Sparse Incremental Aggregation in Multi-Hop Federated Learning

Sourav Mukherjee*, Nasrin Razmi*, Armin Dekorsy*, Petar Popovski†*, Bho Matthiesen*
*University of Bremen, Department of Communications Engineering, Germany
†Aalborg University, Department of Electronic Systems, Denmark
email: {mukherjee, razmi, dekorsy, matthiesen}@ant.uni-bremen.de, petarp@es.aau.dk

***Abstract*—This paper investigates federated learning (FL) in a multi-hop communication setup, such as in constellations with inter-satellite links. In this setup, part of the FL clients are responsible for forwarding other client's results to the parameter server. Instead of using conventional routing, the communication efficiency can be improved significantly by using in-network model aggregation at each intermediate hop, known as *incremental aggregation (IA)*. Prior works [1] have indicated diminishing gains for IA under gradient sparsification. Here we study this issue and propose several novel correlated sparsification methods for IA. Numerical results show that, for some of these algorithms, the full potential of IA is still available under sparsification without impairing convergence. We demonstrate a 15× improvement in communication efficiency over conventional routing and a 11× improvement over state-of-the-art (SoA) sparse IA.**

***Index Terms*—Federated learning, correlated sparsification, gradient sparsification, cooperative communications, multi-hop network, in-network computing**

I. Introduction

Federated learning (FL) [2] is an instance of distributed machine learning (ML) over bandwidth-restricted communication networks, in which a large number of clients use the local sets and collaboratively train an ML model. This process is orchestrated by a central parameter server (PS), which is responsible for synchronizing intermediate results among the clients. This involves collecting intermediate results $\{\boldsymbol{g}_k^t\}_k$ from all clients, aggregating these values into a new global iterate $\boldsymbol{w}^{t+1}$ of the ML model parameters, and distributing $\boldsymbol{w}^{t+1}$ back to all clients. As contemporary ML models can consist of billions of parameters [3], communication efficiency in this synchronization phase is paramount.

Here, we are interested in FL over multi-hop (MH) networks, where some of the FL clients are responsible for forwarding other client's results to the PS. Such a setup arises in the implementation of FL in satellite constellations [4], [5], when inter-satellite links are used for communicating with the PS [1], [6]. Similar setups are also applicable to FL in wireless mesh and multi-hop sensor networks [7]. It is shown in [1], [6], [7] that a network topology-aware implementation of FL, leveraging in-network computing, leads to a massive reduction in the communication load during the result-collection phase. To elaborate, consider the MH network in Fig. 1, where nodes $1, \dots, K$ are FL clients. Using conventional routing to transmit $\{\boldsymbol{g}_k^t\}_k$ results in a total of $1 + 2 + \cdots + (K-1) + K = (K^2 + K)/2$ transmissions the size of $\boldsymbol{g}_k^t$. However, the PS is primarily interested in the weighted sum of $\{\boldsymbol{g}_k^t\}_k$ to compute the new model iterate. By computing this weighted sum incrementally within the network, i.e., each hop combines their own update with the results of previous hops before forwarding, each node needs only transmit a single vector of size $\boldsymbol{g}_k^t$, resulting in K transmissions in total. However, it has been observed in [1] that the efficiency of this incremental aggregation (IA) procedure is reduced massively when combined with gradient sparsification methods such as Top-Q [8], [9]. This is because individual gradient sparsification at each node leads to (almost) uncorrelated sparsification supports. The result is that the number of non-zero elements in the partial aggregate increases with each hop.

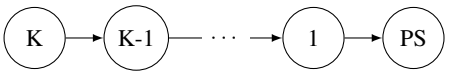


Fig. 1. Multi-hop federated learning system.

In this paper, we consider sparse IA and propose several approaches to the issues outlined above. After establishing the system model in Section II, we analyze the shortcomings of state-of-the-art (SoA) sparse IA and take two different angles towards improving it in Section III. Both methods rely on intentionally creating correlation in the sparsification procedure. In Section IV, we combine our methods with time correlated sparsification (TCS) [10] to further increase this correlation. The communication cost of the proposed algorithms is discussed in Section V, and evaluated numerically in Section VI.

Notation: The L^2 vector norm is $\|\boldsymbol{x}\|$, $\|\boldsymbol{x}\|_0$ is the number of nonzero elements in $\boldsymbol{x}$, and $\mathbb{1}(\boldsymbol{x})$ is the indicator vector of $\boldsymbol{x}$. The Hadamard product of two vectors $\boldsymbol{a}, \boldsymbol{b}$ is $\boldsymbol{a} \circ \boldsymbol{b}$. The operations $\lceil \cdot \rceil$ and $\lfloor \cdot \rceil$ round to the nearest greater and nearest integer, respectively. The function $S(\boldsymbol{x}, Q)$ returns the Top-Q sparsification of $\boldsymbol{x}$ and $s(\boldsymbol{x}, Q) = \mathbb{1}(S(\boldsymbol{x}, Q))$ returns the corresponding sparsification mask.

II. System Model

Consider the MH communication system in Fig. 1 with $K+1$ nodes, where node k, $k = 1, \dots, K-1$, is connected to nodes

This work is supported by the German Research Foundation (DFG) under Grant EXC 2077 (University Allowance).

