AsyncFL: Asynchronous Federated Learning Using Majority Voting with Quantized Model Updates (Student Abstract)

Suji Jang and Hyuk Lim
AI Graduate School
Gwangju Institute of Science and Technology (GIST)
Gwangju 61005, Republic of Korea
sujijang@gm.gist.ac.kr, hlim@gist.ac.kr

Abstract

Federated learning (FL) performs the global model updating in a synchronous manner in that the FL server waits for a specific number of local models from distributed devices before computing and sharing a new global model. We propose asynchronous federated learning (AsyncFL), which allows each client to continuously upload its model based on its capabilities and the FL server to determine when to asynchronously update and broadcast the global model. The asynchronous model aggregation at the FL server is performed by the Boyer–Moore majority voting algorithm for the k-bit quantized weight values. The proposed FL can speed up the convergence of the global model learning early in the FL process and reduce data exchange once the model is converged.

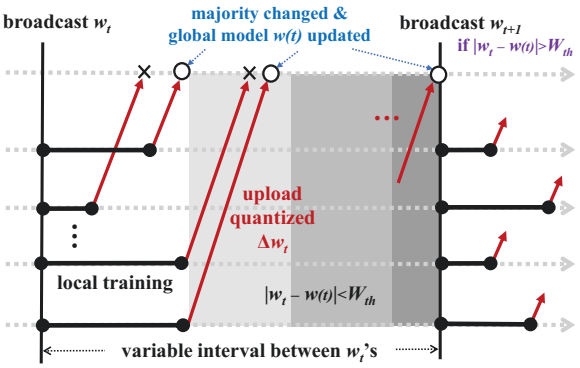

Figure 1: Asynchronous federated learning scheme.

Introduction

Federated learning (FL) is a distributed learning mechanism, in which distributed clients share their local models to construct a global model. The client uses its data to perform local model learning and then sends the model parameters to an FL server. The FL server aggregates the model parameters sent by distributed clients to construct a global model and distributes it back for global model sharing. In (McMahan et al. 2017), the authors proposed an iterative model averaging scheme called federated averaging *(FedAvg)*, which demonstrated the convergence of the distributed learning. In (Yin et al. 2018), the authors proposed a coordinate-wise median operation scheme called *Median-based gradient descent (GD)* for robustness against Byzantine failures. In (Alistarh et al. 2017), the authors proposed a quantization scheme in which each client model is quantized before being uploaded to the server for reducing the communication overhead. In (Xie, Koyejo, and Gupta 2019), the authors proposed an asynchronous federated learning structure, in which clients upload their local parameters to the server individually. The global model is updated by a weighted averaging operation, and the server distributes global parameters regularly regardless of whether the model is converged or not.

Asynchronous Federated Learning

We propose a FL scheme in which the FL server uses a majority voting algorithm to update a global parameter vector using a continuous sequence of clients' local parameter vectors and broadcasts the global parameter vector only when there exists a significant change of elements in the global parameter vector. Figure 1 illustrates a scenario of the proposed FL. Let w_t and w_t^i denote the server parameter vector and the ith client's parameter vectors for $i \in \{1, 2, ..., M\}$, respectively, at a time t. In FL, the client updates its local parameter vector w_t^i using its local dataset. If the client receives a new global parameter vector from the server, it keeps training after replacing its local parameter vector with the new global parameter vector. The client may decide on whether or not to upload its trained local parameter vector or to be configured to upload according to a predetermined policy. At any time the server receives a local parameter vector w_{t+1}^i, it executes a majority voting algorithm. If the discrepancy between the updated global parameter vector and the latest global parameter broadcast exceeds a certain threshold, the server broadcasts the global parameter vector w_{t+1} to all the clients.

The proposed scheme enables each client to upload the difference $\Delta w_{t+1}^i := (w_{t+1}^i - w_t)$ to the server instead of w_{t+1}^i and quantizing the difference into k-bit value using a

simple linear quantization function $Q_k(\cdot)$, which is given by

$$Q_k(x) = \min\{\max\{\text{round}(\frac{2^{k-1} \cdot x}{R}) + 2^{k-1}, 0\}, 2^k - 1\},$$

where R is a constant value sufficiently large for representing Δw's. Note that $Q_k(x)$ quantizes a value x into from zero to $2^k - 1$ for $-R \leq x \leq R$. Once the server receives the quantized difference values $q^i_{t+1} := Q_k(\Delta w^i_{t+1})$'s for the selected clients, it reconstructs the local parameter vectors $\tilde{w}^i_{t+1}$'s.

$$\tilde{w}^i_{t+1} = w_t + Q^{-1}_k(q^i_{t+1}), \quad (1)$$

where $Q^{-1}_k(\cdot)$ is the inverse function of $Q(\cdot)$. Then, the global parameter is obtained by $w_{t+1} = \mathcal{F}(\tilde{\Theta})$, where $\mathcal{F}(\cdot)$ is an aggregation function, and $\tilde{\Theta} \subset \{\tilde{w}^0_{t+1}, \cdots, \tilde{w}^M_{t+1}\}$.

