Cost-Efficient Federated Learning over Wireless Networks: A Proactive Stop Policy

SPS Seasonal PhD School, Aalto University

Carlo Fischione

Division of Network and Systems Engineering

KTH Royal Institute of Technology, Stockholm, Sweden

Email: carlofi@kth.se

March 2022

Outline

- State of the Art
- System Model and Problem Formulation
- Solution for Convex Loss Functions
- Solution for Non-convex Loss Functions
- Numerical Results
- Conclusion

State of the Art

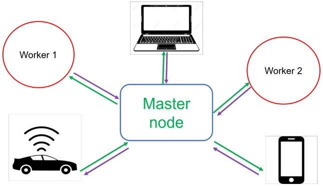


Figure 2

Limitation of existing communication-efficient ML over networks

× Not considering the cost of running each iteration of an iterative ML over networks even while applying the well-known communication-efficient methods as:
  - Top-q sparsification [1]
  - Lazily aggregated gradient quantization (LAQ) [2]

- ▶ Adapt to the underlying communication protocols
- ▶ What we address is called "ML over networks" [3]
  - Distributed optimization for ML over a wireless communication network
  - The computations and communications must be efficient

[1] Sattler et al. *Robust and communication-efficient federated learning from non-iid data*. IEEE Transactions on Neural Networks and Learning Systems 2019.

[2] Sun et al. *Lazily aggregated quantized gradient innovation for communication-efficient federated learning*. IEEE Transactions on Pattern Analysis and Machine Intelligence 2020.

[3] Mahmoudi et.al *Cost-efficient Distributed Optimization In Machine Learning Over Wireless Networks*. IEEE ICC 2020.

Overview of Federated Learning (FL)

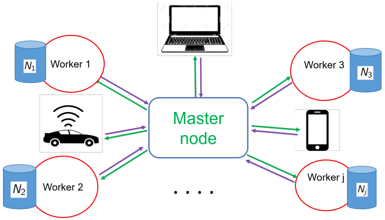


Figure 3

Federated learning network setup [4], [5]:

- Star network: one master node and M workers
- Every worker $j = 1, 2, \ldots, M$ has its own dataset with N_j samples
- Data sample in each worker j is $\boldsymbol{x}_{ij}, y_{ij}$
- Workers aim to collaboratively solve the optimization problem (1)

$$\boldsymbol{w}^\star \in \min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w}) = \sum_{j \in [M]} \frac{1}{\sum_{j \in [M]} N_j} \sum_{i \in [N_j]} f(\boldsymbol{w}; \boldsymbol{x}_{ij}, y_{ij}) \tag{1}$$

[4] Konečn et al. *Federated learning: Strategies for improving communication efficiency*. arXiv preprint arXiv:1610.05492, 2016.

[5] Chen et al. *A Joint Learning and Communications Framework for Federated Learning Over Wireless Networks*. IEEE Transactions on Wireless Communications, 2021.

FL: Iterative Solution

$$\boldsymbol{w}^\star \in \min_{\boldsymbol{w}\in\mathbb{R}^d} f(\boldsymbol{w}) = \sum_{j\in[M]} \frac{1}{\sum_{j\in[M]} N_j} \sum_{i\in[N_j]} f(\boldsymbol{w};\boldsymbol{x}_{ij}, y_{ij}), \quad (1)$$

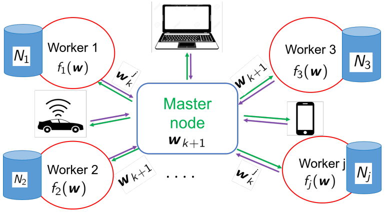


Figure 4

To solve (1), the iterative procedure at each iteration $k = 1, \ldots, K$ is

1. Every worker $j \in [M]$ updates its local parameter $\boldsymbol{w}^j_{k+1}$ as

$$\boldsymbol{w}^j_{k+1} = \boldsymbol{w}_k - \frac{\alpha_k}{N_j} \sum_{i\in[N_j]} \nabla_{\boldsymbol{w}} f(\boldsymbol{w}_k; \boldsymbol{x}_{ij}, y_{ij}) \quad (2)$$

2. All workers transmit their local parameter $\boldsymbol{w}^j_{k+1}$ to the master node
3. Master node computes the global parameter $\boldsymbol{w}_{k+1}$ by taking a weighted sum over all local parameters

$$\boldsymbol{w}_{k+1} = \sum_{j\in[M]} \frac{N_j}{\sum_{j\in[M]} N_j} \boldsymbol{w}^j_{k+1} \quad (3)$$

Federated Averaging (FedAvg)

