QSFL: A Two-Level Uplink Communication Optimization Framework for Federated Learning

ICML 2022

Liping Yi, Gang Wang, Xiaoguang Liu
Nankai-Baidu Joint Lab, Nankai University

Parallel and Distributed
Software Technology Lab

Nankai - Baidu
Joint Laboratory

Background

Federated Learning:

- Federated learning (FL) [1] is a popular paradigm for collaboratively training a share model with privacy protection. It allows clients only exchange parameter/gradient updates with the server, and data is always stored locally.

Challenges:

- The typical FL algorithm such as FedAvg [1] faces the following challenges: privacy, model performance, robustness to adversarial attacks, fairness, **communication cost**, etc.

  Uplink communication cost is influenced by:

  - Thousands of devices collaboratively train one GB-level model, uplink transmission can reach TB-level;
  - Expensive and unreliable wireless connections between edge devices and the server;
  - Large-scale deep networks often have lots of redundant parameters;
  - For internet connection, uploading speed is often slower than downloading speed, so compressing client-to-server (uplink) transmission traffic will yield greater benefits.

[1] McMahan, Brendan, et al. "Communication-efficient learning of deep networks from decentralized data." AISTATS, 2017.

Motivation

Existing Methods:

- Delay communication: increasing local iterations;
- Model sparsification: replacing full-precision parameters with sparse representations;
- Model quantization: representing parameters with fewer bits;
- Parameter encoding: encoding parameters into sketches or other more compact forms;
- Client sampling: sampling partial important clients to join in FL.

Either of them have no theoretical convergence guarantee or have limited compression ratios.

Our Insights:

- Inspired by *client sampling* and *model sparsification*, we propose a novel FL framework QSFL to optimize FL uplink communication at two levels:
  - **Client-level**: Qualification Judgment (QJ) algorithm samples high-qualification clients to upload models;
  - **Model-level**: Sparse Cyclic Sliding Segment (SCSS) algorithm further compresses transmitted models.

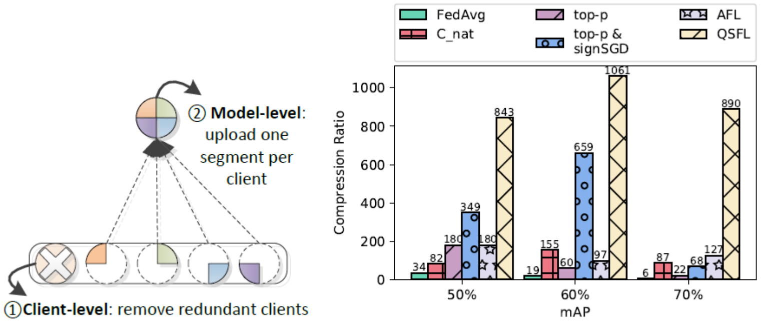


Figure 1: Left: insights of QSFL. Right: QSFL outperforms alternatives in the compression ratio under 50%, 60% and 70% target mAP on FEMINIST dataset respectively.

Methodology

Overview of QSFL framework:

- **Client-level**: Qualification Judgment (QJ) algorithm samples high-qualification clients to upload models;
- **Model-level**: Sparse Cyclic Sliding Segment (SCSS) algorithm further compresses transmitted models.

