Federated Model Search via Reinforcement Learning

Dixi Yao, Lingdong Wang, Jiayu Xu, Liyao Xiang,
Shuo Shao, Yingqi Chen, Yanjun Tong

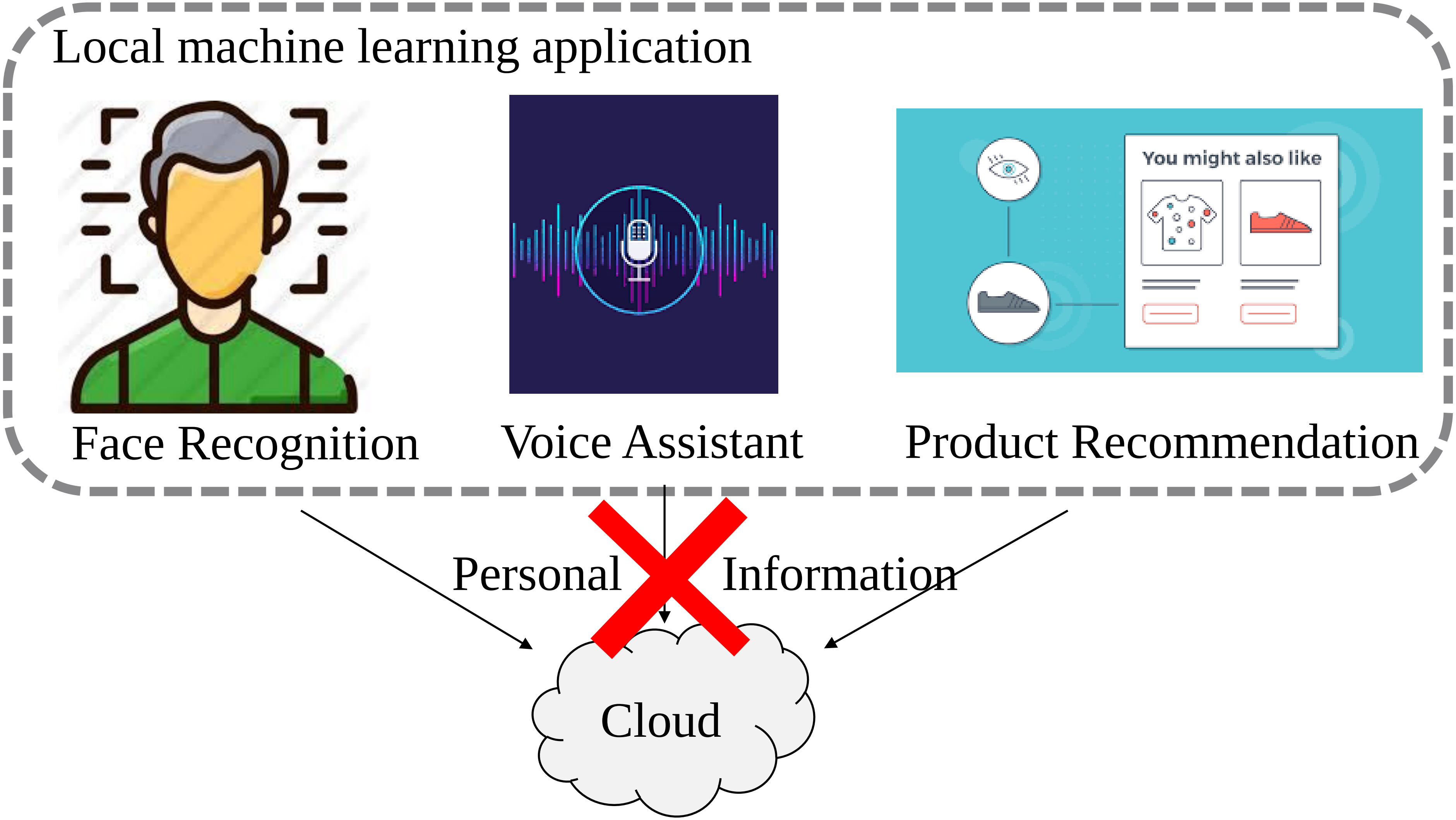
John Hopcroft Center, Shanghai Jiao Tong University