Iterative procedure

$$\boldsymbol{w}^\star \in \min_{\boldsymbol{w}\in\mathbb{R}^d} f(\boldsymbol{w}) = \sum_{j\in[M]} \frac{1}{\sum_{j\in[M]} N_j} \sum_{i\in[N_j]} f(\boldsymbol{w}; \boldsymbol{x}_{ij}, y_{ij}), \quad (1)$$

To solve (1), the procedure ´ at each global iteration $k = 1, \ldots, K$ is

1. Every worker $j \in [M]$ updates its local parameter $\boldsymbol{w}_k^j$ after $l = 1, \ldots, k_l$ local iterations (define $\boldsymbol{w}_{k+1}^{j,0} := \boldsymbol{w}_k$, and $\boldsymbol{w}_{k+1}^j := \boldsymbol{w}_{k+1}^{j,k_l}$)

$$\boldsymbol{w}_{k+1}^{j,l} = \boldsymbol{w}_k^{j,l-1} - \frac{\alpha_k}{N_j} \sum_{i\in[N_j]} \nabla_{\boldsymbol{w}} f(\boldsymbol{w}_k^{j,l-1}; \boldsymbol{x}_{ij}, y_{ij}) \quad (4)$$

2. All workers transmit their local parameter $\boldsymbol{w}_{k+1}^j$ to the master node
3. Master node computes the global parameter $\boldsymbol{w}_{k+1}$ by taking a weighted sum over all local parameters

$$\boldsymbol{w}_{k+1} = \sum_{j\in[M]} \frac{N_j}{\sum_{j\in[M]} N_j} \boldsymbol{w}_{k+1}^j \quad (5)$$

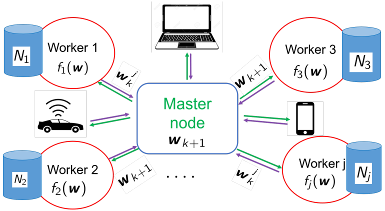


Figure 4

Outline

System Model

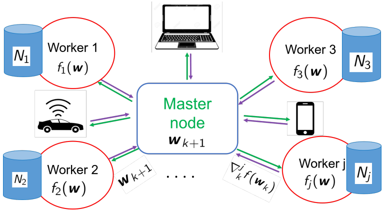


Figure 4

K: the stopping step of the FL iterations

$$K := \text{the first value of } k \mid \|f(\boldsymbol{w}_k) - f(\boldsymbol{w}^*)\| < \epsilon \tag{6}$$

$\epsilon > 0$: the threshold at which we decide to stop the distributed FL algorithm

c_k: the cost of iteration k

$\sum_{k=1}^{K} c_k :=$ total cost of running the iterations (iteration-cost function)

Problem Formulation

- Let us consider as the general cost a multi-objective function of the iteration-cost $\sum_{k=1}^{K} c_k$ and the loss function $f(\boldsymbol{w}_K)$
- Our problem consists in finding the stopping iteration that minimizes the cost of solving problem (1)

$$\underset{K}{\text{minimize}} \quad \left(\beta \sum_{k=1}^{K} c_k + (1-\beta) f(\boldsymbol{w}_K)\right) \tag{7a}$$

$$\text{s.t.} \quad \boldsymbol{w}_{k+1}^{j,l} = \boldsymbol{w}_k^{j,l-1} - \frac{\alpha_k}{N_j} \sum_{i \in [N_j]} \nabla_w f(\boldsymbol{w}_k^{j,l-1}; \boldsymbol{x}_{ij}, y_{ij}) \tag{7b}$$

$$\boldsymbol{w}_{k+1} = \sum_{j \in [M]} \frac{N_j}{\sum_{j \in [M]} N_j} \boldsymbol{w}_{k+1}^{j} \tag{7c}$$

- $\beta \in (0,1)$: Scalarization factor

To solve problem (7), we need the future information of $(f(\boldsymbol{w}_k), c_k)_{k=1,\dots,K}$

Non-Causal Problem!

Solution Approach

- Define $G(K) := \beta \sum_{k=1}^{K} c_k + (1-\beta) f(\boldsymbol{w}_K)$,

$$k^* \in \arg\min_{K} \; G(K) \tag{8a}$$

$$\text{s.t. } \boldsymbol{w}_{k+1}^{j,l} = \boldsymbol{w}_k^{j,l-1} - \frac{\alpha_k}{N_j} \sum_{i \in [N_j]} \nabla_w f(\boldsymbol{w}_k^{j,l-1}; \boldsymbol{x}_{ij}, y_{ij}) \quad \forall k \tag{8b}$$

$$\boldsymbol{w}_{k+1} = \sum_{j \in [M]} \frac{N_j}{\sum_{j \in [M]} N_j} \boldsymbol{w}_{k+1}^{j} \quad \forall k \tag{8c}$$

- $G(k^*)$: the optimum of (7)

The question is now how to solve optimization problem (8) in a causal way!

