (layer selection) and precision reduction (quantization) specifically optimized for resource-constrained scenarios.

A. From XFL to XFL-Q

XFL addresses communication bottlenecks by transmitting only a single layer per communication round per client. Given a neural network with L layers $W_1, W_2, ..., W_L$, each device selects one layer W_i for transmission at each round, achieving immediate communication reduction by a factor of L (Equation 1).

Layer selection can be deterministic (round-robin) or stochastic. In this study's round-robin approach, layers are

selected cyclically by each client (e.g., layer $j+1$ in round 1, $j+2$ in round 2), ensuring all layers are covered.

$$Wg^l \leftarrow Wg^l - L_r \frac{1}{n_l} \sum \Delta W_{l(i,r)} \tag{1}$$

Where Wg^l is the parameter of layer l in the global model, $\Delta W_{l(i)}$ is the gradient calculated for layer l by the i^{th} node, $l(i,r)$ is the layer index chosen by the i^{th} node in the current round r, L_r is the learning rate of the gradient descent algorithm, n_l is the number of updates received for layer l.

Theoretical analysis of XFL [4] shows convergence properties close to standard federated learning under appropriate conditions. Our idea was that XFL can be viewed as block coordinate descent in the federated setting, where coordinate blocks correspond, in our case, to network layers. Despite transmitting only partial model updates, the global model converges to the good optimum close to full-parameter transmission approaches.

Our new algorithm XFL-Q (Algorithm 1) operates in two phases: layer selection and quantization. During layer selection, each participating device identifies the layer to be transmitted according to the predefined selection strategy (round-robin or stochastic). During quantization, the selected layer parameters are compressed using uniform quantization to the target precision (int4 or int8) before transmission to the central server. The algorithm ensures all layers are eventually updated while greatly reducing per-round communication overhead.

Algorithm 1 begins by initializing the participating nodes and sending a full model with L layers to each client (lines 1–2), followed by setting the number of training rounds (line 3). In each round (line 4), clients perform local training (line 5) and select one layer to communicate using a cyclic strategy (line 6). The selected layer is quantized using computed scaling and zero-point parameters (lines 8–9) and transmitted to the server (line 10). The server then dequantizes the received layers (line 13), aggregates them (line 14), and sends the updated model back to the clients (line 15).

B. Layer Selection Mechanism

The layer selection mechanism in XFL-Q follows the established XFL approach. In the basic round-robin strategy (lines 7-8 in algorithm 1), layer i is selected at round r according to:

$$i = (r-1) \bmod L + 1 \tag{2}$$

where L is the total number of layers. This ensures that all layers are updated with the same frequency during the training process, and our experimental evaluation focuses on uniform selection to maintain simplicity and ensure a fair comparison with baseline approaches.

C. Quantization Process

The quantization process converts floating-point layer parameters to low-precision integer representations for transmission. We implement uniform quantization schemes for both int8 and int4 target precisions (lines 9-10 in algorithm 1).

Algorithm 1 XFL-Q Algorithm

1: Server sets number of participating nodes N
2: Server sends the initial model to client nodes with L layers
3: Server sets number of rounds R ($R \gg L$)
4: **for** each round **do**
5: **for** each node **do**
6: Run local training on node's data
7: l = GetCyclicLayer(round, node)
8: Choose the l-th layer to exchange with the server
9: Compute quantization parameters: scaling factor s_l and zero-point z_l for layer l
10: Quantize the layer l: $q_l = \text{round}((W_l - z_l)/s_l)$
11: Transmit quantized layer q_l along with (s_l, z_l)
12: **end for**
13: Server receives quantized layers and quantization parameters from participating nodes
14: **for** each received layer l **do**
15: Dequantize layer: $\hat{W}_l = s_l \cdot q_l + z_l$
16: **end for**
17: Server updates its model's layers with the averages of dequantized layer parameters
18: Server sends back the aggregated model to the nodes
19: **end for**

