IntelliSys Lab

HeteroFL: Computation and Communication Efficient Federated Learning for Heterogeneous Clients

Enmao Diao, et al

Published in ICLR, 2021

LSU

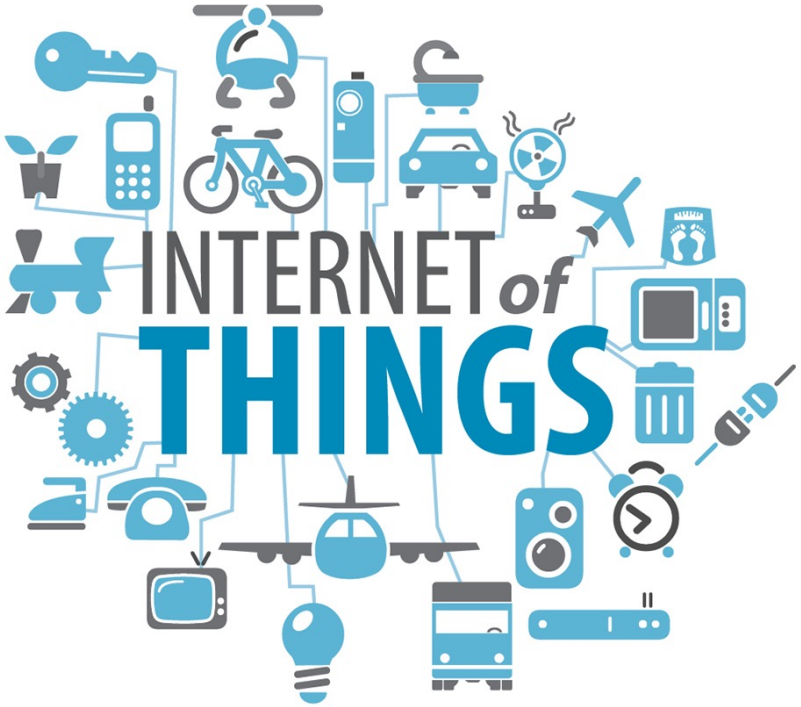
Outline

- Motivation
- HeteroFL
- Evaluation
- Discussion

Motivation

IntelliSys Lab

Mobile devices and the IoT devices are becoming the primary computing resource

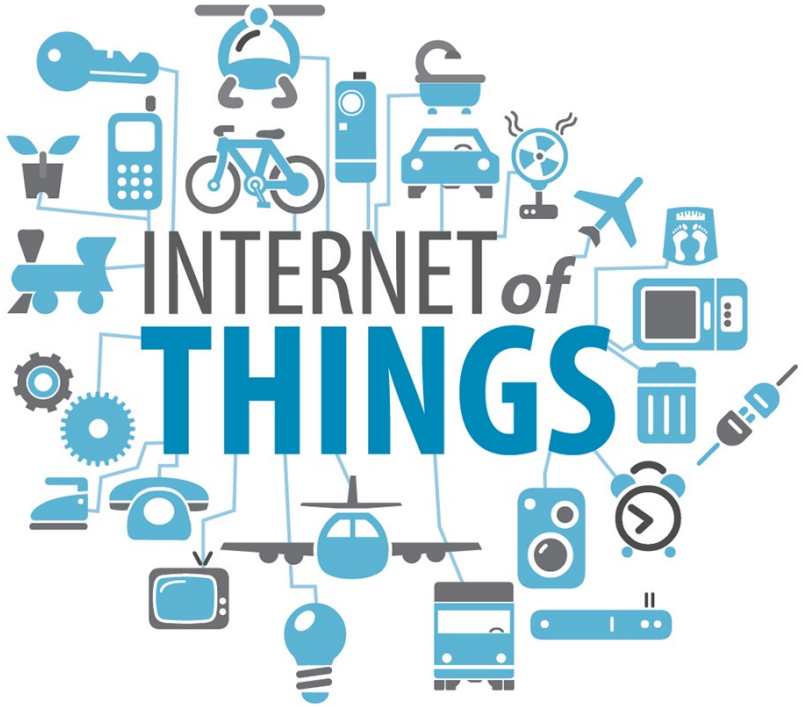