For the asynchronous parameter aggregation, we propose using the Boyer–Moore majority vote algorithm. In the conventional FL, the server waits for a fixed number of local parameter vectors from clients and computes the median value of the vectors for the aggregation. On the contrary, we utilize the Boyer-Moore majority voting algorithm to compute the majority of a continuous sequence of local parameter vectors, which maintains a single counter variable for each element of the vectors for the median computation. The counter is initially set to zero. The corresponding counter is increased by one if an element of a newly arriving weight vector equals the current majority value. Otherwise, the counter will be decreased by one. When the counter is zero, the newly arriving value is selected as the majority value, and the counter is set to one. Whenever an update of the local model is received, the median values of the global model may change. However, the global model distribution is deferred until the discrepancy between the new model and the latest one exceeds a certain threshold. This threshold based algorithm can expedite the global model distribution while the model is being converged and slows down the distribution to reduce the communication overhead once the model is converged. The value of the threshold should be appropriately selected because there exists a trade-off between the convergence speed and the communication overhead.

Experiments and Conclusion

We compared the performance of the proposed *AsyncFL* with that of *FedAvg* and *Median-based GD*. The simulation was performed with the MNIST dataset. The total number of clients is 100. Each client uses a simple CNN classifier, which includes two convolution layers and two fully-connected layers. The local training time is assumed to have a normal distribution, which has μ values from 60 to 6000 seconds and σ values from 18 to 100. For *AsyncFL*, the quantization range R is set to 0.1.

Table 1 shows the simulation results for classification accuracy, convergence time (i.e., 97% performance rising time), and communication overhead for 83 hours. *FedAvg* achieved 97.78 and 97.98 accuracy for C=0.1 and 0.2, respectively. When 20% of clients participated in the global model aggregation (i.e., C=0.2), the convergence was shorter, and the communication overhead was lower even though the server had to wait longer for the model aggregation. *Median-based GD* failed to achieve 97% accuracy. The communication overhead was the same as that of *FedAvg* because both are synchronous methods. The proposed *AsyncFL* shows better performance than both *FedAVG* and *Median GD*. In all the cases, *AsyncFL* achieved the fastest convergence even though there exists a trade-off relationship between the convergence time and communication overhead. Once the learning for the global model is converged, *AsyncFL* can slow down the broadcasting of the global model, resulting in the reduction of communication overhead, while the synchronous methods keep broadcasting at almost the same frequency.

		Accuracy	Conv	Comm
FedAvg	C=0.1	97.78	128,856	1,728
	C=0.2	97.98	119,257	937
Median GD	C=0.1	77.37	-	1,704
	C=0.2	90.06	-	937
AsyncFL	W_{th}=1, k=12	98.05	57,276	1,119
	W_{th}=2, k=12	98.15	76,684	914
	W_{th}=1, k=16	97.28	105,664	1,285
	W_{th}=2, k=16	97.96	97,790	917

Table 1: Experiment results for accuracy (%), convergence time (97% rising time in seconds), and communication overhead (Kb/parameter) with MNIST dataset.

In conclusion, we have proposed the AsyncFL scheme using the Boyer-Moore majority vote algorithm to enable a continuous global model aggregation and a threshold based global model broadcasting algorithm to achieve fast convergence and communication overhead reduction.

Acknowledgments

This work was supported by IITP grant funded by the Korea government (MSIT) (No. 2021-0-00379, Privacy Risk Analysis and Response Technology Development for AI Systems, and No. 2019-0-01842, AI Graduate School Program (GIST))

References

Alistarh, D.; Grubic, D.; Li, J.; Tomioka, R.; and Vojnovic, M. 2017. QSGD: Communication-efficient SGD via gradient quantization and encoding. *Advances in Neural Information Processing Systems*, 30: 1709–1720.

McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, 1273–1282. PMLR.

Xie, C.; Koyejo, S.; and Gupta, I. 2019. Asynchronous federated optimization. *arXiv preprint arXiv:1903.03934*.

Yin, D.; Chen, Y.; Kannan, R.; and Bartlett, P. 2018. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*, 5650–5659. PMLR.