POSTER: ON-DEVICE TRAINING WITH LOCAL SPARSITY FOR FL

Xinchi Qiu [*1] **Javier Fernandez-Marques** [*2] **Pedro P.B Gusmao** [1] **Yan Gao** [1] **Titouan Parcollet** [3] **Nicholas D. Lane** [1]

ABSTRACT

In Federated Learning (FL), nodes are orders of magnitude more constrained than traditional server-grade hardware and are often battery powered, severely limiting the sophistication of models that can be trained under this paradigm. While most research has focused on designing better aggregation strategies to improve convergence rates and alleviate the communication costs of FL, fewer efforts have been devoted to accelerating on-device training. Such stage, which repeats hundreds of times (i.e. every round) and can involve thousands of devices, accounts for the majority of the time required to train federated models and, the totality of the energy consumption at the client side. In this work, we present the first study on the unique aspects that arise when introducing sparsity at training time in FL workloads. We then propose `ZeroFL`, a framework that relies on highly sparse operations to accelerate on-device training and achieves +2.3% and +1.5% higher accuracy at 90% and 95% sparsity ratios.

1 INTRODUCTION

In order to adjust the memory and compute footprints of complex ML models to the FL setting, the research community has presented a number of approaches including: the use of distillation (Zhu et al., 2021); federated dropout (Caldas et al., 2019); and, aggregation strategies that enable faster convergence (Li et al., 2018; Reddi et al., 2021). Other optimization techniques such as quantization and sparsity have been considered to reduce communication costs (Amiri et al., 2020) but not to accelerate on-device training.

The use of sparse operations at training time has recently been shown to be an effective technique to accelerate training in centralised settings (Goli & Aamodt, 2020; Raihan & Aamodt, 2020). The resulting models are as good or close to their densely-trained counterparts despite reducing by up to 90% their FLOPs budget and, resulting in an overall up to $3.3\times$ training speedup. Acceleration is achieved by performing sparse convolutions during the forward and/or backward pass, which requires at least one of the operands (i.e. inputs, weights, gradients) to be sufficiently sparse and, software and hardware support for such operations. However, it is unclear how the different FL-specific challenges (i.e. data imbalance, stateless clients, periodic aggregation) will restrict the quality of the global model.

This work considers the challenges and opportunities of inducing high levels of sparsity to accelerate training on-device for FL workloads, and provides the following contributions: (1) A study on the unique aspects that arise when introducing sparsity at training time; (2) `ZeroFL`, a method that alleviates the accuracy degradation when applying a state-of-the-art off-the-shelf sparsification method to the FL domain; (3) a comprehensive analysis on CIFAR-10, FEMNIST (Caldas et al., 2018) and Speech Commands datasets (Warden, 2018) in terms of model performance and communication costs.

2 SPARSE TRAINING FOR FL

The `ZeroFL` formulation builds upon SWAT (Raihan & Aamodt, 2020) and introduces a masking mechanism more suitable to the FL setting. The SWAT framework operates as follows: During each forward pass, the weights are partitioned into active weights and non-active weights by a `top-K` (in magnitude) operator and only the active weights are used. Similarly in the backward pass, the retained layer inputs a_{l-1} are also partitioned into active and non-active by using the same `top-K` procedure. This results in full and dense gradients being used to compute the gradients w.r.t inputs (using sparse active weights) and w.r.t the layer weights (using sparse active retained inputs). The resulting gradients are dense. Therefore, the resulting model remains dense. Due to the data heterogeneity between clients in FL, performance of training using vanilla SWAT degrades significantly compared to centralised training as shown in our poster.

`ZeroFL` also proposes to not send the entire model weights to the central server for aggregation, since only the `top-K` active are required in the forward during inference time. By investigating the position of the `top-K` active weights, we found that the position of majority of the `top-K` weights remain the same throughout the training process. There-

[*]Equal contribution [1]Department of Computer Science and Technology, University of Cambridge, UK [2]Department of Computer Science, University of Oxford, UK [3]Laboratoire Informatique d'Avignon, Avignon Universite. Correspondence to: Xinchi Qiu <xq227@cam.ac.uk>.

fore, `ZeroFL` introduces a masking mechanism by varying the mask ratio that determined how much weights are communicated from client to the server. In this way, the communication cost is saved, and it can also reduce the training noise induced by local data heterogeneity among clients. Let sp be the level of sparsity in the local training and r_{mask} be the mask ratio for communication. Then the client only communicate the top-$(1-sp+r_{mask})$ of weights to the server for aggregation. By reducing the amount of weight communicated, it also reduces noise resulting from the heterogeneous data distribution among different clients.