FL Causal Setting (FLCau) Algorithm to Solve Problem (8)

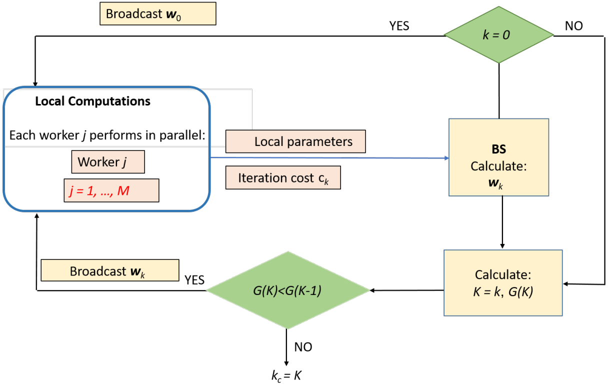


Figure 5

- The algorithm is applicable in real world: solves problem (8) without future information
- We prove that $G(K)$ is discrete-convex
- $G(K)$ allows to find the unique minimum (see next slide)

Outline

- State of the Art
- System Model and Problem Formulation
- Solution for Convex Loss Functions
- Solution for Non-convex Loss Functions
- Numerical Results
- Conclusion

Solution for Problem (8) for Convex Loss Functions

Causal Setting

- The FLCau Algorithm finds the optimal or near to the optimal solution

Proposition 1

Let $f(\boldsymbol{w}_k)$ be convex. Let $k^\star$ be the solution to problem (8). Then, the FLCau solution k_c is such that

- $k^\star \leq k_c \leq k^\star + 1$
- $f(\boldsymbol{w}_{k_c}) \leq f(\boldsymbol{w}_{k^\star})$
- $G(k_c) \geq G(k^\star)$

Outline

- State of the Art
- System Model and Problem Formulation
- Solution for Convex Loss Functions
- Solution for Non-convex Loss Functions
- Numerical Results
- Conclusion

Cost-Efficient FL for Non-Convex Loss Functions

Challenges

- Consider non-convex loss function $f(\boldsymbol{w})$ like Neural Networks (NNs) prediction loss
- Issue: the non-convexity leads to early stopping of FLCau
- Solution approach:
  - Define decreasing upper bound $F_u(\boldsymbol{w})$ and lower bound $F_l(\boldsymbol{w})$ functions for $f(\boldsymbol{w}_k)$
  - Form upper and lower bounds on $G(K)$
  - Find an interval of $k_c \in [k_c^u, k_c^l]$
  - The bounds do not change the training process and iteration-cost c_k

Cost-Efficient FL for Non-Convex Objective Functions

Upper and Lower Bounds

- Define multi-objective upper bound $G_u(K)$ and lower bound $G_l(K)$ functions

$$G_u(K) := \beta \sum_{k=1}^{K} c_k + (1-\beta) F_u(\boldsymbol{w}_K)\,, \tag{9a}$$

$$G_l(K) := \beta \sum_{k=1}^{K} c_k + (1-\beta) F_l(\boldsymbol{w}_K) \tag{9b}$$

- Calculating $k_u^\star$ and $k_l^\star$

$$k_u^\star \in \arg\min_{K \in \mathbb{N}} \; G_u(K)\,, \tag{10a}$$

$$k_l^\star \in \arg\min_{K \in \mathbb{N}} \; G_l(K) \tag{10b}$$

- According to Proposition 2 about causal stopping iteration

$$k_u^\star \le k_c^u \le k_u^\star + 1\,, \tag{11a}$$

$$k_l^\star \le k_c^l \le k_l^\star + 1 \tag{11b}$$

How to Build the Bounds of $G_u(K)$

Algorithm

- We build decreasing sequences of $(F_u(\boldsymbol{w}_k))_k$ and $(F_l(\boldsymbol{w}_k))_k$ such that $F_l(\boldsymbol{w}_k) \leq f(\boldsymbol{w}_k) \leq F_u(\boldsymbol{w}_k)$
- Master node updates $F_u(\boldsymbol{w}_i)_{i=1:k}$ and $F_l(\boldsymbol{w}_i)_{i=1:k}$ at each iteration k
- Define $F_A : \mathbb{N} \mapsto \mathbb{R}$ as the approximation function

$$F_A(t; f(\boldsymbol{w}_{k1}), f(\boldsymbol{w}_{k_2})) = gt + B, \quad t \in [k_1, k_2] \tag{12}$$

$$g = \frac{f(\boldsymbol{w}_{k1}) - f(\boldsymbol{w}_{k_2})}{k_1 - k_2}, \quad B = f(\boldsymbol{w}_{k1}) - gk_1$$