$k-1$ and $k+1$, node K connects to node $K-1$, and node 0, denoted as PS, is connected to node 1. These nodes form a FL system in which nodes $1,\dots,K$ collaborate to solve an ML training problem $\min_{\boldsymbol{w}\in\mathbb{R}^d} \frac{1}{D}\sum_{k=1}^{K}\sum_{\boldsymbol{x}\in\mathcal{D}_k} f(\boldsymbol{x};\boldsymbol{w})$, with per-sample loss function $f(\boldsymbol{x};\boldsymbol{w})$, d-dimensional model parameter vector $\boldsymbol{w}$, and $\mathcal{D}_k$ the data set of client k. We define $D_k = |\mathcal{D}_k|$ and the total number of samples $D = \sum_k D_k$. Nodes $1,\dots,K$ are also referred to as clients.

Local data sets $\mathcal{D}_k$ are not shared among clients, making a distributed solution of the training problem necessary. This is an iterative procedure orchestrated by the PS. In iteration t, this PS distributes the current model parameter vector $\boldsymbol{w}^t$ to all clients. Then, each client k computes a local update $\boldsymbol{w}_k^t$ to $\boldsymbol{w}^t$, e.g., by performing one or several steps of stochastic gradient descent (SGD), and transmits the result to the PS. This is done in the form of the effective gradient $\boldsymbol{g}_k^t = \boldsymbol{w}_k^t - \boldsymbol{w}^t$. After collecting all updates $\{\boldsymbol{g}_k^t\}_k$, the PS computes a new iteration of the model parameters as

$$\boldsymbol{w}^{t+1} = \boldsymbol{w}^t + \frac{1}{D}\sum\nolimits_{k=1}^{K} D_k \boldsymbol{g}_k^t. \tag{1}$$

This process is repeated until convergence.

We are primarily interested in the aggregation phase. To this end, we are assuming that the effective gradients $\{\boldsymbol{g}_k^t\}_k$ are obtained by *some* means at the clients and that the PS is only interested in their aggregate value $\sum_{k=1}^{K} D_k \boldsymbol{g}_k^t$.

A. Sparse Incremental Aggregation

Observe from (1) that the PS is not interested in individual updates but only in the weighted sum $\sum_{k=1}^{K} D_k \boldsymbol{g}_k^t$. This is exploited in IA as follows. Instead of forwarding its own gradient $\boldsymbol{g}_k^t$ and the previous hop's gradients $\{\boldsymbol{g}_j^t\}_{j=k-1}^{K}$ separately, node k waits for all previous nodes and computes the partial aggregate $\boldsymbol{\gamma}_k^t = \sum_{i=k}^{K} D_i \boldsymbol{g}_i^t$. This is then transmitted to the PS via the next hop. Applying this approach in each node k, $k=1,\dots,K-1$, we observe that node k will only receive a single transmission from node $k+1$ with the partial aggregate $\boldsymbol{\gamma}_{k+1}^t$. Then, it computes its own partial aggregate as

$$\boldsymbol{\gamma}_k^t = \boldsymbol{\gamma}_{k+1}^t + D_k \boldsymbol{g}_k^t \tag{2}$$

and forwards it to node $k-1$. Thus, the PS receives $\boldsymbol{\gamma}_1^t = \sum_{k=1}^{K} D_k \boldsymbol{g}_k^t$ and computes $\boldsymbol{w}^{t+1} = \boldsymbol{w}^t + \frac{1}{D}\boldsymbol{\gamma}_1^t$.

Our goal is to combine IA with Top-Q gradient sparsification [8], [9]. The Top-Q procedure sets all gradient entries except the Q largest magnitude values to zero and transmits the resulting vector in sparse representation. This results in a significant bandwidth reduction, as only the nonzero entries (and their indices) need to be transmitted. Top-Q sparsification is commonly implemented with an error feedback mechanism to improve convergence. In particular, let $\boldsymbol{e}_k^{t-1}$ be node k's sparsification error from the last iteration. Then, the error-compensated effective gradient at node k is $\tilde{\boldsymbol{g}}_k^t = \boldsymbol{g}_k^t + \boldsymbol{e}_k^{t-1}$. Based on this vector and the incoming $\boldsymbol{\gamma}_{k+1}^t$, a new sparse partial aggregate $\boldsymbol{\gamma}_k^t$ and error vector $\boldsymbol{e}_k^t$ are computed.

In this paper, we explore several methods for computing $\boldsymbol{\gamma}_k$ from $\tilde{\boldsymbol{g}}_k^t$ and $\boldsymbol{\gamma}_{k+1}^t$. The SoA approach [1] is a direct concatenation of IA and Top-Q sparsification applied to $\tilde{\boldsymbol{g}}_k^t$. That is, node k computes $\bar{\boldsymbol{g}}_k^t = S(\tilde{\boldsymbol{g}}_k^t, Q)$ and $\boldsymbol{e}_k^t = \tilde{\boldsymbol{g}}_k^t - \bar{\boldsymbol{g}}_k^t$, where $S(\cdot,Q)$ is the Top-Q sparsification operation. The outgoing partial aggregate $\boldsymbol{\gamma}_k^t$ is then computed as $\boldsymbol{\gamma}_{k+1}^t + D_k \bar{\boldsymbol{g}}_k^t$. This results in Algorithm 1 from [1], where lines 2–4 perform Top-Q sparsification and line 5 implements IA. Note that scaling by D_k is, equivalently, done in line 2 for consistency with later algorithms.