a significant amount of data

LSU

Motivation

Mobile devices and the IoT devices are becoming the primary computing resource



→ store data and train models locally

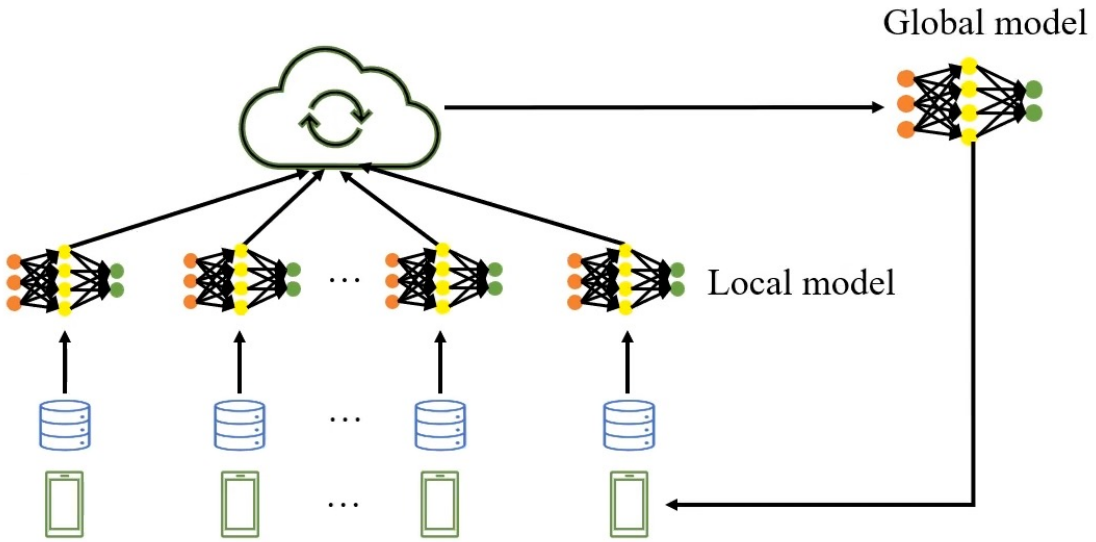
↓

Federated learning

Motivation

IntelliSys Lab

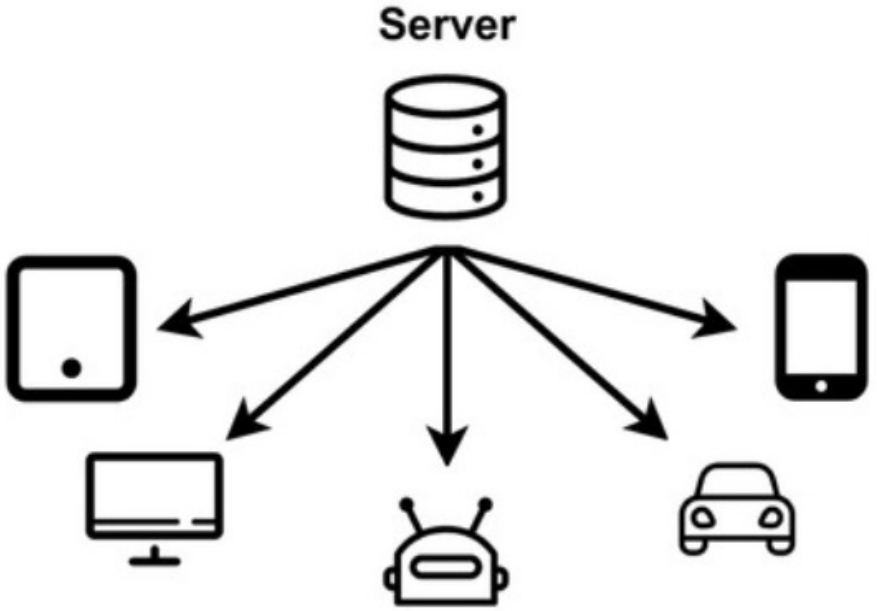
Existing federated learning: one global model architecture for all client devices