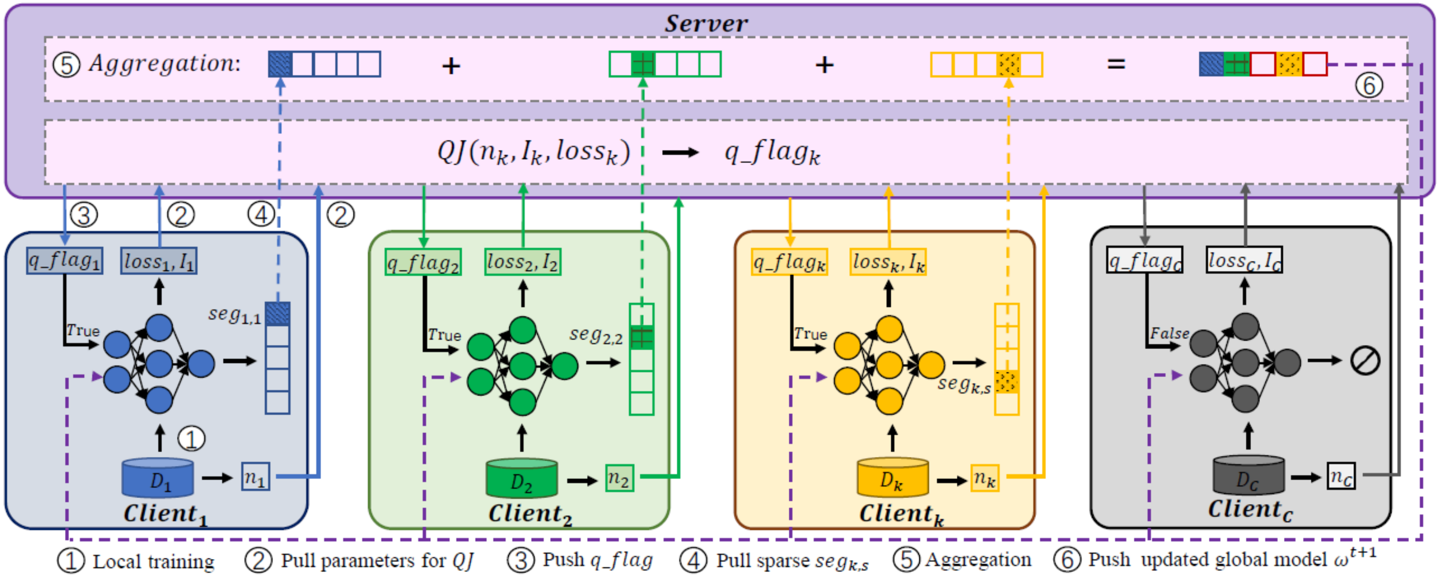


Figure 2: A complete workflow of QSFL framework in one round. ①: clients train local models on local datasets, ②: clients compute and upload the three key parameters for qualification judgment, ③: the server calculates and sends qualification flags back to clients, ④: clients with uploading qualification (flag is set to **true**) select one particular sparse segment to the server, ⑤: the server aggregates segments with same id, ⑥: the server broadcasts the updated global model to all clients.

Methodology | Qualification Judgement (QJ)

Client-level: Qualification Judgement (QJ) Algorithm

$$\begin{aligned} q_k &= \beta \cdot contribution_k + (1-\beta)\cdot relevance_k \\ &= \beta \cdot \frac{loss_k/n_k}{\sum_{i=1}^{C} loss_i/n_i} \\ &+ (1-\beta)\cdot \frac{\sum_{j=1}^{|\omega_k^l|} \mathbb{I}(sgn(\omega_{k,j}^l) == sgn(\omega_j^g))}{|\omega_k^l|}, \end{aligned}$$

- **Computing Contribution.** We measure each client's contribution through normalized loss_k/n_k. The larger mean loss makes the training process gain more by gradient descent optimizer. FL training aims to reduce the mean loss of all clients, so choosing clients with larger loss can *speed up convergence.*

- **Computing Relevance.** We compute the relevance between the received global model and trained local models by (number of parameters with the same signs on the same coordinates) / (total number of model parameters). Higher relevance indicates the local model's gradient direction tends more to the global model. Hence, sampling high-relevance clients can *ensure convergence*.

Methodology | Sparse Cyclic Sliding Segment (SCSS)

Model-level: Sparse Cyclic Sliding Segment (SCSS) Algorithm

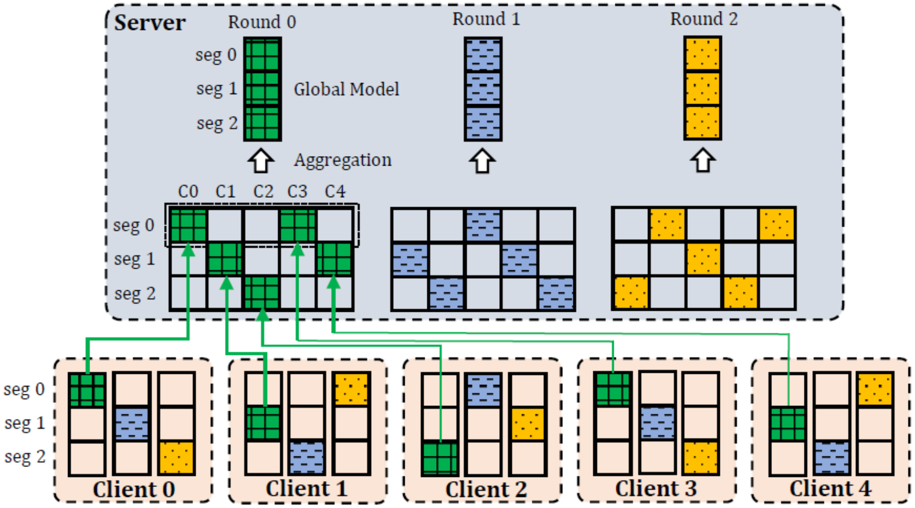
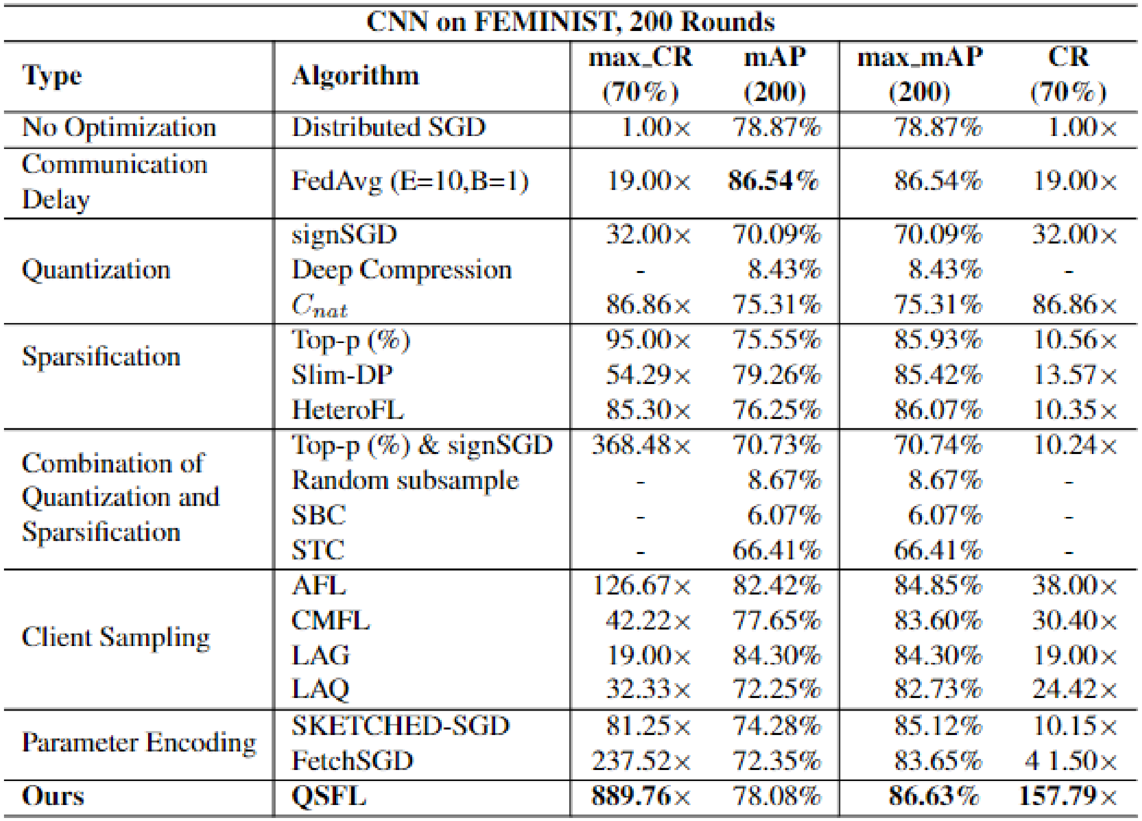