Algorithm 1 Sparse incremental aggregation at node k [1]

1: **Input** $\boldsymbol{g}_k^t$, $\boldsymbol{\gamma}_{k+1}^t$
2: Error feedback $\tilde{\boldsymbol{g}}_k^t \leftarrow D_k \boldsymbol{g}_k^t + \boldsymbol{e}_k^{t-1}$
3: Sparsification $\bar{\boldsymbol{g}}_k^t \leftarrow S(\tilde{g}_k^t, Q)$
4: Update error $\boldsymbol{e}_k^t \leftarrow \tilde{\boldsymbol{g}}_k^t - \bar{\boldsymbol{g}}_k^t$
5: Incremental Aggregation $\boldsymbol{\gamma}_k^t \leftarrow \bar{\boldsymbol{g}}_k^t + \boldsymbol{\gamma}_{k+1}^t$
6: **Return** $\boldsymbol{\gamma}_k^t$

III. Sparse Incremental Aggregation, Revisited

Consider the sparse IA operation in Line 5 of Algorithm 1. If $\bar{\boldsymbol{g}}_k^t$ and $\boldsymbol{\gamma}_{k+1}^t$ have their nonzero elements in exactly the same positions, i.e., they have the same sparsification support, the outgoing partial aggregate $\boldsymbol{\gamma}_k$ will have exactly Q nonzero elements. This, however, is usually not the case and the number of nonzero elements in $\boldsymbol{\gamma}_k^t$ is $\max\{Q, \|\boldsymbol{\gamma}_{k+1}^t\|_0\} \le \|\boldsymbol{\gamma}_k^t\|_0 \le Q + \|\boldsymbol{\gamma}_{k+1}^t\|_0$, with a strong tendency towards the upper bound as K increases and Q decreases [1]. Indeed, the results in [1] indicate that, as $K \to \infty$, the gain of IA over conventional multiple unicast transmissions completely vanishes for $Q < d$.

A. An Error Minimization Perspective on Sparse IA

To gain additional insight into this problem, consider a conventional FL setup with direct client-PS links. Gradient compression, which includes sparsification, is applied to conserve bandwidth in the client-PS link. For any compressor $C(\boldsymbol{x})$ and node k, the compression error is $\|\tilde{\boldsymbol{g}}_k^t - C(\tilde{\boldsymbol{g}}_k^t)\|^2$ and a compressor is considered optimal if it minimizes this error. Restricting the choice of compressors to the set of sparsification functions $\mathcal{S}$, it is well established that Top-Q sparsification is optimal under a strict communication budget of transmitting at most Q nonzero elements per iteration [11, Lemma 2]. That is, the optimization problem

$$\min_{C\in\mathcal{S}} \ \|\tilde{\boldsymbol{g}}_k^t - C(\tilde{\boldsymbol{g}}_k^t)\|^2 \quad \text{s.t.} \quad \|C(\tilde{\boldsymbol{g}}_k^t)\|_0 \le Q, \tag{3}$$

is solved by $C(\boldsymbol{x}) = S(\boldsymbol{x}, Q)$.

Returning to MH aggregation and Algorithm 1, we can make two observations: 1) The outgoing transmission is $\boldsymbol{\gamma}_k^t$ with relevant compression error $\|\boldsymbol{\gamma}_k^t - C(\boldsymbol{\gamma}_k^t)\|^2$; and 2) the effective transmissions budget is $\widetilde{Q} = \|\boldsymbol{\gamma}_{k+1}^t + S(\tilde{\boldsymbol{g}}_k^t, Q)\|_0 \ge Q$. Thus, the error minimization problem corresponding to (3) is

$$\min_{C\in\mathcal{S}} \ \|\boldsymbol{\gamma}_k^t - C(\boldsymbol{\gamma}_{k+1}^t, \tilde{\boldsymbol{g}}_k^t)\|^2 \ \text{s.t.} \ \|C(\boldsymbol{\gamma}_{k+1}^t, \tilde{\boldsymbol{g}}_k^t)\|_0 \le \widetilde{Q}. \tag{4}$$

It is straightforward to show that Algorithm 1 is strictly suboptimal with respect to (4).

Proposition 1: $C(\boldsymbol{\gamma}_{k+1}^t, \tilde{\boldsymbol{g}}_k^t) = S(\tilde{\boldsymbol{g}}_k^t, Q)$ is strictly suboptimal with respect to (4) unless the sparsification supports of $\boldsymbol{\gamma}_{k+1}^t$ and $\tilde{\boldsymbol{g}}_k^t$ are identical.