Bounds Updates

$$F_A(t; f(\boldsymbol{w}_{k1}), f(\boldsymbol{w}_{k_2})) = gt + B, \quad t \in [k_1, k_2], \quad (12)$$

$$g = \frac{f(\boldsymbol{w}_{k1}) - f(\boldsymbol{w}_{k_2})}{k_1 - k_2}, \quad B = f(\boldsymbol{w}_{k1}) - gk_1$$

- For $k \leq 2$, set $F_u(\boldsymbol{w}_k) = F_l(\boldsymbol{w}_k) = f(\boldsymbol{w}_k)$
- Define $\delta_k^u = F_u(\boldsymbol{w}_k) - F_u(\boldsymbol{w}_{k-1})$, and $\delta_k^l = F_l(\boldsymbol{w}_k) - F_l(\boldsymbol{w}_{k-1})$
- At each iteration k:
  - $k_{\max}^u = \max\{t | F_u(\boldsymbol{w}_t) > f(\boldsymbol{w}_k), t < k\}$
  - $k_{\max}^l = \max\{t | F_l(\boldsymbol{w}_t) = f(\boldsymbol{w}_t), t < k\}$

$$F_u(\boldsymbol{w}_k) = \begin{cases} f(\boldsymbol{w}_k), & f(\boldsymbol{w}_{k-1}) < f(\boldsymbol{w}_k) < F_u(\boldsymbol{w}_{k-1}) \\ F_u(\boldsymbol{w}_{k-1}) + \delta_k^l, & f(\boldsymbol{w}_k) < f(\boldsymbol{w}_{k-1}) \leq F_l(\boldsymbol{w}_{k_{\max}^l}) \\ F_A(k), & F_u(\boldsymbol{w}_{k-1}) < f(w_k) < F_u(\boldsymbol{w}_{k_{\max}^u}) \end{cases} \quad (13)$$

$$F_l(\boldsymbol{w}_k) = \begin{cases} F_l(\boldsymbol{w}_{k-1}) + \delta_k^u, & f(\boldsymbol{w}_{k-1}) < f(\boldsymbol{w}_k) < F_u(\boldsymbol{w}_{k-1}) \\ F_A(k), & f(\boldsymbol{w}_k) < f(\boldsymbol{w}_{k-1}) \leq F_l(\boldsymbol{w}_{k_{\max}^l}) \\ F_l(\boldsymbol{w}_{k-1}) + \delta_k^u, & F_u(\boldsymbol{w}_{k-1}) < f(w_k) < F_u(\boldsymbol{w}_{k_{\max}^u}) \end{cases} \quad (14)$$

Solution for Problem (8) for Non-convex Losses

Causal Setting

- The FLCau Algorithm finds the optimal or near to the optimal solution

Proposition 2

Let $f(\boldsymbol{w}_k)$ be non-convex. Let $k^\star$ be the solution to problem (8). Then, the FLCau solution k_c is such that $\min\left\{k_c^l, k_c^u\right\} \;\leq\; k_c \;\leq\; \max\left\{k_c^l, k_c^u\right\}$

- $k^\star \leq k_c \leq k^\star + \Delta$
- $f(\boldsymbol{w}_{k_c}) \leq f(\boldsymbol{w}_{k^\star})$
- $G(k_c) \geq G(k^\star)$

Outline

Numerical Results

Parameters

- Star network with one master node
- Perform FL over an image classification task using the MNIST dataset
- M: Number of workers
- Logistic regression loss function as

$$f(\boldsymbol{w}) = \frac{1}{M} \sum_{j \in [M]} \frac{1}{N_j} \sum_{i \in [N_j]} \log \left(1 + e^{-\boldsymbol{w}^T \boldsymbol{x}_{ij} y_{ij}}\right) \tag{15}$$

- Number of bits as the iteration-cost
- $d = 784$ in MNIST dataset
- Consider $|\mathcal{M}_k| = M$

Outline

- State of the Art
- System Model and Problem Formulation
- Solution for Convex Loss Functions
- Solution for Non-convex Loss Functions
- Numerical Results
  - Slotted-ALHOA and CSMA/CA
  - Top-q Sparsification and Lazily Aggregated Quantized Gradient (LAQ) method
- Conclusion

FLCau Application

Latency components of running every iteration of FL

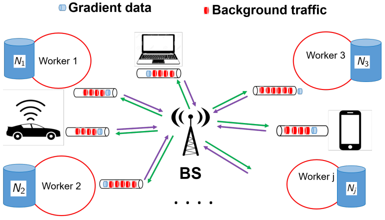


Figure 4: FL problem over wireless networks.