Motivation

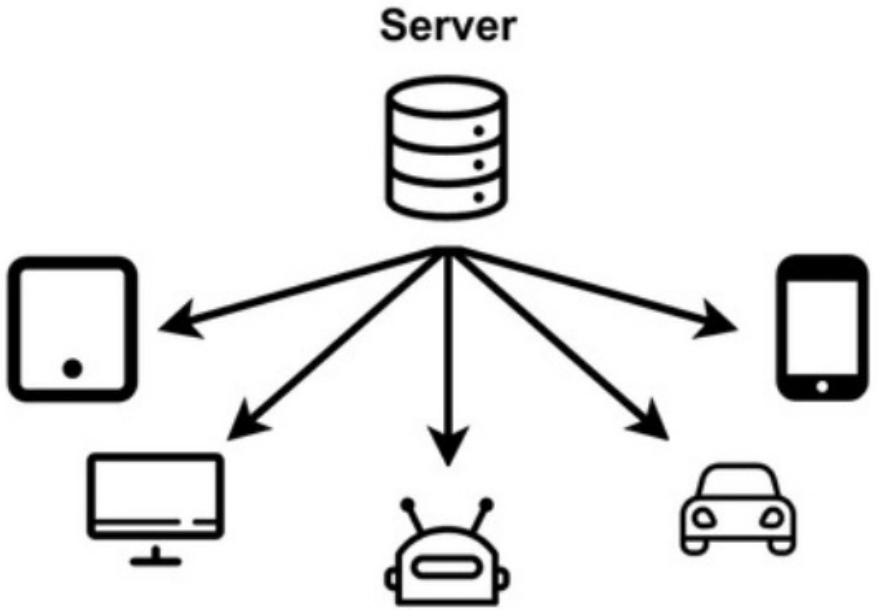
The computation and communication capabilities of each client devices may vary significantly and dynamically

heterogeneous



Motivation

Local models must share the same architecture as the global model?



HeteroFL: train heterogeneous local models with varying computation complexities and still produce a single global inference model