Proof: Let $\boldsymbol{m}_k^t = s(\tilde{\boldsymbol{g}}_k^t, Q) = \mathbb{1}(S(\boldsymbol{x}, Q))$ be the Top-Q sparsification mask and $\widetilde{\boldsymbol{m}}_{k+1}^t = \mathbb{1}(\boldsymbol{\gamma}_{k+1}^t)$ the mask corresponding to the sparsification support of $\boldsymbol{\gamma}_{k+1}^t$, where $\mathbb{1}(\boldsymbol{x})$ is the indicator vector of $\boldsymbol{x}$. Consider $C(\boldsymbol{\gamma}_{k+1}^t, \tilde{\boldsymbol{g}}_k^t) = \boldsymbol{\gamma}_{k+1}^t + \mathbb{1}(\boldsymbol{m}_k^t + \widetilde{\boldsymbol{m}}_{k+1}^t) \circ \tilde{\boldsymbol{g}}_k^t$. Then, the objective of (4) satisfies

$$\begin{aligned} &\left\| \left(\mathbf{1} - \mathbb{1}(\boldsymbol{m}_k^t + \widetilde{\boldsymbol{m}}_{k+1}^t)\right) \circ \tilde{\boldsymbol{g}}_k^t \right\|^2 \\ &= \sum\nolimits_{i \notin \mathcal{I}(\boldsymbol{m}_k^t) \cup \mathcal{I}(\widetilde{\boldsymbol{m}}_{k+1}^t)} |g_{k,i}^t|^2 \leq \sum\nolimits_{i \notin \mathcal{I}(\boldsymbol{m}_k^t)} |g_{k,i}^t|^2, \end{aligned} \quad (5)$$

where $\mathbf{1}$ is the all ones vector, $g_{k,i}^t$ is the ith element of $\boldsymbol{g}_k^t$, and $\mathcal{I}(\boldsymbol{x})$ is a set containing the indices of nonzero elements in $\boldsymbol{x}$. The inequality in (5) is strict unless $\sum_{i \notin \mathcal{I}(\widetilde{\boldsymbol{m}}_{k+1}^t) \setminus \mathcal{I}(\boldsymbol{m}_k^t)} |g_{k,i}^t|^2 = 0$, which is the case either if the corresponding elements in $\boldsymbol{g}_k^t$ are zero or $\widetilde{\boldsymbol{m}}_{k+1}^t = \boldsymbol{m}_k^t$. ■

B. Reduced-Error Sparse Incremental Aggregation

The proof of Proposition 1 suggests a direct way to improve Algorithm 1 without additional communication cost. Instead of transmitting only the nonzero elements in $\boldsymbol{g}_k^t$ returned by Top-Q sparsification, node k can transmit additional gradient elements within the nonzero positions of $\boldsymbol{\gamma}_{k+1}^t$. This is described in Algorithm 2, where the sparsification in line 3 of Algorithm 1 is replaced by lines 3–5. In particular, line 3 computes the Top-Q sparsification mask for $\boldsymbol{g}_k^t$, line 4 retrieves the sparsification mask of the incoming partial aggregate $\boldsymbol{\gamma}_{k+1}^t$, and line 5 sparsifies $\boldsymbol{g}_k^t$ after combining both sparsification masks.

Algorithm 2 Reduced-error sparse IA at node k

1: **Input** $\boldsymbol{g}_k^t$, $\boldsymbol{\gamma}_{k+1}^t$
2: Error feedback $\tilde{\boldsymbol{g}}_k^t \leftarrow D_k \boldsymbol{g}_k^t + \boldsymbol{e}_k^{t-1}$
3: Local sparsification mask $\boldsymbol{m}_k^t \leftarrow s(\tilde{\boldsymbol{g}}_k^t, Q)$
4: Incoming sparsification mask $\widetilde{\boldsymbol{m}}_{k+1}^t \leftarrow \mathbb{1}(\gamma_{k+1}^t)$
5: Sparsification $\bar{\boldsymbol{g}}_k^t \leftarrow \mathbb{1}(\boldsymbol{m}_k^t + \widetilde{\boldsymbol{m}}_{k+1}^t) \circ \tilde{\boldsymbol{g}}_k^t$
6: Update error $\boldsymbol{e}_k^t \leftarrow \tilde{\boldsymbol{g}}_k^t - \bar{\boldsymbol{g}}_k^t$
7: Incremental Aggregation $\boldsymbol{\gamma}_k^t \leftarrow \bar{\boldsymbol{g}}_k^t + \boldsymbol{\gamma}_{k+1}^t$
8: **Return** $\boldsymbol{\gamma}_k^t$

This algorithm has the same communication cost as Algorithm 1 at a lower sparsification error. Since this directly corresponds to more information being transmitted to the PS, better training performance (in terms of accuracy) is expected. However, Algorithm 2 is neither optimal with respect to (4) nor does it adhere to a strict communication budget.

C. Constant-Length Sparse Incremental Aggregation

Equation (4) also offers insight into the design of a sparse IA strategy that strictly adheres to a communication budget of Q nonzero elements per hop. Indeed, the optimal sparsification strategy with respect to (4) is $C(\boldsymbol{\gamma}_{k+1}^t, \tilde{\boldsymbol{g}}_k^t) = S(\tilde{\boldsymbol{g}}_k^t + \boldsymbol{\gamma}_{k+1}^t, \widetilde{Q})$ and by setting $\widetilde{Q} = Q$, the outgoing partial aggregate $\boldsymbol{\gamma}_k^t = S(\tilde{\boldsymbol{g}}_k^t + \boldsymbol{\gamma}_{k+1}^t, Q)$ will have at most Q nonzero entries. This immediately leads to Algorithm 3. Observe that the order of error feedback and IA in lines 2 and 3 is inconsequential, as long as the sparsification error in line 5 is tracked correctly.