We model the cost $c_k = \sum_{i=1}^{4} \ell_{i,k}$ as the latency for wireless communication and computation, defined as follows:

$\ell_{1,k}$: latency in broadcasting parameters by master node

$\ell_{2,k}$: latency in computing $\boldsymbol{w}_k^j$ for every worker node j

$\ell_{3,k}$: latency in sending $\boldsymbol{w}_k^j$ to master node in a multiple access channel

$\ell_{4,k}$: latency in updating parameters at the master node

Computation latency: Computation latency in master node + Local computation latency

- Computation latency in master node ($\ell_{4,k}$) $\ll$ Local computation latency($\ell_{2,k}$)
- Parallel computations in workers
- All workers wait for the slowest worker before transmission
- The computation latency of each worker $j \in [M]$ obtained as $\ell^j_{2,k} = a^j_k |D_j| / \nu^j_k$ [6]
  - a^j_k is the number of processing cycles to execute one sample of data (cycles/sample)
  - ν^j_k is the central processing unit (cycles/sec)
- The computation latency at each iteration k is upper-bounded by

$$\ell_{2,k} \leq |D| \max_{j \in [M]} \left\{ \frac{a^j_k}{\nu^j_k} \right\} \tag{16}$$

where $|D| = \sum_{j \in [M} |D_j|$.

[6] Nguyen et.al *Efficient Federated Learning Algorithm for Resource Allocation in Wireless IoT Networks*. IEEE Internet of Things Journal 2020.

Iteration Cost

Communication Latency

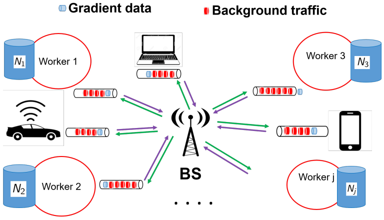


Figure 4: FL problem over wireless networks.

- Multiple Access protocols like Slotted-ALOHA and CSMA/CA
- Local FL parameters are head-of-line packets at each iteration k.
- p_x and p_r are transmission probability, and background packet arrival probability at each time slot

Transitions Probabilities in State Graph

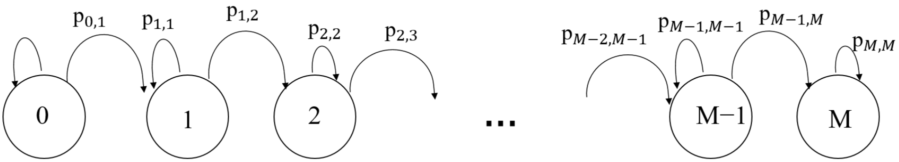


Figure 5: Overall view of the state graph with $M + 1$ states.

- $p_{i,i}$: the probability that no new node transmits. Possible scenarios:
  - Pr{No successful transmission in the system},
  - Pr{Idle time slot},
  - Pr{Just one of the node $j \in \{1, 2, \ldots, i\}$ transmits a background packet successfully}.
- $p_{i,i+1}$: the probability of a new node transmits successfully.

Upper Bound on Communication Cost

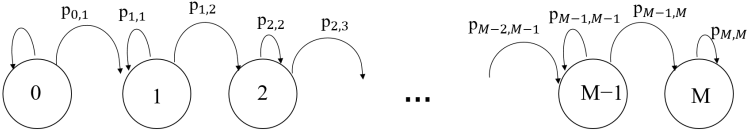


Figure 5: Overall view of the state graph with $M+1$ states.