Outline

- Motivation
- HeteroFL
- Evaluation
- Discussion

HeteroFL

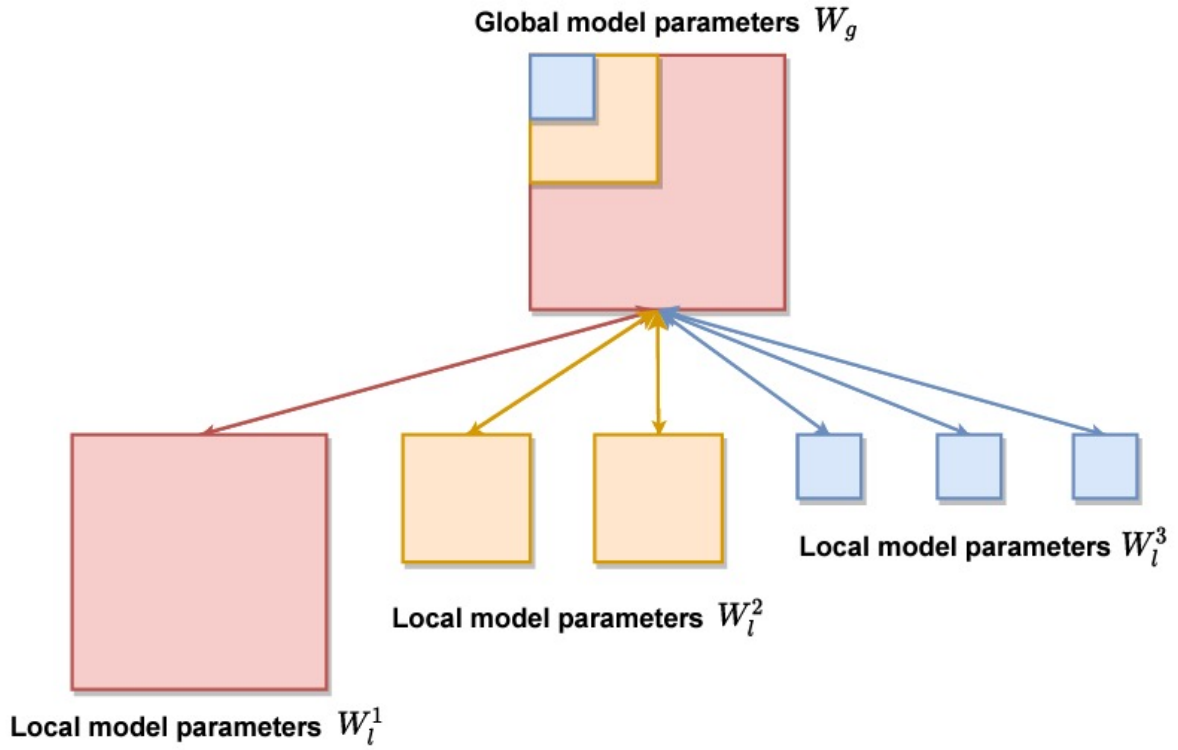


Figure 1: Global model parameters W_g are distributed to $m = 6$ local clients with $p = 3$ computation complexity levels.

HeteroFL

To reduce the computation and communication complexity, consider local models to have similar architecture but can shrink their complexity within the same model class

To simplify global aggregation and local update, consider local model parameters to be a subset of global model parameters

$$W_i^{t+1} \subseteq W_g^t$$

To reduce the size of deep neural network, the authors choose to vary the width of hidden channels, while the local and global model architectures are also within the same model class, which stabilizes global model aggregation

HeteroFL

Select subsets of global model parameters W_l for a single hidden layer parameterized by $W_g \in \mathbf{R}^{d_g \times k_g}$. It is possible to have multiple computation complexity levels

$$W_l^p \subset W_l^{p-1} \cdots \subset W_l^1$$

Let r be the hidden channel shrinkage ratio such that $d_l^p = r^{p-1} d_g$ and $k_l^p = r^{p-1} k_g$ where d and k are the output and input channel size of the layer.

The size of local model parameters is $|W_l^p| = r^{2(p-1)} |W_g|$ and the model shrinkage ratio $R = \frac{|W_l^p|}{|W_g|} = r^{2(p-1)}$.

Global aggregation:

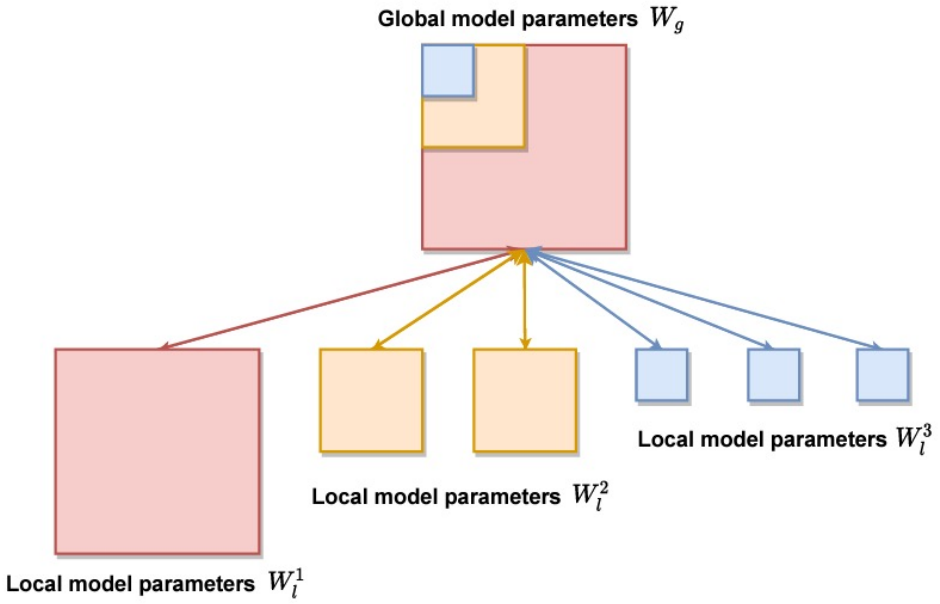
$$W_l^p = \frac{1}{m}\sum_{i=1}^{m} W_i^p, \quad W_l^{p-1} \setminus W_l^p = \frac{1}{m-m_p}\sum_{i=1}^{m-m_p} W_i^{p-1} \setminus W_i^p, \ldots \tag{1}$$

$$W_l^1 \setminus W_l^2 = \frac{1}{m-m_{2:p}}\sum_{i=1}^{m-m_{2:p}} W_i^1 \setminus W_i^2 \tag{2}$$

$$W_g = W_l^1 = W_l^p \cup (W_l^{p-1} \setminus W_l^p) \cup \cdots \cup (W_l^1 \setminus W_l^2) \tag{3}$$