Algorithm 3 Constant-length sparse IA at node k

1: **Input** $\boldsymbol{g}_k^t$, $\boldsymbol{\gamma}_{k+1}^t$
2: Error feedback $\tilde{\boldsymbol{g}}_k^t \leftarrow D_k \boldsymbol{g}_k^t + \boldsymbol{e}_k^{t-1}$
3: Incremental Aggregation $\tilde{\boldsymbol{\gamma}}_k^t \leftarrow \tilde{\boldsymbol{g}}_k^t + \boldsymbol{\gamma}_{k+1}^t$
4: Sparsification $\boldsymbol{\gamma}_k^t \leftarrow S(\tilde{\boldsymbol{\gamma}}_k^t, Q)$
5: Update error $\boldsymbol{e}_k^t \leftarrow \tilde{\boldsymbol{\gamma}}_k^t - \boldsymbol{\gamma}_k^t$
6: **Return** $\boldsymbol{\gamma}_k^t$

IV. Time-Correlated Sparse IA

In Section III, we improved upon Algorithm 1 in two different ways. Algorithm 2 has exactly the same communication cost as Algorithm 1, but utilizes the bandwidth more efficiently. Instead, Algorithm 3 fixes the issue of increasing communication cost with each hop and, thus, recovers the communication efficiency of IA. These strategies represent two extremes among potential improvements of Algorithm 1: we can either embrace the increasing communication cost and reduce the sparsification error, or we can strictly enforce the communications budget and obtain the superior communication performance of IA. A controllable trade-off between these two extremes can be obtained by combining sparse IA with TCS. The central idea of TCS [10] is to compute a Top-Q sparsification mask from the global model parameter vector. This leads to identical sparsification supports at all clients. Temporal dynamics in the model development are captured through a small number Q_L of local additions to the global mask, which implements the desired trade-off.

A. Time-Correlated Sparse Incremental Aggregation

The combination of IA and TCS operates on the same principles as Algorithm 2. Let Q_G and Q_L denote the number of nonzero elements in the global and local sparsification masks, respectively. With $\boldsymbol{w}^t$ and $\boldsymbol{w}^{t-1}$ the current and previous iteration of the global model parameters, respectively, the effective gradient for the global model parameters is $\boldsymbol{w}^t - \boldsymbol{w}^{t-1}$. Then, the global sparsification mask is obtained from Top-Q_G sparsification of $\boldsymbol{w}^t - \boldsymbol{w}^{t-1}$, i.e., $\boldsymbol{m}^t = s(\boldsymbol{w}^t - \boldsymbol{w}^{t-1}, Q_G)$. The local sparsification mask $\boldsymbol{m}_k^t$ should capture the most relevant local gradient entries not yet part of the global mask. Thus, node k computes $\boldsymbol{m}_k^t$ from the error-compensated gradient $\tilde{\boldsymbol{g}}_k^t$ after applying the global mask, i.e., $\boldsymbol{m}_k^t = s((\mathbf{1} - \boldsymbol{m}^t) \circ \tilde{\boldsymbol{g}}_k^t, Q_L)$. Similarly to Algorithm 2, node k can transmit further nonzero elements within the nonzero positions of $\boldsymbol{\gamma}_{k+1}^t$. Hence, the sparsified effective local gradient is $\bar{\boldsymbol{g}}_k^t = \mathbb{1}(\boldsymbol{m}^t + \boldsymbol{m}_k^t + \widetilde{\boldsymbol{m}}_{k+1}^t) \circ \tilde{\boldsymbol{g}}_k^t$ with $\widetilde{\boldsymbol{m}}_{k+1}^t = \mathbb{1}(\boldsymbol{\gamma}_{k+1}^t) - \boldsymbol{m}^t$.

Since the global mask $\boldsymbol{m}^t$ is known by all participants, it is not necessary to transmit the indices of these nonzero entries. This leads to a considerable bandwidth reduction and a mixed storage format for the outgoing partial aggregate $\boldsymbol{\gamma}_k^t = [\boldsymbol{\Gamma}_k^t,\ \boldsymbol{\Lambda}_k^t]$, where $\boldsymbol{\Gamma}_k^t$ contains the nonzero elements due to the global sparsification mask and $\boldsymbol{\Lambda}_k^t$ stores the elements due to local sparsification. Thus, the IA operation is

$$\begin{aligned} \boldsymbol{\Gamma}_k^t &= \boldsymbol{\Gamma}_{k+1}^t + \boldsymbol{m}^t \circ \bar{\boldsymbol{g}}_k^t \\ \boldsymbol{\Lambda}^t &= \boldsymbol{\Lambda}_{k+1}^t + (\mathbf{1} - \boldsymbol{m}^t) \circ \bar{\boldsymbol{g}}_k^t \end{aligned} \quad (6)$$

The complete procedure is stated in Algorithm 4, which extends Algorithm 2 with TCS in lines 3, 4, 6, and 8.