- t_s: Duration of one time slot (sec)
- $\hat{p}$: Probability of an idle time slot
- $\mathbb{E}\left\{\ell_{3,k}\right\}$: Average communication latency in iteration k

$$\mathbb{E}\left\{\ell_{3,k}\right\} \leq t_s \left(\sum_{i=0}^{M-1} \left\{ p_{i,i+1} + \frac{p_{i,i}}{(1-p_{i,i})^2} \right\} \right), \tag{17}$$

$$p_{i,i} = p_r p_x \sum_{j=1}^{i-1} \frac{(i-1)!}{j!(i-1-j)!} \left\{ p_r^j (1-p_x)^j (1-p_r)^{i-1-j} \right\},$$

$$p_{i,i+1} = (M-i)\, p_x (1-p_x)^{M-i-1}$$

$$\sum_{j=1}^{i} p_r^j (1-p_x)^j (1-p_r)^{i-j}.$$

Numerical Results

Parameters

- Star networks with one master node
- Slotted-ALOHA and CSMA/CA as the uplink channel
- Perform FedAvg over MNIST dataset
- Neural Networks (NNs) prediction loss functions
- Latency as the iteration-cost
- p_x: Transmission probability at each time slot
- p_r: Packet arrival probability in each time slot
- M: Number of workers

Numerical Results: Slotted-ALOHA and CSMA/CA

NNs prediction loss functions

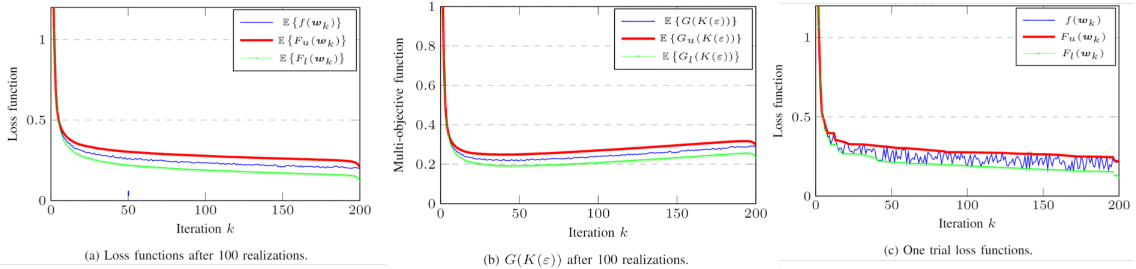
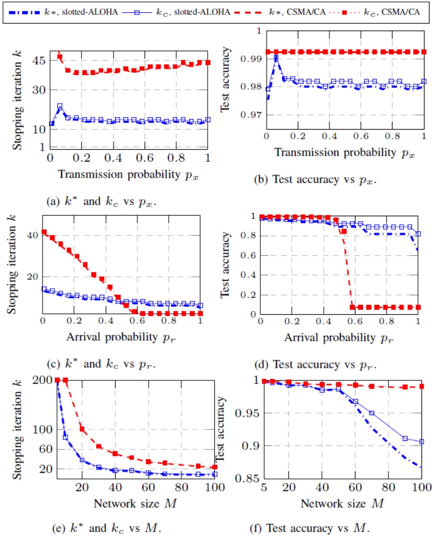


Figure 6

- Bounds correctly track true loss function (a)
- $k_c^u = 39,\ k_c = 43,\ k_c^l = 48$ shows that the difference between optimal and suboptimal number of iterations is small (b)
- Loss function and its bounds after one realization (c) and 100 realizations (a)

Numerical Results: Slotted-ALOHA and CSMA/CA

Stopping Iteration



(a) k^* and k_c vs p_x.

(b) Test accuracy vs p_x.

(c) k^* and k_c vs p_r.

(d) Test accuracy vs p_r.

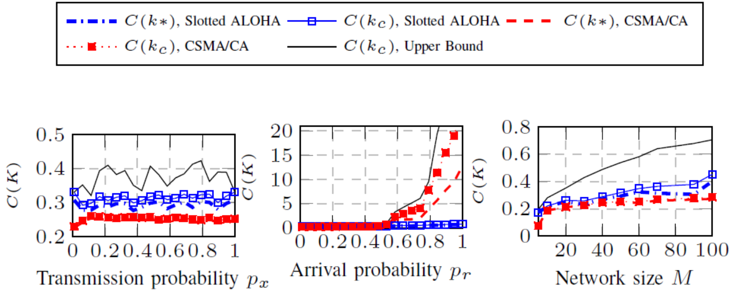
(e) k^* and k_c vs M.

(f) Test accuracy vs M.

Figure 7

Numerical Results: Slotted-ALOHA and CSMA/CA

Iteration-cost



(a) Iteration-cost $C(k^*)$ and $C(k_c)$ vs p_x.

(b) Iteration-cost $C(k^*)$ and $C(k_c)$ vs p_r.

(c) Iteration-cost $C(k^*)$ and $C(k_c)$ vs M.