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

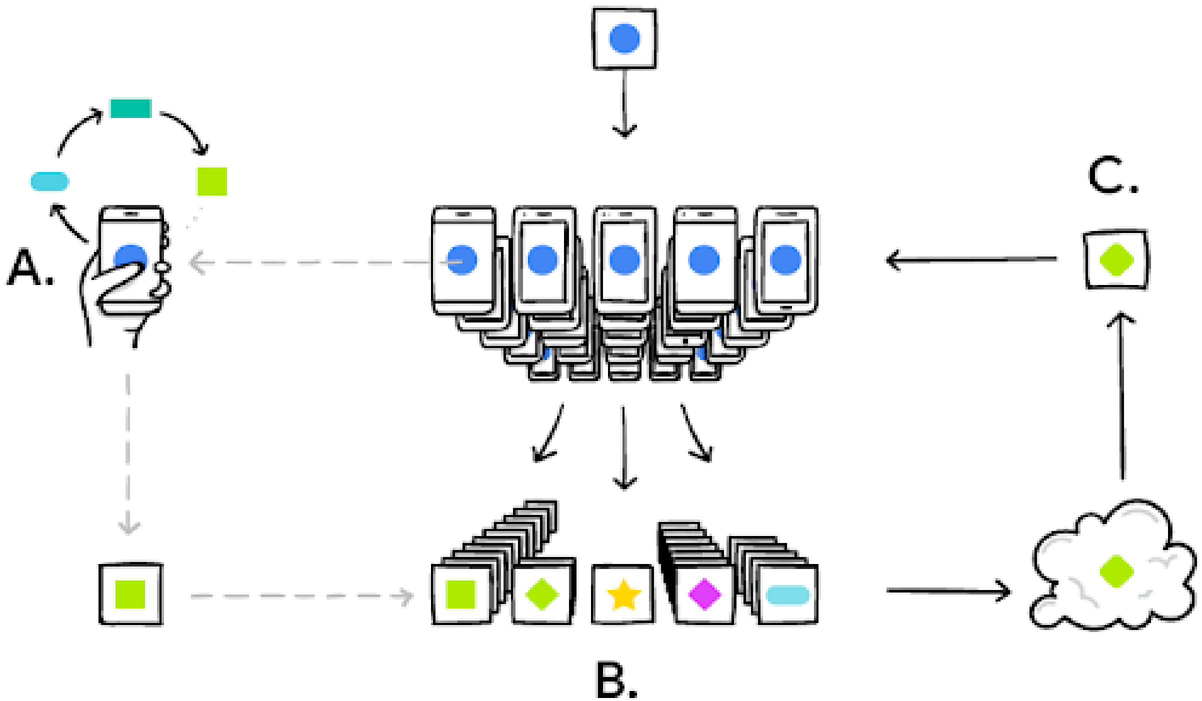
BACKGROUND

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Distributed machine learning



Using Federated Learning!

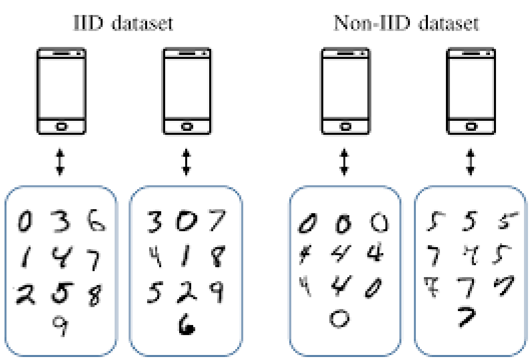


Computational parties collaboratively learn a shared model while keeping all training data local

