Poster: Accelerate Cross-Device Federated Learning With Semi-Reliable Model Multicast Over The Air

Yunzhi Lin Shouxi Luo
Southwest Jiaotong University, Chengdu, China

***Abstract*—To achieve efficient model multicast for cross-device Federated Learning (FL) over shared wireless channels, we propose SRMP, a transport protocol that performs semi-reliable model multicast over the air by leveraging existing PHY-aided wireless multicast techniques. The preliminary study shows that, with novel designs, SRMP could reduce the communication time involved in each round of training significantly.**

I. Introduction

Because of its ability of privacy-preserving, cross-device Federated Learning (FL) has been widely employed by many of today's smartphone applications for various purposes in production [1], and is predicted to have critical usages in emerging scenarios like *unmanned aerial vehicles* and *self-driving cars* in the near future [2]. As Figure 1 shows, to conduct a round of the iterative FL training in these scenarios, each device (e.g., vehicles, cars) first reads the current model along with the training configurations from the central parameter server (PS) (i.e., Step 1), then performs local training and reports the model updates back to the PS (i.e., Step 2); by aggregating these model updates, the PS finally obtains the new global model and starts the next round of training [1], [2]. Obviously, in this process, the download of the model would take a non-trivial portion of, or even dominate, the time cost of the entire training [1]. Thus, optimizing the model delivery over shared wireless channels is critical for optimizing the efficiency of these cross-device FL training tasks.

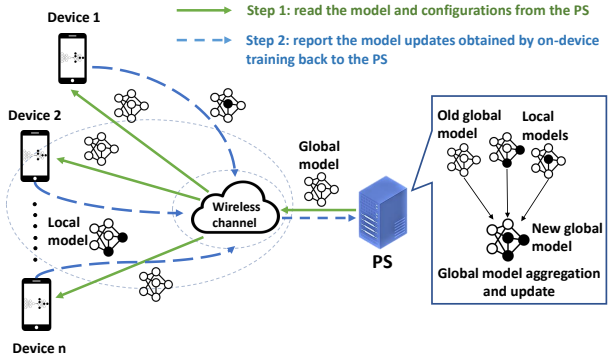


Fig. 1: Cross-device federated learning at the wireless edge, in which devices like unmanned aerial vehicles, self-driving cars train a shared model collectively over a shared wireless channel, with the assistance of a central parameter server (PS).

Corresponding author: Shouxi Luo (e-mail: sxluo@swjtu.edu.cn). This work was supported in part by the NSFC under Grant 62002300.

978-1-6654-4131-5/21/$31.00 ©2021 IEEE

The delivery of the model is a typical one-to-many transmission task. In view of the broadcast nature of wireless channels, a straightforward optimization is to perform the delivery with abundant existing PHY-aided techniques (e.g., DirCast) that could conduct efficient Layer 2 multicast/broadcast over various wireless networks at the last hop [3]. Unfortunately, implementing such a design in practice faces two challenges. On one side, existing PHY-aided Layer 2 multicast techniques provide either totally unreliable or reliable delivery service, mismatching with the fact that FL tasks in practice generally tolerate loss-bounded model transmissions [4].[1] Indeed, as this poster will show, by exploring this type of tolerance, there is a large room for the optimization of multicast performance. On the other, all these techniques are PHY-specified and work at Layer 2, thus hard to use for widespread FL applications.

For these problems, a fundamental solution is to provide efficient yet reliability-controlled one-to-many data delivery services upon existing PHY-aided wireless multicast techniques [1], for wireless-channel-shared FL training. In this poster, we propose our case design of SRMP (Semi-Reliable Multicast Protocol), and focus on its key algorithm designs:

1) A novel congestion control algorithm that could tolerate non-congestion based packet loss; and
2) A selective retransmission algorithm that could significantly reduce the number of total retransmissions by leveraging the FL task's tolerances of bounded loss.

II. Proposed Protocol

At the high level, SRMP enables each FL task to specify its acceptable loss rate (alr) along with the model to deliver. Then, at the low level, these model values are encapsulated in specific UDP packets to send; and PHY-aided Layer 2 multicast techniques will be employed when they reach the last hop. On getting these packets, SRMP receivers generate acknowledgment (ACK) selectively, based on which, the SRMP sender estimates lost packets and available bandwidth. To reduce the number of ACKs, SRMP receivers are selected to generate ACKs in a round-robin.

To support various PHY-aided multicast techniques, SRMP assumes that the underlying wireless channels only provide unreliable multicast, and guarantees the semi-reliability of model values efficiently with novel congestion control and retransmission algorithm designs. In practice, the multicast

[1] If a parameter's value is lost, the worker/device will use the default value of 0, analogous to performing a *Dropout* augmentation on the model [5].

rate at Layer 2 could be fixed or self-adaptive, depending on the employed PHY techniques [3].

Congestion Control. Similar to the congestion control of TCP, SRMP maintains its sending rate with a sliding congestion control window (cwnd). In SRMP, the loss of a packet can be caused either by the unstable wireless channel at the last hop, or by any link congestion along its journey. Let γ be the current loss rate of the wireless channel observed by the device. SRMP receivers piggyback their γ values and indexes of lost model values along with ACKs, based on which, the sender calculates their real loss rates caused by link congestion and adjusts the cwnd. Currently, SRMP employs a preliminary *additive increase multiplicative decrease* design like that of TCP Reno for cwnd. Notably, to accelerate the convergence of congestion control, besides round-robin, a receiver will immediately generate specific ACKs to infer the sender once it suffers the events of timeout or serious packet loss.