Figure 8

- The bound on iteration cost c_k in MAC protocols works on the simulation results
- Transmission probability p_x has the least effect on $C(K)$, (a)
- Packet arrival probability has the most effect, (b)
- Number of workers M also affects the iteration cost, but the effect is small, (c)

Outline

- State of the Art
- System Model and Problem Formulation
- Solution for Convex Loss Functions
- Solution for Non-convex Loss Functions
- Numerical Results
  - Slotted-ALHOA and CSMA/CA
  - Top-q Sparsification and Lazily Aggregated Quantized Gradient (LAQ) method
- Conclusion

Top-q Sparsification

Methodology

- Communicate only a fraction (q) of largest elements with full precision [1]
  - Other elements are not communicated
  - New dimension $d_s := \lceil q * d \rceil$, $q \in (0, 1]$
- $\nabla_k^j := \sum_{i \in [N_j]} \nabla_w f(\boldsymbol{w}_k; \boldsymbol{x}_{ij}, y_{ij}), \nabla_k^j \in \mathbb{R}^d$
- $\tilde{\nabla}_k^j :=$ Reduced vector of ∇_k^j, $\tilde{\nabla}_k^j \in \mathbb{R}^{d_s}$
- Estimated global update $\tilde{\boldsymbol{w}}_k := \tilde{\boldsymbol{w}}_{k-1} - \sum_{j \in [M]} \tilde{\nabla}_k^j / M$
- $c_k :=$ number of communication bits
- Calculate $\tilde{G}(k)$ with estimated $\tilde{\boldsymbol{w}}_k$, and $f(\tilde{\boldsymbol{w}}_k)$
- Robustness to non-i.i.d. data is due to mainly two reasons:
  - The frequent communication of weight updates prevents the weights from diverging too far
  - The noise in the stochastic gradients is not amplified by quantization

[1] Sattler et al. *Robust and communication-efficient federated learning from non-iid data*. IEEE Transactions on Neural Networks and Learning Systems 2019.

FLCau and Top-q Sparsification

Characterizing Iteration-Cost

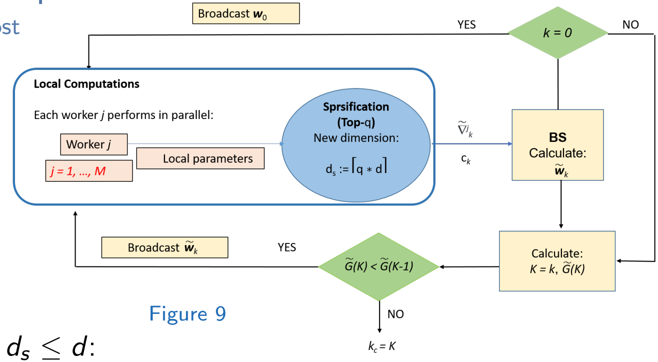


Figure 9

- Dimension-reduction to $d_s \leq d$:
  - $d_s = \lceil q * d \rceil$, $q \in (0, 1]$
  - Each worker j transmits the largest d_s components of its local vector
- Element-wise quantization of each reduced vector
  - $b_t \leq 32$: number of bits of each element when applying top-q method
- The total number of transmitted bits at each iteration k:

$$c_k = M d_s b_t \leq 32 M d_s \tag{18}$$

Lazily Aggregated Quantized Gradient (LAQ)

Methodology

- Reduces the number of worker-to-BS uplink communications [2]
- $\boldsymbol{u}^j(\boldsymbol{w}_k)$: the quantized gradient per worker $j \in \mathcal{M}_k$, $|\mathcal{M}_k| \leq M$
- $\hat{\boldsymbol{w}}_k^j = \begin{cases} \boldsymbol{w}_k, & j \in \mathcal{M}_k \\ \hat{\boldsymbol{w}}_{k-1}^j, & j \notin \mathcal{M}_k \end{cases}$
- $R_k^j := \|\nabla_{\boldsymbol{w}} f(\boldsymbol{w}_k^j) - \boldsymbol{u}^j(\hat{\boldsymbol{w}}_{k-1}^j)\|_\infty$
- Quantization granularity is defined as $\tau := 1/(2^b - 1)$
    - b: number of communication bits
- $\delta \boldsymbol{u}_k^j := \boldsymbol{u}^j(\boldsymbol{w}_k) - \boldsymbol{u}^j(\hat{\boldsymbol{w}}_{k-1}^j) = 2\tau R_k^j \boldsymbol{u}^j(\boldsymbol{w}_k) - R_k^j \mathbf{1}$
    - $\mathbf{1} = [1, \ldots, 1]^T$
- Global update at each iteration in BS: $\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \alpha_k \boldsymbol{\nabla}_k$
    - $\boldsymbol{\nabla}_k := \boldsymbol{\nabla}_{k-1} + \sum_{j \in \mathcal{M}_k} \delta \boldsymbol{u}_k^j$
    - Transmitted by $32 + bd$ bits instead of $32d$

[2] Sun et al. *Lazily aggregated quantized gradient innovation for communication-efficient federated learning*. IEEE Transactions on Pattern Analysis and Machine Intelligence 2020.