$W_l^{p-1} \setminus W_l^p$ means the set of elements included in W_l^{p-1} but excluded in W_l^p

HeteroFL

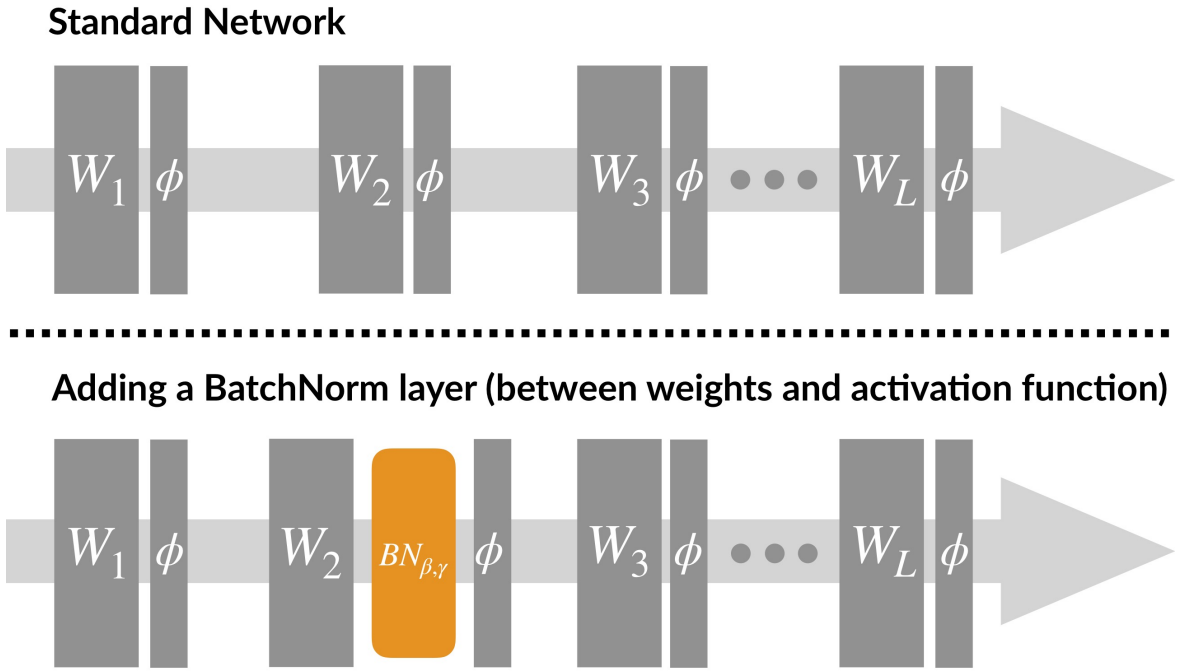


$$W_l^p = \frac{1}{m}\sum_{i=1}^{m} W_i^p, \quad W_l^{p-1} \setminus W_l^p = \frac{1}{m-m_p}\sum_{i=1}^{m-m_p} W_i^{p-1} \setminus W_i^p, \ldots \tag{1}$$

$$W_l^1 \setminus W_l^2 = \frac{1}{m-m_{2:p}}\sum_{i=1}^{m-m_{2:p}} W_i^1 \setminus W_i^2 \tag{2}$$

$$W_g = W_l^1 = W_l^p \cup (W_l^{p-1} \setminus W_l^p) \cup \cdots \cup (W_l^1 \setminus W_l^2) \tag{3}$$

Static Batch Normalization



Latest deep learning models usually adopt Batch Normalization (BN) to facilitate and stabilize optimization. However, classical FedAvg and most recent works avoid BN. A major concern of BN is that it requires running estimates of representations at every hidden layer.