For int8 quantization, parameters are mapped to the range $[-128, 127]$ using:

$$q = \text{round}\left(\frac{x - z}{s}\right) \tag{3}$$

where x is the original parameter value, s is the scaling factor, and z is the zero-point offset. The scaling factor is computed as:

$$s = \frac{\max(x) - \min(x)}{255} \tag{4}$$

with zero-point:

$$z = \min(x) + 128 \cdot s \tag{5}$$

For int4 quantization, the range is reduced to $[-8, 7]$ with corresponding adjustments to the scaling computation:

$$s = \frac{\max(x) - \min(x)}{15} \tag{6}$$

$$z = \min(x) + 8 \cdot s \tag{7}$$

The quantized parameters are transmitted as integer values along with the scaling factor s and zero-point z (line 11 in algorithm 1), which require minimal additional overhead (typically 8 bytes total per layer regardless of layer size).

D. Dequantization and Aggregation

Upon receiving quantized layer parameters from participating devices, the central server performs dequantization to reconstruct approximate floating-point values (line 15 in algorithm 1):

$$\hat{x} = s \cdot q + z \tag{8}$$

In line 17 of algorithm 1 the dequantized parameters are then aggregated using standard federated averaging:

$$W_{\text{global}} = \frac{1}{N} \sum_{i=1}^{N} W_i \tag{9}$$

where N is the number of participating devices. The aggregated layer parameters are incorporated into the global model, replacing the corresponding layer from the previous round.

E. Compression Theoretical Analysis

The compression ratio achieved by XFL-Q combines the layer selection technique from XFL with quantization compression. For a model with L layers, XFL achieves a base compression ratio of $1/L$. Additional quantization compression depends on the target precision:

- Int8 quantization: $32/8 = 4\times$ compression
- Int4 quantization: $32/4 = 8\times$ compression

The combined compression ratio for XFL-Q is:

$$\text{Compression Ratio} = \frac{1}{L} \times \frac{32}{\text{bits per parameter}} \tag{10}$$

For our 6-layer CNN with int8 quantization: $\frac{1}{6} \times 4 \approx 24\times$ compression (95.8% reduction). For int4 quantization: $\frac{1}{6} \times 8 \approx 48\times$ compression (97.9% reduction).

IV. Experimental Setup

In this section we present our experimental setup through the model architecture and dataset, the federated learning setup and the metrics used to evaluate the approach.

A. Model Architecture and Dataset

We evaluate XFL-Q using an 6-layer Convolutional Neural Network (CNN) trained on the CIFAR-10 dataset. The CNN architecture consists of four convolutional layers followed by two fully connected layers:

- Conv1: $3 \times 16 \times 3 \times 3$ (448 parameters)
- Conv2: $16 \times 32 \times 3 \times 3$ (4,640 parameters)
- Conv3: $32 \times 64 \times 3 \times 3$ (18,496 parameters)
- Conv4: $64 \times 128 \times 3 \times 3$ (73,856 parameters)
- FC1: 512×256 (131,328 parameters)
- FC2: 256×10 (2,570 parameters)

The total model parameters are: 231,338.

The CIFAR-10 dataset provides a standard benchmark with 50,000 training images and 10,000 test images across 10 classes. We use IID data distribution across participating devices to isolate the effects of quantization from data heterogeneity challenges.

B. Federated Learning Setup

Our experimental setup simulates 18 participating devices in a federated learning environment. Each device receives an equal portion of the training data (5,000 images per device). The code used for validation is publicly accessible on our Gitlab repository[1] Training parameters include:

- Learning rate: 0.01
- Batch size: 20
- Local epochs: 40 (E in FedAvg terminology)
- Communication rounds: 10
- Optimizer: SGD

[1]https://gitlab.com/rachid-el-mokadem/xfl-q

TABLE I
Communication Volume Comparison

Method	Bytes per Round	Reduction vs FedAvg
FedAvg	925,352	0%
XFL	154,225	83.3%
XFL-Q (int8)	38,556	95.8%
XFL-Q (int4)	19,278	97.9%

- Framwork: PyTorch

We compare four configurations :

1) **FedAvg (Baseline)**- Standard federated averaging with full model transmission
2) **XFL**- Layer-wise transmission without quantization (32-bit float)
3) **XFL-Q (int8)**- Layer-wise transmission with int8 quantization
4) **XFL-Q (int4)**- Layer-wise transmission with int4 quantization