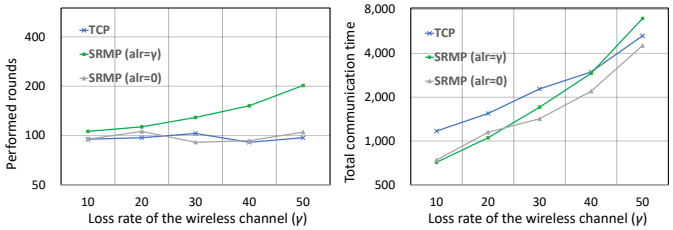
Restransmission. As devices in FL can tolerate bounded packet loss, the SRMP sender retransmits packets only when the requirement of alr is not satisfied at some devices. In case several devices call for retransmissions, to be bandwidth-efficient, SRMP would perform the retransmissions with joint optimizations. A simple yet efficient design is to let SRMP retransmit packets that get lost by most of the devices at first. We will extend SRMP to support advanced designs like using coding techniques (e.g., network coding) to merging different diverse retransmissions requests in future work.

III. Preliminary Evaluation and Future Work

Simulation System. To verify the advantage of SRMP, we build a simulation system upon Mininet and Pytorch to simulate the behavior of cross-device FL tasks over shared channels. Basically, in this system, multiple Pytorch-based workers train a model collectively (see Figure 1) over a shared wireless channel simulated by Mininet. At its core, the half-duplex and broadcast natures of wireless channels are simulated using the Linux semaphores and the broadcast ability provided by open vswitch, respectively. As for the bandwidth and loss ratio of a link, virtual links in Mininet already support these features.

Case Study. We consider a cross-device FL task in which 5 workers/devices train the MobileNet model using the dataset of CIFAR10. We stop the training when its accuracy reaches 30%. The training batch size is set to 32 and model values are split into 1KB chunks to suit the payload size of UDP. To highlight the advantage of SRMP, the bandwidth and latency of the wireless channel in tests are set to 500kbps and 0ms, respectively, while γ, the loss rate of the channel, ranges from 10% to 50%. By default, in each round of training, workers download the model with SRMP, then upload their gradients with its unicast variant. The timeout configuration of SRMP is set to 2.5s. For SRMP, we consider two simple use cases, in which their alrs are configured to 0 and γ, respectively.

Besides, we also consider a specific instance akin to the case in which both the model download and gradient upload are carried out with TCP. Figure 2b and Figure 2a show the time costs and performed rounds of TCP, SRMP ($alr = \gamma$) and SRMP ($alr = 0$) when the wireless channel's loss rate ranges from 10% to 50%.



(a) Impact of γ on the performed rounds

(b) Impact of γ on the total communication time

Fig. 2: The increase of the packet loss rate γ slows the convergence speed of SRMP (alr=γ) down, as more rounds are performed as Figure 2a shows. However, provided the value of γ does not exceed the threshold (e.g., 0.2 in this test), SRMP (alr=γ) would still obtain the smallest total communication time (see Figure 2b), indicating that the communication time of each round of training is reduced significantly.

As Figure 2a shows, with the growth of γ, the performed rounds of both TCP and SRMP (alr=0) based training keep nearly consistent. This is reasonable since the lost packets would be re-transmitted by them. However, these re-transmutation operations result in more computation time as Figure 2b shows. In contrast, despite it takes more training rounds for SRMP (alr=γ) to reach the same accuracy due to the loss of packets, the total communication time might still be reduced because the heavy re-transmission operations are simplified or even eliminated (e.g., the case of SRMP (alr=γ). For instance, in the test case shown in Figure 2b, SRMP (alr=γ) is the best when the packet loss rate is no more than 20%. When γ is 20%, SRMP (alr=γ)'s communication times in total and each round of training, are about 8.2% and 14.2% less than those of SRMP (alr=0), respectively.

Future Work. Indeed, there is a trade-off between the costs of increased training rounds and the benefits of reduced per-round communication times. Configuring the value of alr respecting the characteristics of both the trained FL model and underlying network environments to explore the best performance, is one of the most important future works of SRMP. Besides, exploring the design of SRMP in more detail and extending it to support cross-layer optimization will be considered.

References

[1] P. Kairouz *et al.*, "Advances and open problems in federated learning," *CoRR*, vol. abs/1912.04977, 2019.

[2] D. Gündüz *et al.*, "Machine learning in the air," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2184–2199, 2019.

[3] J.-M. Vella *et al.*, "A survey of multicasting over wireless access networks," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 2, pp. 718–753, 2013.

[4] S. Luo *et al.*, "Selective coflow completion for time-sensitive distributed applications with poco," in *49th ICPP*. ACM, 2020.

[5] S. Li *et al.*, "Taming unbalanced training workloads in deep learning with partial collective operations," in *25th PPoPP*. ACM, 2020, pp. 45–61.