Static Batch Normalization

Static Batch Normaliztion (sBN):
During the training phase, sBN does not track running estimates and simply normalize batch data.

After the training process finishes, the server sequentially query local clients and update global BN statistics.

Scaler

We need to optimize local models for multiple epochs and local model parameters at different computation complexity levels will digress to various scales.

To directly use the model during the inference phase without scaling, inverted dropout with dropout rate q scales representations with $\frac{1}{1-q}$ during the training phase. In practice, dropout is usually attached after the activation layer.

The authors append a Scaler module right after the parametric layer and before the sBN and activation layers. The Scaler module scales representations by $\frac{1}{r^{p-1}}$ during the training phase.

Scaler

A typical linear hidden layer used in our HeteroFL framework can be formulated as

$$y = \phi(\text{sBN}(\text{Scaler}(X_m W_m^p + b_m^p)))$$

Outline

- Motivation
- HeteroFL
- Evaluation
- Discussion

Experimental Setup

- Models
  - CNN for MNIST
  - Preactivated ResNet for CIFAR10
  - Transformer for WikiText2

Experimental Setup

- Data
  - For IID data partition, assign the same number of data examples for each client
  - For balanced non-IID data partition, assume that the label distribution is skewed, where clients will only have examples at most from two classes and the number of examples per class is balanced.

Experimental Setup

- Computation complexity levels
  - $\{a, b, c, d, e\}$ with complexity levels 0.5, 0.25, 0.125, and 0.0625
- Fix
  - experiments with a fixed assignment of computation complexity levels
- Dynamic
  - local clients uniformly sampling computation complexity levels at each communication round

