# Trading-off Accuracy and Communication Cost in Federated Learning

Extended Abstract

Mattia Jacopo Villani
King's College London
London, United Kingdom
mattia.villani@kcl.ac.uk

Emanuele Natale
Université Côte d'Azur
Sophia-Antipolis, France
emanuele.natale@univ-cotedazur.fr

Frederik Mallmann-Trenn
King's College London
London, United Kingdom
frederik.mallmann-trenn@kcl.ac.uk

## ABSTRACT

Leveraging the training-by-pruning paradigm introduced by Zhou et al. [NeurIPS'19], Isik et al. [ICLR'23] introduced a federated learning protocol that achieves a 34-fold reduction in communication cost. We achieve a compression improvements of orders of orders of magnitude over the state-of-the-art. The central idea of our framework is to encode the network weights $\vec{w}$ by a the vector of trainable parameters $\vec{p}$, such that $\vec{w} = Q \cdot \vec{p}$ where $Q$ is a carefully-generate sparse random matrix (that remains fixed throughout training). In such framework, the previous work of Zhou et al. [NeurIPS'19] is retrieved when $Q$ is diagonal and $\vec{p}$ has the same dimension of $\vec{w}$.

We instead show that $\vec{p}$ can effectively be chosen much smaller than $\vec{w}$, while retaining the same accuracy at the price of a decrease of the sparsity of $Q$. Since server and clients only need to share $\vec{p}$, such a trade-off leads to a substantial improvement in communication cost. Moreover, we provide theoretical insight into our framework and establish a novel link between training-by-sampling and random convex geometry.

## KEYWORDS

training-by-pruning, communication-efficient federated learning, parameter sharing, networks with random weights, compression, random convex geometry, zonotopes

**ACM Reference Format:**

Mattia Jacopo Villani, Emanuele Natale, and Frederik Mallmann-Trenn. 2025. Trading-off Accuracy and Communication Cost in Federated Learning: Extended Abstract. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 3 pages.

## 1 INTRODUCTION

To enable efficient and secure training on mobile devices, *federated learning* was introduced [2, 3]. In this approach, multiple agents or clients train on separate partitions of the data, periodically sharing learned parameters with a central server. In federated learning, transmitting binary masks instead of exact parameter values not only reduces communication cost but also enhances privacy. Additionally, the sparse network architecture lowers inference costs. Using such approach, [1] achieved high accuracy in training artificial neural networks (ANNs) with a 32-fold communication reduction. They further applied compression techniques that capitalize on patterns of consecutive 1s or 0s, yielding a total communication reduction of 33-34 times. Our paper extends far beyond this 34-fold reduction in communication cost, achieving a 1024-fold total reduction.

*Our Contribution.* We present Zampling (Zonotope Sampling), a new training-by-sampling framework inspired by convex random geometry that achieves small reductions in accuracy for state of the art factors of compression in communication cost in the federated learning setting. Given a an arbitrary neural network architecture, Zampling replaces the model's parameters $\vec{w}$ with a product of a probability vector $\vec{p}$ and a sparse influence matrix $Q$, enabling both training-by-sampling for any model and state of the art compression in parameter communication costs in the federated setting. This lowers the communication cost by several orders of magnitude in the context of federated learning. Moreover, our work is a generalisation of Zhou et. al. [4] Concretely, the main result shows that we can reduce the client communication cost by a factor of 1024 in comparison to the naïve algorithm (and a factor of 32 w.r.t. the state-of-the-art) while witnessing only a 3% point reduction in accuracy.

## 2 METHODOLOGY

We consider the setting of federated learning, where a server and $K$ clients jointly train a neural network model. For any neural network architecture, let $m$ be the total number of parameters of the model and let $n$ be a number of *trainable parameters* with $n \leq m$. Let $\vec{p}(t) \in [0,1]^n$ be the vector of these parameter at time $t$ and we refer to it as the *probability distribution vector*.