Algorithm 4 Time-correlated sparse IA at node k

1: **Input** $\boldsymbol{g}_k^t$, $\boldsymbol{\gamma}_{k+1}^t = \left[\boldsymbol{\Gamma}_{k+1}^t,\ \boldsymbol{\Lambda}_{k+1}^t\right]$
2: Error feedback $\tilde{\boldsymbol{g}}_k^t \leftarrow D_k \boldsymbol{g}_k^t + \boldsymbol{e}_k^{t-1}$
3: Retrieve global mask $\boldsymbol{m}^t \leftarrow s(\boldsymbol{w}^t - \boldsymbol{w}^{t-1}, Q_G)$
4: Local sparsification mask $\boldsymbol{m}_k^t \leftarrow s((\mathbf{1} - \boldsymbol{m}^t) \circ \tilde{\boldsymbol{g}}_k^t, Q_L)$
5: Incoming sparsification mask $\widetilde{\boldsymbol{m}}_{k+1}^t \leftarrow \mathbb{1}(\boldsymbol{\gamma}_{k+1}^t) - \boldsymbol{m}^t$
6: Sparsification $\bar{\boldsymbol{g}}_k^t \leftarrow \mathbb{1}(\boldsymbol{m}^t + \boldsymbol{m}_k^t + \widetilde{\boldsymbol{m}}_{k+1}^t) \circ \tilde{\boldsymbol{g}}_k^t$
7: Update error $\boldsymbol{e}_k^t \leftarrow \tilde{\boldsymbol{g}}_k^t - \bar{\boldsymbol{g}}_k^t$
8: Incremental Aggregation as in (6)
9: **Return** $\boldsymbol{\gamma}_k^t = \left[\boldsymbol{\Gamma}_k^t,\ \boldsymbol{\Lambda}_k^t\right]$

B. Constant-Length Time-Correlated Sparse IA

The rationale behind communicating a small amount of locally selected nonzero elements is to allow new elements to enter the global mask during the training process. This is also realized by the constant-length sparse IA procedure in Algorithm 3, which we combine with TCS in Algorithm 5. The vectors $\boldsymbol{e}_k^t$ and $\boldsymbol{\Lambda}_k^t$ are both implemented as d-dimensional sparse vectors. It is important to apply the error feedback in $\tilde{\boldsymbol{g}}_k^t$ (line 2) and not only in the computation of $\tilde{\boldsymbol{\Lambda}}_k^t$ (line 4), as the global mask might change between iterations.

Algorithm 5 Constant-length time-correlated sparse IA

1: **Input** $\boldsymbol{g}_k^t$, $\boldsymbol{\gamma}_{k+1}^t = \left[\boldsymbol{\Gamma}_{k+1}^t,\ \boldsymbol{\Lambda}_{k+1}^t\right]$
2: Error feedback $\tilde{\boldsymbol{g}}_k^t \leftarrow D_k \boldsymbol{g}_k^t + \boldsymbol{e}_k^{t-1}$
3: Retrieve global mask $\boldsymbol{m}^t \leftarrow s(\boldsymbol{w}^t - \boldsymbol{w}^{t-1}, Q_G)$
4: Incremental Aggregation $\boldsymbol{\Gamma}_k^t \leftarrow \boldsymbol{\Gamma}_{k+1}^t + \boldsymbol{m}^t \circ \tilde{\boldsymbol{g}}_k^t$
$\tilde{\boldsymbol{\Lambda}}_k^t \leftarrow \boldsymbol{\Lambda}_{k+1}^t + (\mathbf{1} - \boldsymbol{m}^t) \circ \tilde{\boldsymbol{g}}_k^t$
5: Local sparsification $\boldsymbol{\Lambda}_k^t \leftarrow S(\tilde{\boldsymbol{\Lambda}}_k^t, Q_L)$
6: Update error $\boldsymbol{e}_k^t \leftarrow \tilde{\boldsymbol{\Lambda}}_k^t - \boldsymbol{\Lambda}_k^t$
7: **Return** $\boldsymbol{\gamma}_k^t = \left[\boldsymbol{\Gamma}_k^t,\ \boldsymbol{\Lambda}_k^t\right]$

V. Communication Cost

The primary motivation for sparsification and IA is to reduce the communication cost of FL. The communication cost (in iteration t) of Algorithms 1–5 directly depends on the number of transmitted nonzero elements $\sum_{k=1}^K \|\boldsymbol{\gamma}_k^t\|_0$ and the storage representation of each element. For the time-correlated sparse IA method in Algorithm 4, the number of outgoing nonzero elements at node k is $\|\boldsymbol{\gamma}_k^t\|_0 = \|\boldsymbol{\Gamma}_k^t\|_0 + \|\boldsymbol{\Lambda}_k^t\|_0$. While $\|\boldsymbol{\Gamma}_k^t\|_0$ is deterministically Q_G, $\|\boldsymbol{\Lambda}_k^t\|_0$ is a random number. Each nonzero element in $\boldsymbol{\Lambda}_k^t$ requires ω bit for the numerical value and additional $\lceil \log_2 d \rceil$ bit to store its location in $\boldsymbol{\gamma}_k^t$, while each element in $\boldsymbol{\Gamma}_k^t$ only requires a total of ω bit. Thus, the expected total communication cost for Algorithm 4 is

$$\sum_{k=1}^K \left(\omega \mathbb{E}\left[\|\boldsymbol{\Gamma}_k^t\|_0\right] + (\omega + \lceil \log_2 d \rceil)\, \mathbb{E}\left[\|\boldsymbol{\Lambda}_k^t\|_0\right]\right) = K\omega Q_G + (\omega + \lceil \log_2 d \rceil) \sum_{k=1}^K \mathbb{E}\left[\|\boldsymbol{\Lambda}_k^t\|_0\right]. \quad (7)$$

Exact analytical expressions for $\mathbb{E}\left[\|\boldsymbol{\Lambda}_k^t\|_0\right]$ are challenging to obtain. However, the following upper bound can be derived along the lines of [1, Prop. 1] by observing that $\boldsymbol{\Lambda}_k^t$ is effectively a vector of dimension $d - Q_G$. The proof is omitted due to space limitations.