C. Evaluation Metrics

To extensively evaluate the performance of XFL-Q, we consider multiple metrics that jointly capture both communication efficiency and model accuracy:

1) **Communication Volume** – Total bytes transmitted per communication round in uplink (device to server) traffic.
2) **Model Accuracy** – Classification accuracy on the CIFAR-10 test set, measured after each communication round.
3) **Accuracy per Byte** – A composite metric calculated as the test accuracy divided by cumulative bytes transmitted, providing insight into communication efficiency relative to model performance.
4) **Convergence Speed** – Number of communication rounds required to reach target accuracy thresholds (e.g., 50% test accuracy).
5) **Energy Estimation** – Approximate energy consumption based on transmitted data volume using typical IoT device energy consumption rates (100 mJ per transmitted byte for low-power wireless communication).

V. Experimental Validation and Discussion

This section presents the results of our experiments and provides a discussion on the performance outcomes and their significance to the proposed approach.

A. Communication Reduction Analysis

Table I presents the communication volume analysis for different approaches, showing the dramatic reductions achieved by XFL-Q. The results demonstrate that XFL-Q achieves remarkable communication reductions while building upon the foundation established by XFL. The int8 quantization provides a 95.8% reduction compared to standard FedAvg, while int4 quantization reaches 97.9% reduction. These improvements represent practical significance for IoT deployments where communication costs directly impact battery life and network

TABLE II
ENERGY CONSUMPTION APPROXIMATION

Method	Energy per Round (J)	Total Energy (J)
FedAvg	1.46	14.63
XFL	0.24	2.44
XFL-Q (int8)	0.06	0.61
XFL-Q (int4)	0.03	0.31

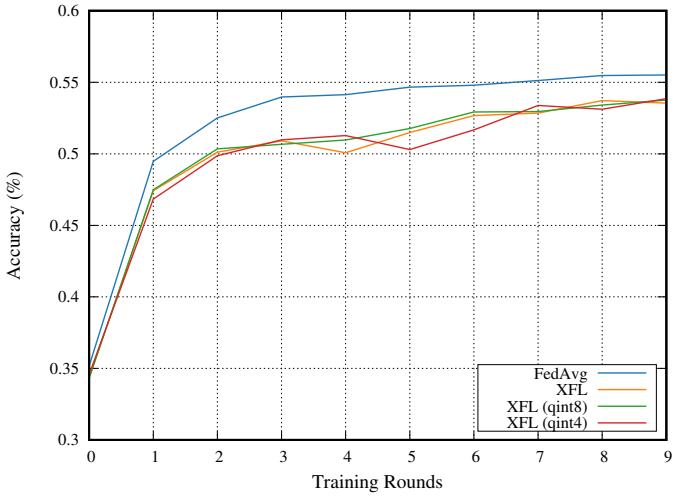

Fig. 1. Test accuracy convergence comparison across different methods over 10 communication rounds.

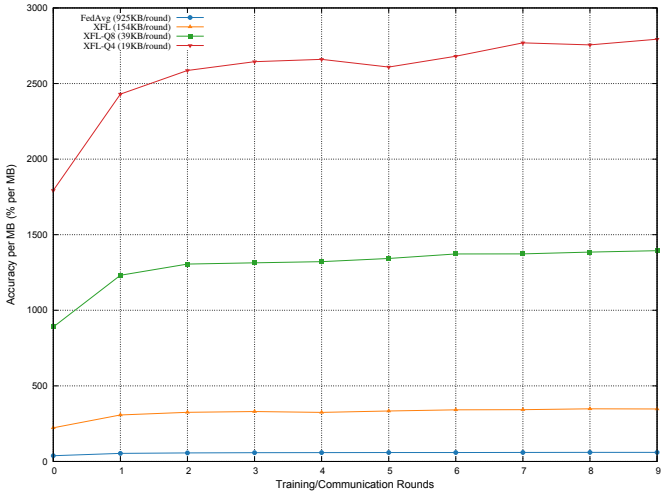

Fig. 2. Accuracy per byte transmitted comparison showing communication efficiency of different methods.