Evaluation

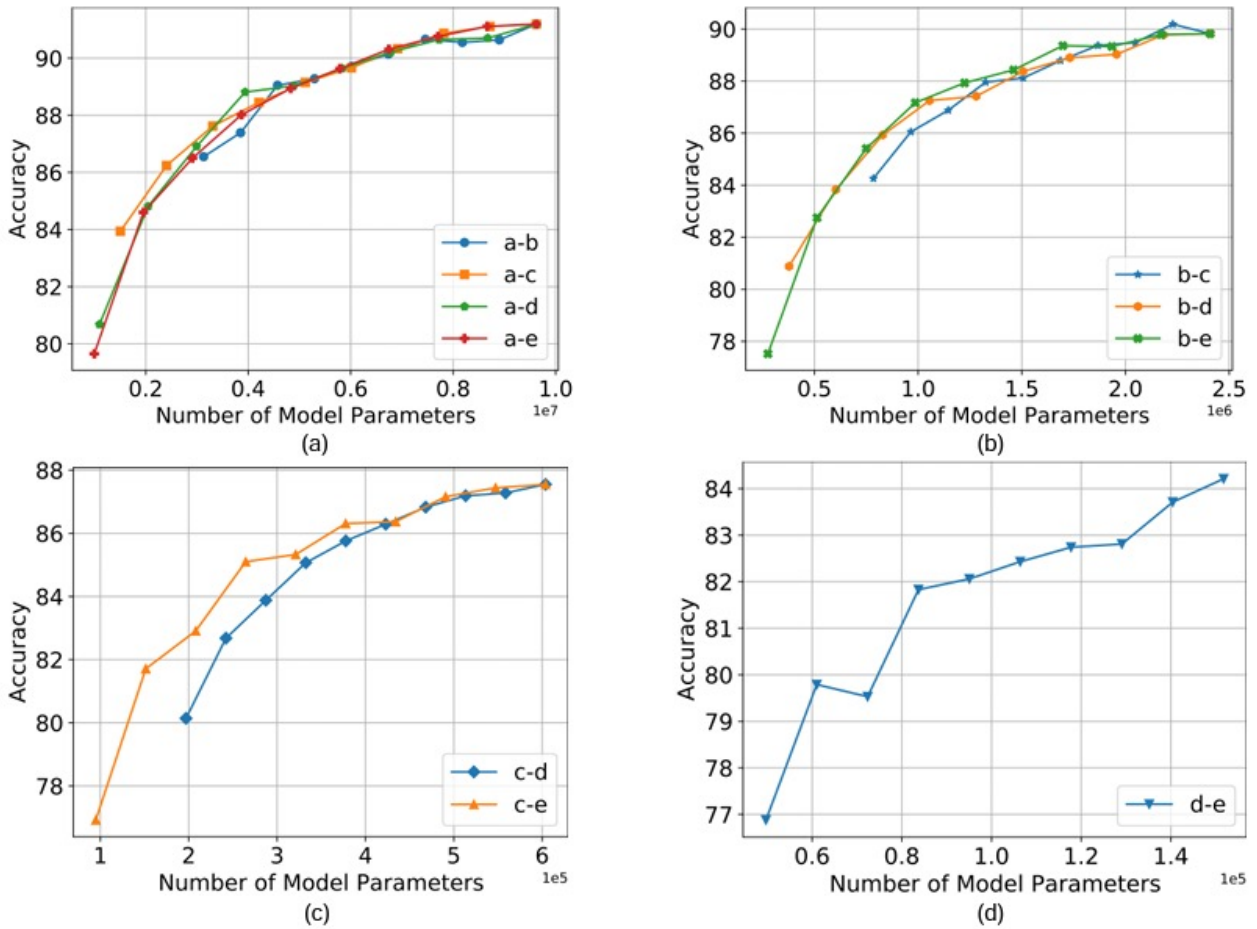


Figure 2: Interpolation experimental results for CIFAR10 (IID) dataset between global model complexity ((a) a, (b) b, (c) c, (d) d) and various smaller model complexities.

Evaluation

IntelliSys Lab

Model	Ratio	Parameters	FLOPs	Space (MB)	Accuracy		
					IID	Non-IID	
						Local	Global
a	1.00	1.6 M	80.5 M	5.94	99.53	99.85	98.92
a-e	0.50	782 K	40.5 M	2.98	99.46	99.89	98.96
a-b-c-d-e	0.27	416 K	21.6 M	1.59	99.46	99.85	98.29
b	1.00	391 K	20.5 M	1.49	99.53	99.87	99.10
b-e	0.51	199 K	10.4 M	0.76	99.51	99.67	98.51
b-c-d-e	0.33	131 K	6.9 M	0.50	99.52	99.88	98.99
c	1.00	99 K	5.3 M	0.38	99.35	99.56	96.34
c-e	0.53	53 K	2.9 M	0.20	99.39	99.79	97.27
c-d-e	0.44	44 K	2.4 M	0.17	99.31	99.76	97.85
d	1.00	25 K	1.4 M	0.10	99.17	99.86	97.86
d-e	0.63	16 K	909 K	0.06	99.19	99.63	97.70
e	1.00	7 K	400 K	0.03	98.66	99.07	92.84
Standalone (Liang et al., 2020)	1.00	633 K	1.3 M	2.42	86.24	98.72	30.41
FedAvg (Liang et al., 2020)	1.00	633 K	1.3 M	2.42	97.93	98.20	98.20
LG-FedAvg (Liang et al., 2020)	1.00	633 K	1.3 M	2.42	97.93	98.54	98.17

Table 1: Results of combination of various computation complexity levels for MNIST dataset. Full results can be found in Table 7.