Let $Q = (q_{i,j})_{i \leq m, j \leq n}$ be a randomly initialised but non-trainable matrix $\in \mathbb{R}^{m \times n}$ that describes how each trainable parameter (in $\vec{p}$) affects each weight, i.e., $q_{i,j}$ describes how the $j$th trainable parameter influences the $i$th weight. Let $d$ the weight degree (each weight is influenced by $d$ trainable parameters), i.e., the number of non-zero entries per row. The matrix $Q$ does not change over time and will never be sent — we assume that server and clients both have $Q$ which can be realised by sharing the same random seed to generate identical matrices. We assume that the data is distributed IID among the clients.

*Further Notation.* Let $f(x) = \max(\min(x, 1), 0)$ be the ReLU function clipped at 1. Let $D_i$ be the dataset at agent $i$ and $D = \bigcup_i D_i$. Given a weight vector $\vec{w}$, we use $g_{\vec{w}} : X \to Y$ to describe the resulting network (note that server and clients use the same architecture and hence the weight vector fully determines the model). We define the *compression factor* to be $m/n$. In our terminology each *round* has



*Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA.* © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

up to 100 (training) epochs. Clients and server exchange messages at the beginning and end of each round.

*Initialization.* We generate a coefficient matrix $Q \in \mathbb{R}^{m\times n}$. For each $i \le m$ sample a set of $d$ indices $\mathcal{I}_i \in [n]^d$ without replacement. Then generate $Q = (q_{i,j})_{i\in[m],j\in[n]}$ as follows.

$q_{i,j} \sim \begin{cases} N(0,\sigma_i^2) & \text{if } j \in \mathcal{I}_i \\ 0 & \text{otherwise} \end{cases}$, where $\sigma_i^2 = \frac{6}{dn_\ell}$ and $n_\ell$ is the fan-in (number of incoming weights) of the target neuron associated to weight $w_i$ (akin to Kaiming-He initialization).

The initial values of $\vec{p}$ are drawn from an $n$-dimensional uniform distribution $\vec{p}(0) \sim U(0,1)^n$. The initial values of the weights are now calculated by setting $\vec{w}_{init} = \vec{w}(0) = Q\vec{p}(0)$. In Figure 1 we present an illustration of the training protocol.

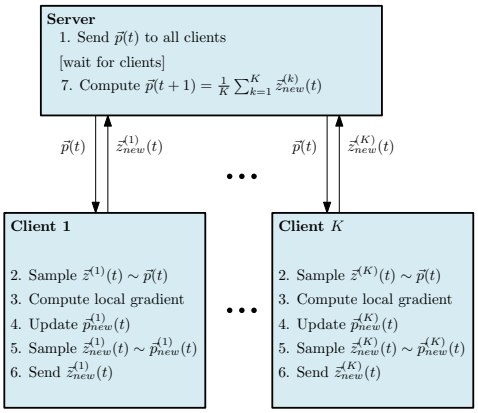


**Figure 1: An illustration of the Federated Zampling algorithm.**

## 3 EXPERIMENTS

The goal of these experiments is to evaluate the performance of Federated Zampling, as we compress the number of parameters, forcing the weight sharing scheme defined by the matrix $Q$. In all our experiments we use the MNIST dataset and use the framework described in Section 2.[1] We run each training round for 100 epochs with early stopping, using 10 epochs of patience and a delta of $10^{-4}$. All our training is run using Adam optimizer, with momentum 0.9 and varying learning rate. Everywhere, we use the standard MNIST data splits with batches of size 128. The model's parameter initialization followed a uniform distribution on $\vec{p}$ and we choose $q_{i,j}$ to be distributed as to recover Kaiming-He initialisation. The architecture we use, MNISTFC, is exactly as the one in Zhou: two hidden layers with three hundred and one hundred neurons respectively.