3 EVALUATION

We conduct our experiments using Flower toolkit (Beutel et al., 2020) on CIFAR10, FEMNIST and Speech Commands datasets. We follow the latent Dirichlet allocation (LDA) partition method for both CIFAR10 and Speech Commands to construct the partition among a pool of 100 clients. FEMNIST, on the other hand, is naturally partitioned by writers' IDs and it has 3597 total clients. We only shows results for Non-IID partition ($\alpha = 1$ in LDA) here among clients to better mimic the cross-device FL training in real scenarios. For both CIFAR10 and Speech Commands, we use Resnet-18 (He et al., 2016). For FEMNIST, we use simple a CNN model.

This work considers accelerating the convolutions involved during forward and backward propagation following a `top-K` sparsity inducing mechanism at the weight level. As a result, the expected sparse pattern would be unstructured, which can only be accelerated if tensors are sufficiently sparse. While sufficiently is mostly hardware-specific, for the target platforms often considered in FL, we expect at least 90% sparsity to be required. Hence, we consider 95% sparsity ratios in our experiments.

As shown in Table 1, it is worth noticing that the masking mechanism with ratio 0.2 performs better than vanilla SWAT, and performance improved with mask ratios from 0 to 0.2, indicating that there exist an optimal interval level. A mask ratio of 0 degenerates the system to the vanilla SWAT, which obtains worse results. It also implies that there is a trade-off between communication cost and performance. By using mask ratio of 0.2, all 3 datasets exhibits higher performance than the baseline, where all weights are communicated from clients to the server.

`ZeroFL` enables us to reduce the performance degradation induced by high level of sparsity in the training. During communication, sparse weight matrices can be transmitted using the Compressed Sparse Row (CSR) format. Such representation requires exactly one integer index for each non-zero weight value in the model. Table 1 shows the level of communication saving with different level of mask ratios,

Dataset	**Sp Level**	**Mask Ratio**	Full Model NIID	Top-K-W. NIID	**File Size**	**Comms. Savings**
CIFAR-10 (100 clients)	95 %	—	74.00±0.74		43.7	1×
		0.0		65.38±0.60	5.9	7.4×
		0.2		**75.54±1.15**	23.0	1.9×
Speech Commands (100 clients)	95 %	—	81.12±0.82		43.7	1×
		0.0		64.79±3.02	5.9	7.4×
		0.2		**81.79±0.33**	23.0	1.9×
FEMNIST (3597 clients)	95 %	—	83.34±0.41		23.0	1×
		0.0		76.79±0.90	1.3	17.7 ×
		0.2		**83.78±0.19**	4.4	5.2 ×

Table 1. Results with ZeroFL on CIFAR10 and SpeechCommands for the non-IID (α=1) setting. FEMNIST is a naturally partitioned dataset. We report the size (in MB) of the artifact to be transmitted to the server for aggregation, which has been compressed following the CSR sparse format representation.

which is calculated as the size ratio between original and compressed models considering both weights and indices in the CSR file. Our findings call for further investigations on the device-oriented optimisation of federated learning to motivate realistic deployments of this training methodology.

REFERENCES

Amiri, M. M., Gunduz, D., Kulkarni, S. R., and Poor, H. V. Federated learning with quantized global model updates, 2020.

Beutel, D. J., Topal, T., Mathur, A., Qiu, X., Parcollet, T., and Lane, N. D. Flower: A friendly federated learning research framework, 2020.

Caldas, S., Duddu, S. M. K., Wu, P., Li, T., Konečnỳ, J., McMahan, H. B., Smith, V., and Talwalkar, A. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

Caldas, S., Konečny, J., McMahan, H. B., and Talwalkar, A. Expanding the reach of federated learning by reducing client resource requirements, 2019.

Goli, N. and Aamodt, T. M. Resprop: Reuse sparsified backpropagation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018.

Raihan, M. A. and Aamodt, T. M. Sparse weight activation training. *arXiv preprint arXiv:2001.01969*, 2020.

Reddi, S. J., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečný, J., Kumar, S., and McMahan, H. B. Adaptive federated optimization. In *ICLR*, 2021.

Warden, P. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*, 2018.

Zhu, Z., Hong, J., and Zhou, J. Data-free knowledge distillation for heterogeneous federated learning, 2021.