FLCau and LAQ

Characterizing Iteration-Cost

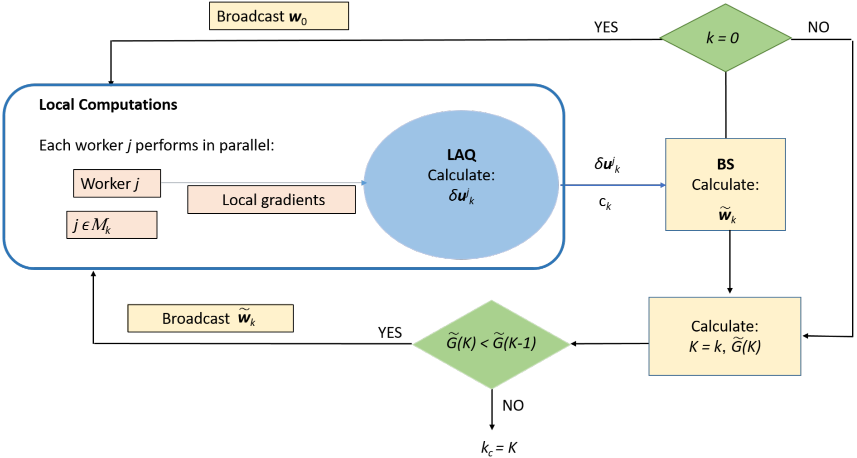


Figure 10

- Element-wise quantization with b bits and vector dimension d

$$c_k = |\mathcal{M}_k|(32 + bd) \leq M(32 + bd) \tag{19}$$

Comparison Between FLCau and Traditional Methods

Table 1: $M = 50,\ |\mathcal{M}_k| = M,\ b_t = 32$

Method	**Stop iteration**	**Total cost ($\times 10^6$ bits)**	**Test accuracy (%)**
FLCau LAQ, $b = 2$	57	4.56	94.2
FLCau LAQ, $b = 10$	43	16.92	87.8
FLCau Top-q, $q = 0.1$	49	6.19	92.4
FLCau Top-q, $q = 0.6$	43	32.4	80.9
FLCau	56	70.24	96.4
FL LAQ, $b = 2$	200	16	98
FL LAQ, $b = 10$	200	78.7	98.7
FL Top-q, $q = 0.1$	200	25	98.9
FL Top-q, $q = 0.6$	200	150.72	97.5
FL	200	250.88	99.02

- Significant reduction in total cost with FLCau
  - Trade off between test accuracy and total cost
  - More than 60% improvement in cost reduction
  - Test accuracy reduction of 1-15 %

Numerical Results

FLCau applied on top of LAQ and top-q

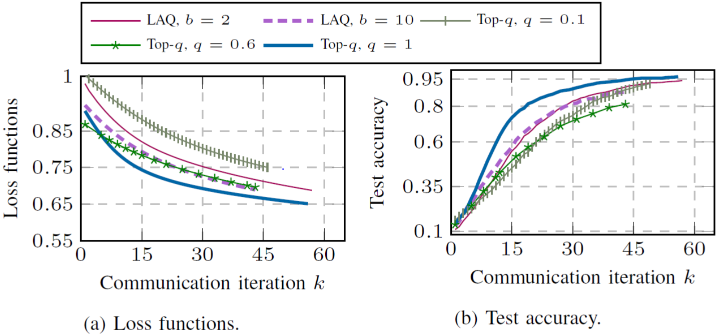


Figure 11

- Numerical results with FLCau, $M = 50$
- Test accuracy while applying FLCau:
  - Top-q with $q = 0.1$ outperforms the case $q = 0.6$ by 13%
  - LAQ with $b = 2$ has the closest accuracy (94%) to the FLCau (96%)

Outline

- State of the Art
- System Model and Problem Formulation
- Convex Loss Functions
- Non-Convex Loss Functions
- Numerical Results
- Conclusion

Conclusion

- We proposed an optimization of the communication-computation costs for solving an FL training problem
- We established a novel cost-efficient FL algorithm (FLCau) for both convex and non-convex stochastic loss functions.
- FLCau can be applied on top of existing cost-efficient methods, such as Top-q and LAQ
- Numerical results indicated that FLCau reduces the total cost by 60% while achieves a near-optimal test accuracy

Thanks for your attention.

Questions?