Evaluation

IntelliSys Lab

Model	Ratio	Parameters	FLOPs	Space (MB)	Accuracy		
					IID	Non-IID	
						Local	Global
a	1.00	9.6 M	330.2 M	36.71	91.19	92.38	56.88
a-e	0.50	4.8 M	165.9 M	18.43	90.29	92.10	59.11
a-b-c-d-e	0.27	2.6 M	88.4 M	9.78	88.83	92.49	61.64
b	1.00	2.4 M	83.3 M	9.19	89.82	93.83	55.45
b-e	0.51	1.2 M	42.4 M	4.67	89.10	90.68	59.81
b-c-d-e	0.33	801 K	27.9 M	3.05	87.92	91.90	59.10
c	1.00	604 K	21.2 M	2.30	87.55	91.09	55.12
c-e	0.53	321 K	11.3 M	1.22	86.88	91.83	63.47
c-d-e	0.44	265 K	9.4 M	1.01	85.79	91.49	55.42
d	1.00	152 K	5.5 M	0.58	84.21	90.77	61.13
d-e	0.63	95 K	3.5 M	0.36	82.93	90.89	56.16
e	1.00	38 K	1.5 M	0.15	77.09	89.62	54.16
Standalone (Liang et al., 2020)	1.00	1.8 M	3.6 M	6.88	16.90	87.93	10.03
FedAvg (Liang et al., 2020)	1.00	1.8 M	3.6 M	6.88	67.74	58.99	58.99
LG-FedAvg (Liang et al., 2020)	1.00	1.8 M	3.6 M	6.88	69.76	91.77	60.79

Table 2: Results of combination of various computation complexity levels for CIFAR10 dataset. Full results can be found in Table 8.

Evaluation

IntelliSys Lab

Model	Ratio	Parameters	FLOPs	Space (MB)	Perplexity
a	1.00	19.3 M	1.4 B	73.49	3.37
a-e	0.53	10.2 M	718.6 M	38.86	3.75
a-b-c-d-e	0.37	7.2 M	496.6 M	27.55	3.55
b	1.00	9.1 M	614.8 M	34.74	3.46
b-e	0.56	5.1 M	342.0 M	19.49	3.90
b-c-d-e	0.46	4.2 M	278.7 M	16.07	3.64
c	1.00	4.4 M	290.1 M	16.92	3.62
c-e	0.62	2.8 M	179.7 M	10.57	3.89
c-d-e	0.58	2.6 M	166.7 M	9.85	3.66
d	1.00	2.2 M	140.7 M	8.39	3.83
d-e	0.75	1.7 M	105.0 M	6.31	3.90
e	1.00	1.1 M	69.3 M	4.23	7.41

Table 3: Results of combination of various computation complexity levels for WikiText2 dataset. Full results can be found in Table 9.

Evaluation

IntelliSys Lab

Data		MNIST	CIFAR10	WikiText2
Model		CNN	PreResNet18	Transformer
Hidden size		[64, 128, 256, 512]	[64, 128, 256, 512]	[512, 512, 512, 512]
Local Epoch E		5	5	1
Local Batch Size B		10	10	100
Optimizer			SGD	
Momentum			0.9	
Weight decay			5.00E-04	
Learning rate η		0.01	0.1	0.1
Communication rounds	IID	200	400	100
	non-IID	400	800	200
Decay schedule (0.1)	IID	[100]	[150, 250]	[25, 50]
	non-IID	[200]	[300, 500]	[50, 100]
Embedding Size				256
Number of heads			N/A	8
Dropout				0.2
Sequence length				64

Table 6: Hyperparameters and model architecture used in our experiments.

Evaluation

IntelliSys Lab

Model	Normalization	Scaler	Accuracy IID	
			MNIST	CIFAR10
a	None	N/A	99.2	81.3
	IN		99.5	87.7
	GN		99.5	81.0
	LN		99.5	77.3
	sBN		**99.6**	**91.7**
e	None	N/A	98.6	58.1
	IN		97.4	66.4
	GN		98.7	62.6
	LN		98.6	53.7
	sBN		**98.7**	**77.0**
a-e	None	✗	99.5	80.1
	sBN		99.0	90.1
	None	✓	99.2	80.4
	IN		99.5	86.6
	GN		99.5	76.0
	LN		99.4	71.7
	sBN		**99.5**	**90.1**

Table 4: Ablation Study of IID scenarios. The single-letter models 'a' and 'e' are FedAvg equiped with various normalization methods. The sBN significantly outperforms other existing normalization methods, including the InstanceNorm (IN), GroupNorm (GN) (the number of group G=4), and LayerNorm (LN). Scaler is used for HeteroFL to train models of different sizes and moderately improve the results.

Outline

- Motivation
- HeteroFL
- Evaluation
- Discussion

Discussion

- In sBN, there exist privacy concerns about calculating global statistics cumulatively

Thank You