*Setup.* We ran three simulations with 10 clients and one server. Each client was trained over a total of 100 rounds. The data was partitioned with a random split. In this experiment, we tested the MNISTFC model with training-by-sampling, measuring the accuracy on the expected network. The model was initialized with $n = m/i, i = 1, 8, 32$, where $m = 266610$ and a $d = 10$, with $\vec{p}$ initialized uniformly and learning rate is 0.1, random seed is 1. We compute the mean sampled accuracy at each round, together with the standard deviation out of 100 sampled networks.

*Analysis.* Results are displayed in Figure 2. Bench-marking against $m/n = 1$, we see that our performance receives virtually no loss in performance (.22%) for a 8 fold reduction in parameters in the $m/n = 8$ experiment. When $m/n = 32$, the loss in performance is just 2.55%! In particular, the client savings are 1024x and the server savings 32x in communicaiton cost, versus 33.69x and 1.05x, due to arithmetic compression, in [1]; however, their test accuracy remains close to unchanged, with 0.99 final accuracy.[2]

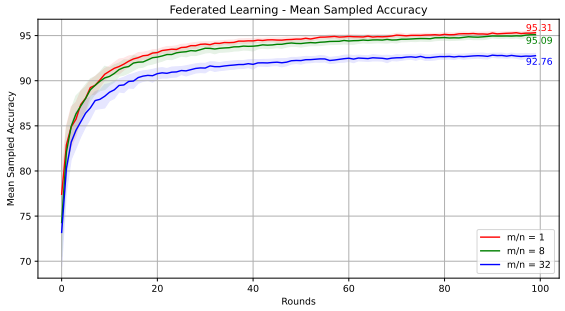


**Figure 2: Results of training Federated Zampling in the federated learning framework with varying levels of $d$.**

## 4 CONCLUSIONS

In this work, we introduced Zampling, a novel training-by-sampling framework inspired by convex random geometry, and demonstrated its effectiveness in federated learning settings. Our method achieves an unprecedented 1024-fold reduction in client communication cost while incurring only a minor accuracy drop compared to the state-of-the-art. By replacing model parameters with a probability vector and a structured influence matrix, we enable large-scale compression while maintaining competitive performance. In a more extensive exposition of this work, we will delve into the underlying mathematical theory of *zonotopes* that provides a theoretical foundation for our framework. We will also release the code and additional empirical results to facilitate further exploration and replication of our findings. Our approach offers a promising direction for efficient, scalable, and communication-aware deep learning in federated environments.

*Acknowledgments.* This work was supported by the EPSRC Grant EP/W005573/1 and EP/S023356/1 (www.safeandtrustedai.org) , and by the 3IA Côte d'Azur Investments in the Future project ANR-19-P3IA-0002.

[1] The evaluation of basic method was run on a machine with GPU RTX3080 with 12GBs of VRAM, with AMD Ryzen Threadripper 3960X 24-Core CPU and RAM 256GBs.

[2] Compared to the 1,933,258 parameter ConvNet architecture in Isik et. al. [1], we use a 266,610 parameter feedforward (about 7 times smaller) architecture. The clusters we have access to were unable to run their architecture, which is why we only have a test accuracy of 0.95 (instead of their 0.99) even without compression. We believe that our results on their architecture would result in even higher accuracies and with higher compression factors due to their model being much more over-parameterised.

## REFERENCES

[1] Berivan Isik, Francesco Pase, Deniz Gunduz, Tsachy Weissman, and Zorzi Michele. 2023. Sparse Random Networks for Communication-Efficient Federated Learning. In *The Eleventh International Conference on Learning Representations*.

[2] Jakub Konečnỳ, Brendan McMahan, and Daniel Ramage. 2015. Federated optimization: Distributed optimization beyond the datacenter. In *8th NIPS Workshop on Optimization for Machine Learning*. https://opt-ml.org/oldopt/opt15/papers.html

[3] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 1273–1282.

[4] Hattie Zhou, Janice Lan, Rosanne Liu, and Jason Yosinski. 2019. Deconstructing Lottery Tickets: Zeros, Signs, and the Supermask. *NIPS* (2019), 11.