Figure 3: A toy example of SCSS.

- **Cyclic Sliding Segment.** Clients evenly divide local models into NS segments, each client only upload *one* segment in one round. The id of the uploaded segment is controlled by (client_id + round) % NS, i.e., achieving cyclic sliding segment.
- **Sparse Trick.** We further apply top-p(%) sparsification as a trick to compress each segment.
- **Server Aggregation.** The server first decodes received sparse segments, then averages segments with the same seg_id, finally splices NS averaged segments to reassemble the updated global model.

Experiments



CNN on FEMINIST, 200 Rounds

Type	Algorithm	max_CR	mAP	max_mAP	CR
		(70%)	(200)	(200)	(70%)
No Optimization	Distributed SGD	1.00×	78.87%	78.87%	1.00×
Communication Delay	FedAvg (E=10,B=1)	19.00×	**86.54%**	86.54%	19.00×
Quantization	signSGD	32.00×	70.09%	70.09%	32.00×
	Deep Compression	-	8.43%	8.43%	-
	C_{nat}	86.86×	75.31%	75.31%	86.86×
Sparsification	Top-p (%)	95.00×	75.55%	85.93%	10.56×
	Slim-DP	54.29×	79.26%	85.42%	13.57×
	HeteroFL	85.30×	76.25%	86.07%	10.35×
Combination of Quantization and Sparsification	Top-p (%) & signSGD	368.48×	70.73%	70.74%	10.24×
	Random subsample	-	8.67%	8.67%	-
	SBC	-	6.07%	6.07%	-
	STC	-	66.41%	66.41%	-
Client Sampling	AFL	126.67×	82.42%	84.85%	38.00×
	CMFL	42.22×	77.65%	83.60%	30.40×
	LAG	19.00×	84.30%	84.30%	19.00×
	LAQ	32.33×	72.25%	82.73%	24.42×
Parameter Encoding	SKETCHED-SGD	81.25×	74.28%	85.12%	10.15×
	FetchSGD	237.52×	72.35%	83.65%	4 1.50×
Ours	**QSFL**	**889.76×**	78.08%	**86.63%**	**157.79×**

- QSFL achieves the highest compression ratio (889.76×) and also has a similar mAP with Distributed SGD.
- QSFL achieves the highest mAP (86.63%) while also maintaining the highest compression ratio (157.79×).

Conclusions

- QSFL optimizes FL uplink communication overhead from two levels: a) client-level, rejecting redundant clients uploading model updates, and b) model-level, compressing uploaded models to unique segments.

- Theoretical proof and extensive experiments verify that QSFL can effectively reduce uplink communication costs with marginal model accuracy degradation.

Thanks for your listening!

Email: yiliping@nbjl.nankai.edu.cn