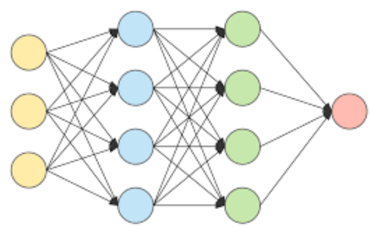
Problem of Predefined Model

Local Data in real world

- Complicated
- Heterogeneous
- Non i.i.d. dataset



Predefined model structure

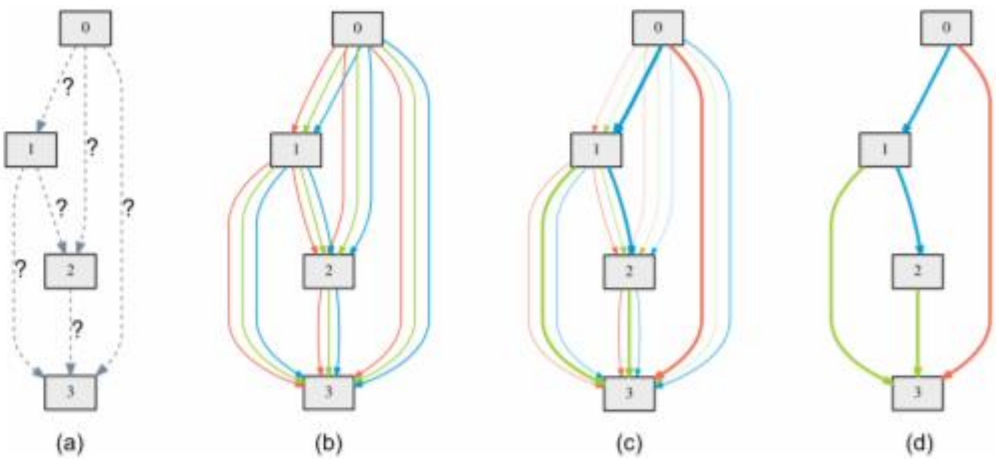


ResNet
VGG
MobileNet
…

Some Serious Problem

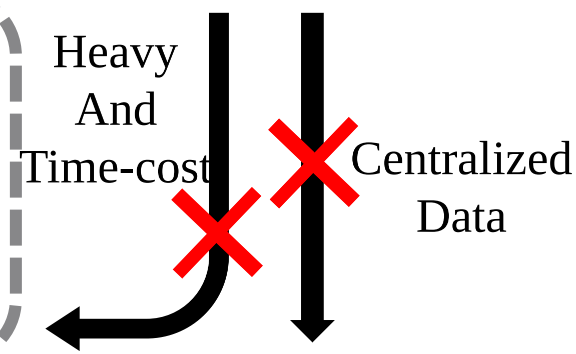
- Fail to converge
- Sub-optimal solution

Using Neural Architecture Search(NAS)!



- Gradient based
- Evolutionary algorithm
- Reinforcement-learning based

Distributed devices



Decentralized data

Besides Computation

Communication also matters

1. Network Context Changes

2. Connection loss or blocked

- ➢ Hard Synchronized SGD
  - Low efficiency if some connection blocked
  - Permanent blocking if some connection lost
- ➢ Asynchronized SGD
  - Update on super-net weights and the global search controller can hardly be parallelized

Our Contribution

- An efficient RL-based federated model search algorithm → achieve low communication and computation costs at the participant end.
- adaptively distribute sub-models according to the transmission conditions. → speed up convergence
- Develop a soft synchronization scheme with delay compensation →fully utilize the stale update to improve searching performance

RL-BASED FEDERATED MODEL SEARCH

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Why Reinforcement Learning?