TABLE III
FINAL ACCURACY PER BYTE COMPARISON

Method	Final Accuracy per Byte ($\times 10^{-6}$)
FedAvg	0.597
XFL	3.554
XFL-Q (int8)	14.025
XFL-Q (int4)	27.805

capacity. To have an idea of the impact on energy efficiency, we attempt to use typical energy consumption approximations for IoT wireless communication. For example, in the case of LoRaWAN [16], we have:

- Lower bound (larger payloads)- $\approx 0.84mJ/byte$
- Upper bound (small payloads)- $\approx 2.32mJ/byte$

We approximate by taking the average value $1.58mJ/Byte$ to estimate the energy requirements for different approaches.

B. Accuracy Performance

Figure 1 shows the convergence behavior of different approaches over 10 communication rounds. The results reveal several important findings.

Regarding convergence speed, all methods demonstrate similar convergence patterns, reaching approximately 50% accuracy within the first few communication rounds. XFL-Q variants show slightly slower initial convergence but achieve comparable final accuracy.

With regards to accuracy stability, XFL-Q with int8 quantization maintains accuracy very close to the baseline methods, with final test accuracy within $1-2\%$ of FedAvg, and very close to XFL (0.7%). The int4 quantization shows slightly more accuracy degradation but remains within acceptable bounds ($2-3\%$) for many practical applications.

As for quantization impact, the difference between int8 and int4 quantization is relatively modest, suggesting that aggressive quantization (int4) may be viable for applications where communication constraints are particularly severe.

Final accuracy results after 10 rounds are as follows.

- FedAvg: 55.2%
- XFL: 54.8%
- XFL-Q (int8): 54.1%
- XFL-Q (int4): 53.6%

The accuracy per byte metric is very important to us, as it provides insight into the communication efficiency of different approaches. Figure 2 illustrates this composite metric over the training process. XFL-Q demonstrates dramatic improvements in accuracy per byte, with int4 quantization achieving nearly $47\times$ better efficiency than standard FedAvg. This metric particularly highlights the practical value of XFL-Q for resource-constrained scenarios where communication budget directly limits learning performance.

C. Layer-wise Quantization Effects

Analysis of quantization effects across different layer types reveals the following interesting patterns.

Convolutional Layers. These layers show high tolerance to aggressive quantization, likely due to their spatial structure and redundancy. Int4 quantization of convolutional layers shows minimal accuracy impact.

Fully Connected Layers. These layers, particularly the larger FC1 layer, show more sensitivity to quantization. However, the round-robin transmission pattern helps distribute quantization errors, maintaining overall model performance.

Output Layer. The final classification layer (FC2) demonstrates high sensitivity to quantization, but its small size (2,570 parameters) means it has minimal impact on overall communication volume.

The round-robin layer transmission pattern in XFL-Q provides inherent robustness to quantization errors. Since each layer is updated multiple times throughout the training process, quantization errors in individual rounds can be corrected in subsequent rounds. This error distribution mechanism is a key factor enabling aggressive quantization without severe accuracy degradation.

D. IoT and Edge Computing Applications

Experimental results show that XFL-Q offers significant improvements in communication efficiency, which is an important challenge in IoT and edge computing. By cutting communication volume by more than 97% compared to FedAvg, our approach could overcome strict bandwidth and duty-cycle limits typical of LPWAN protocols like LoRaWAN and NB-IoT.

Even with aggressive quantization at the layer level (int4), XFL-Q maintains accuracy close to baseline methods. The round-robin update helps spread and correct quantization errors across training rounds, and convolutional layers show resilience to low-bit compression, which could inspire model design adapted to wireless constraints.

Moreover, XFL-Q achieves up to 47× better accuracy per byte transmitted compared to FedAvg: a metric that captures both learning performance and communication cost. This highlights its potential for federated learning systems optimized for the limitations of wireless and mobile networks.

Overall, these findings suggest XFL-Q could be a promising step towards practical federated learning on real-world, bandwidth and energy limited IoT/edge platforms.