Proposition 2: The expected number of nonzero elements due to local sparsification communicated by Algorithm 4 over K hops is upper bounded as

$$\sum_{k=1}^K \mathbb{E}\left[\|\boldsymbol{\Lambda}_k^t\|_0\right] \leq (d - Q_G) \left(K + 1 - \tfrac{d-Q_G}{Q_L}\left(1 - \left(1 - \tfrac{Q_L}{d-Q_G}\right)^{K+1}\right)\right) \quad (8)$$

if $Q_L > 0$ and zero otherwise.

From a communications cost perspective, Algorithms 1 and 2 are equivalent to Algorithm 4 with $Q_G = 0$ and $Q_L = Q$. Furthermore, the communications cost of Algorithm 5 follows from (7) by observing that $\mathbb{E}\left[\|\boldsymbol{\Lambda}_k^t\|_0\right] = Q_L$ as $K\omega Q_G + (\omega + \lceil \log_2 d \rceil)\, K Q_L$. Finally, since Algorithm 3 might be regarded as a special case of Algorithm 5 with $Q_G = 0$ and $Q_L = Q$, its communication cost is $KQ\,(\omega + \lceil \log_2 d \rceil)$.

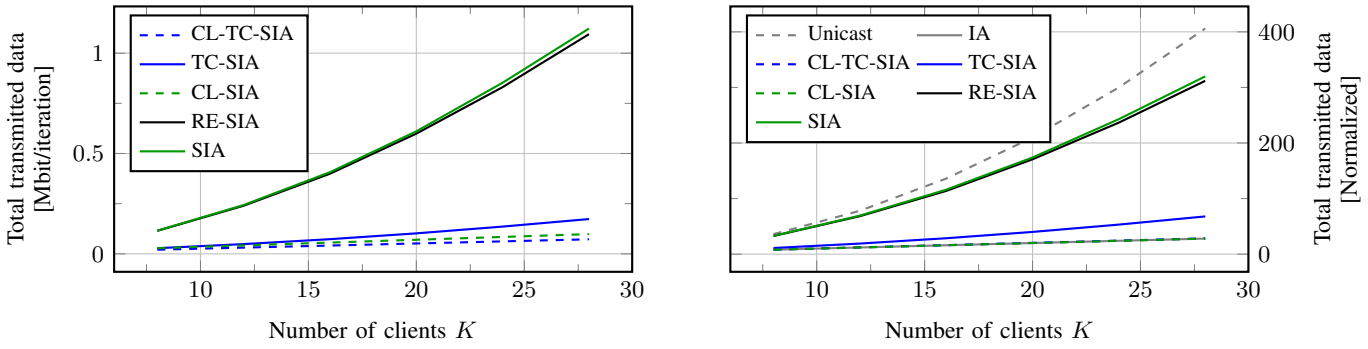
VI. Numerical Evaluation

We evaluate the performance of the proposed sparse IA methods for a logistic regression model with $d = 7850$ trainable parameters on the MNIST data set [12]. This model is trained using SGD with batch size 20 and learning rate 0.1. The first experiment uses a fixed $Q = 78$, corresponding to retaining 1 % of nonzero elements. Following [10], we set $Q_L = 8$ in Algorithms 4 and 5 to 10 % of Q, and $Q_G = Q - Q_L$. We refer to Algorithms 1–5 as SIA, RE-SIA, CL-SIA, TC-SIA, and CL-TC-SIA, respectively, where CL and TC refer to the constant-length and time-correlated properties, respectively.

Figure 2a shows the total data transmitted in the aggregation step of a single iteration, averaged over the complete training process. As expected, the communication cost for SIA and RE-SIA is significantly higher than for the other algorithms. The TC-SIA approach shows the same quadratic growth, but at a much reduced rate. Finally, the CL algorithms are transmitting the smallest amount of data. The constant gap between these two is due to TCS transmitting Q_G fewer indices.

The communication efficiency of IA, adjusted to exclude any sparsification effects, is displayed in Fig. 2b. There, we normalized the total transmitted data of each algorithm with the size of a single gradient transmission. We also include the normalized communication costs for the case without any IA, i.e., conventional routing with multiple unicast transmissions, and IA without any sparsification. Most notably, we observe that CL-SIA and CL-TC-SIA are meeting the same performance as IA without sparsification. This implies that those two algorithms do not suffer from the decreasing efficiency of IA under increasing sparsification ratios. We also see that, while not being able to fully utilize the benefits of IA, SIA and RE-SIA still show much better performance than conventional routing.

Figure 3 shows convergence of the training process in terms of the test accuracy for $K = 28$ clients. The higher communication cost of SIA and RE-SIA directly translates to better learning performance. This is not surprising, as these algorithms send much more information to the PS. However, the reduced sparsification error of RE-SIA only results in a slight



(a) Absolute (b) Relative

Fig. 2. Total transmitted data per global iteration for fixed $Q = 78$ with respect to the number of clients.