VI. Conclusion

Summary. This paper introduced XFL-Q, a communication-efficient federated learning approach that combines layer-wise transmission with aggressive quantization for IoT and edge computing applications. Block coordinate gradient descent was used to optimize one layer while keeping others fixed. int4/int8 quantization was employed to compress its parameters before transmission to the server. Through experimental evaluation on CIFAR-10 using a 6-layer CNN, we showed that XFL-Q achieves a 95.8% reduction in communication volume using the int8 encoding and 97.9% reduction using int4, with an accuracy close to the standard FL approach. We further analyzed the trade-offs between communication efficiency and model accuracy in resource-constrained environments.

We believe that the results demonstrate that XFL-Q addresses some critical challenges in federated learning deployment for IoT/edge applications and protocols, where sometimes battery life requirements are as important as Quality of Service.

Future works. This work opens at least 2 promising avenues for investigation: Adaptive Quantization and the Impact of Non-IID Environments. Indeed, dynamic quantization schemes that adjust precision based on layer significance or training status could further optimize the communication-accuracy trade-off in our approach. Regarding non-IID Environments, understanding how data heterogeneity interacts with layer-wise quantization effects is crucial to improve our algorithm, as real-world non-IID conditions may amplify quantization-induced performance impacts.

References

[1] H. 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 the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*. PMLR, 2017, pp. 1273–1282.

[2] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[3] F. Marcelloni and M. Vecchio, "A simple algorithm for data compression in wireless sensor networks," *IEEE Communications Letters*, vol. 12, no. 6, pp. 411–413, 2008.

[4] R. El Mokadem, Y. Ben Maissa, and Z. El Akkaoui, "extreme federated learning (XFL): A layer-wise approach," *Cluster Computing*, vol. 27, no. 5, pp. 5741–5754, 2024.

[5] A. Beck and L. Tetruashvili, "On the convergence of block coordinate descent type methods," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2037–2060, 2013.

[6] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, "signsgd: Compressed optimisation for non-convex problems," in *International Conference on Machine Learning (ICML)*. PMLR, 2018, pp. 560–569.

[7] R. El Mokadem, Y. B. Maissa, and Z. El Akkaoui, "Federated learning communications optimization using sparse single-layer updates," *Procedia Computer Science*, vol. 236, pp. 168–176, 2024.

[8] S. Lu, R. Li, W. Liu, C. Guan, and X. Yang, "Top-k sparsification with secure aggregation for privacy-preserving federated learning," *Computers & Security*, vol. 124, p. 102993, 2023.

[9] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "Qsgd: Communication-efficient sgd via gradient quantization and encoding," in *Advances in Neural Information Processing Systems (NeurIPS 30)*, 2017.

[10] T. Vogels, S. P. Karimireddy, and M. Jaggi, "Powersgd: Practical low-rank gradient compression for distributed optimization," in *Advances in Neural Information Processing Systems (NeurIPS 32)*, 2019, pp. 14 618–14 628.

[11] C. Zhu, S. Han, H. Mao, and W. J. Dally, "Trained ternary quantization," *arXiv preprint arXiv:1612.01064*, 2016.

[12] H. Qin, R. Gong, X. Liu, X. Bai, J. Song, and N. Sebe, "Binary neural networks: A survey," *Pattern Recognition*, vol. 105, p. 107281, 2020.

[13] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie, and R. Pedarsani, "Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*. PMLR, 2020, pp. 2021–2031.

[14] Z. Zhu, Y. Shi, J. Luo, F. Wang, C. Peng, P. Fan, and K. B. Letaief, "Fedlp: Layer-wise pruning mechanism for communication-computation efficient federated learning," in *ICC 2023-IEEE International Conference on Communications*. IEEE, 2023, pp. 1250–1255.

[15] Y. Mei, B. Guo, D. Xiao, and W. Wu, "Fedvf: Personalized federated learning based on layer-wise parameter updates with variable frequency," in *2021 IEEE International Performance, Computing, and Communications Conference (IPCCC)*. IEEE, 2021, pp. 1–9.

[16] S. Mathur, S. Sahu, X. Li, and S. Ganu, "Energy analysis of lorawan technology for traffic sensing applications," in *Intelligent Transportation Systems (ITS) World Congress*, 2017.