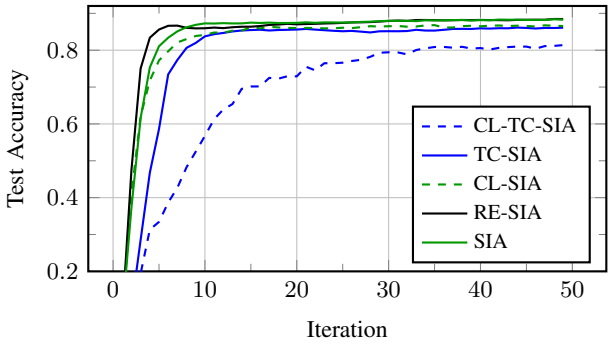


Fig. 3. Test accuracy for a fixed $Q = 78$ and $K = 28$ clients.

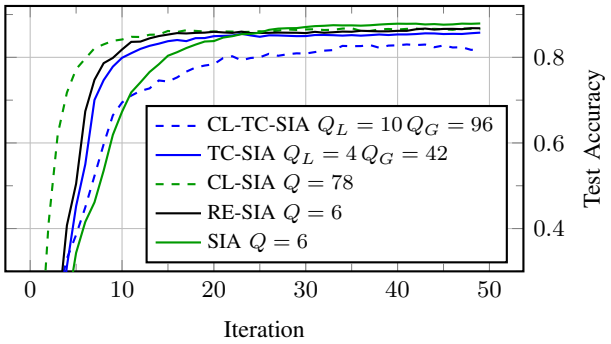


Fig. 4. Test accuracy for $K = 28$ clients under (approximately) equal average bandwidth usage of 98 kbit per global iteration.

edge in convergence speed over SIA. Moreover, convergence of CL-SIA and TC-SIA is only slightly worse than SIA despite their much lower communication cost. We further observe that the convergence speed of CL-TC-SIA is severely impaired. This is likely due to the much smaller effective Q_L, resulting in slower adaption to temporal dynamics.

For a clearer picture of the bandwidth-efficiency, Fig. 4 shows the test accuracy under approximately equal total bandwidth usage. To this end, we consider the same scenario as in Fig. 3 but varied Q such that each algorithm transmits the same amount of data as CL-SIA. For TC-SIA and CL-TC-SIA, we maintain the split $Q_L = 0.1Q$ and $Q_G = 0.9Q$. The result is a slightly higher bandwidth usage for CL-TC-SIA and significantly less for SIA, RE-SIA, and TC-SIA. In terms of training, we see that CL-SIA, RE-SIA, and TC-SIA converge much faster than SIA, with CL-SIA having best performance.

VII. Conclusions

We have considered efficient communication for FL in MH networks. The SoA approach for collecting intermediate FL results in such a system is IA [5], [7], which shows diminishing returns under gradient sparsification [1]. We have developed four novel algorithms to address this issue and evaluated their performance numerically. These results show a distinct advantage of Algorithm 3 over all other methods, both in terms of communication cost and FL performance. However, our results also show a minor gap in final test accuracy over the SoA. This will be subject to further investigation in future work, starting with a rigorous convergence analysis of Algorithm 3.

References

[1] N. Razmi, B. Matthiesen, A. Dekorsy, and P. Popovski, "On-board federated learning for satellite clusters with inter-satellite links," *IEEE Trans. Commun.*, Jan. 2024.

[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Agüera y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artif. Intell. Statist. (AISTATS)*, Fort Lauderdale, FL, Apr. 2017.

[3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[4] B. Matthiesen, N. Razmi, I. Leyva-Mayorga, A. Dekorsy, and P. Popovski, "Federated learning in satellite constellations," *IEEE Netw.*, May 2023.

[5] N. Razmi, B. Matthiesen, A. Dekorsy, and P. Popovski, "Ground-assisted federated learning in LEO satellite constellations," *IEEE Wireless Commun. Lett.*, vol. 11, no. 4, pp. 717–721, Apr. 2022.

[6] ——, "On-board federated learning for dense LEO constellations," in *IEEE Int. Conf. Commun.*, Seoul, Korea, May 2022.

[7] X. Chen, G. Zhu, Y. Deng, and Y. Fang, "Federated learning over multihop wireless networks with in-network aggregation," *IEEE Trans. Wireless Commun.*, vol. 21, no. 6, pp. 4622–4634, Jun. 2022.

[8] A. F. Aji and K. Heafield, "Sparse communication for distributed gradient descent," in *Conf. Empir. Methods Nat. Lang. Process.*, Sep. 2017.

[9] D. Alistarh *et al.*, "The convergence of sparsified gradient methods," in *Adv. Neural Inf. Process. Syst.*, vol. 31, 2018.

[10] E. Ozfatura, K. Ozfatura, and D. Gündüz, "Time-correlated sparsification for communication-efficient federated learning," Jan. 2021. [Online]. Available: http://arxiv.org/abs/2101.08837

[11] A. Sahu *et al.*, "Rethinking gradient sparsification as total error minimization," in *Adv. Neural Inf. Process. Syst.*, vol. 34, 2021.

[12] Y. LeCun, C. Cortes, and C. J. C. Burges. The MNIST database of handwritten digits.