Evolutionary method : Low efficiency of evolutionary method

Gradient-based method: Send whole large super-net to each participant

Reinforcement Learning-based method

- Each FL participant as agent
- RL agent observes states and performs action
- Super-net updates the policy to maximize the reward function

Natural advantage:

- Small sampled sub-nets →Light Weight
- Sampling models individually →Efficient
- Computing reward parallel →Efficient

Problem Formulation

An optimization Problem

$$\underset{\alpha,\theta}{\text{minimize}} \sum_{k=1}^{K} \sum_{i \in \mathbb{D}_k} \mathcal{L}(x_i, y_i; \alpha, \theta)$$

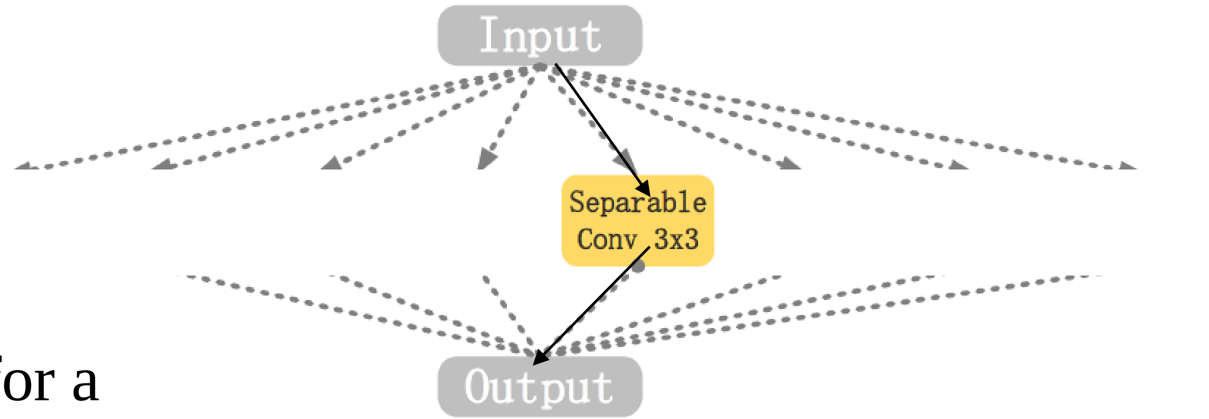
K : total number of participants
k : local participant
$\mathbb{D}_k$: local dataset of participant
α : architecture parameter
θ : model weights

⬇

A Markov Decision Process

State: the structure of the model θ
Policy: parameterized by α
Action: generate sub-models and train them on $\mathbb{D}_k$
Reward: accuracy loss over the training data

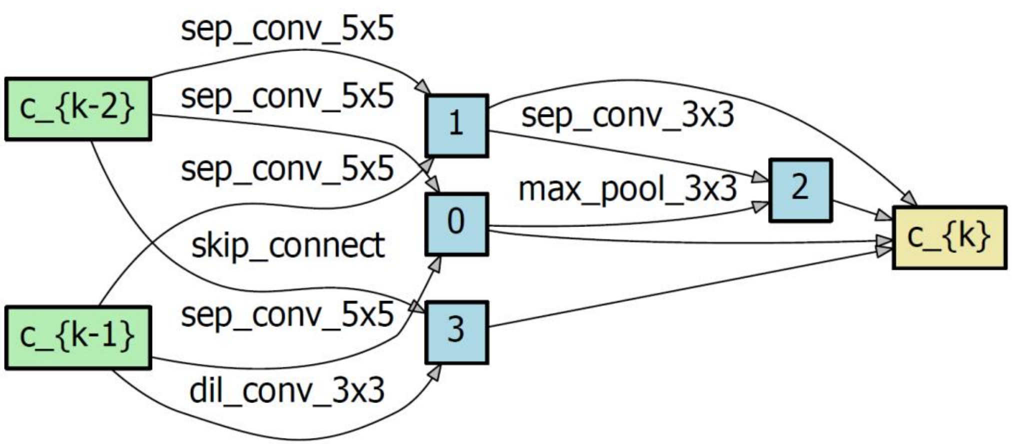
Design Space:



generate a mask for a federated participant

[0,0,0,0,1,0,0,0]

put edges together



Optimization

Two challenges

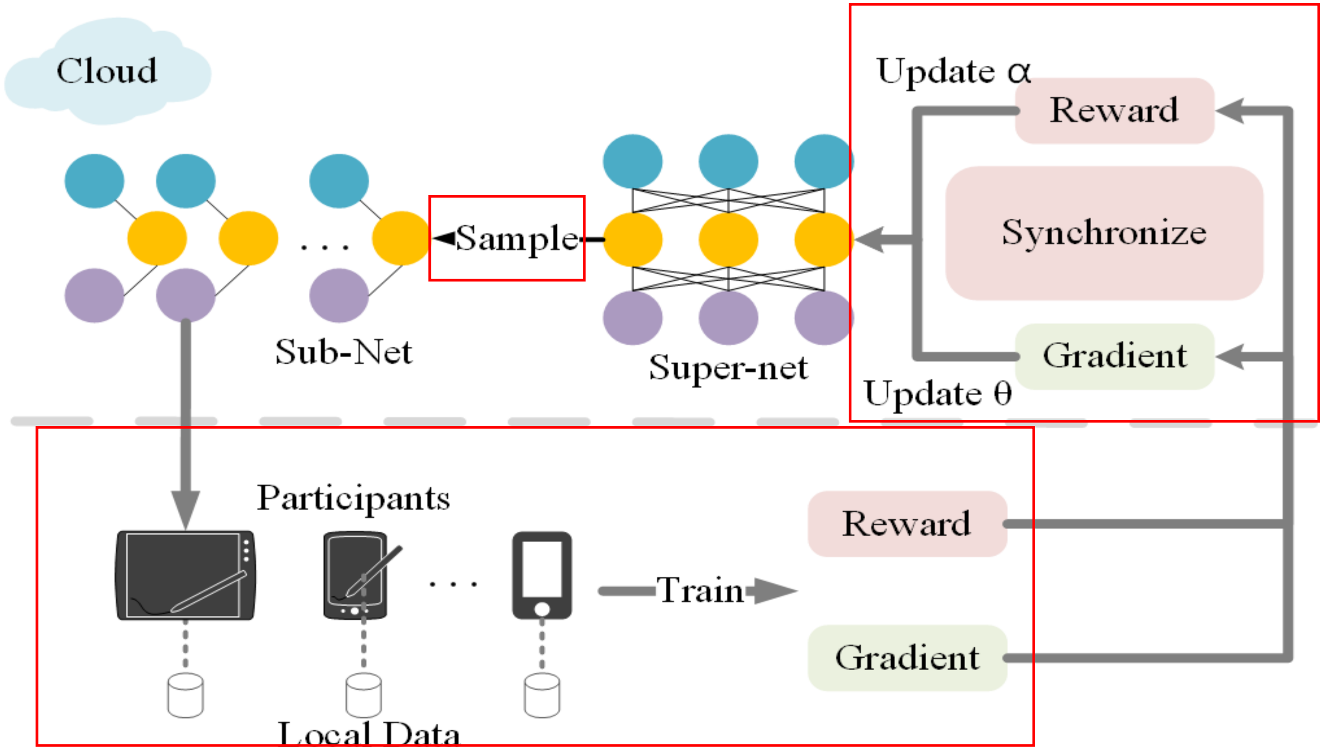
- fail to adapt to non i.i.d.s data or overfitting
- optimize α and $\theta^{`}$ at the same time

$\theta^{`}$ → regular SGD + FedAvg

α → Gather rewards + policy net

FedAvg: B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, ser. Proceedings of Machine Learning Research, A. Singh and J. Zhu, Eds., vol. 54. Fort Lauderdale, FL, USA: PMLR, 4 2017, pp. 1273–1282.

Overall Architecture



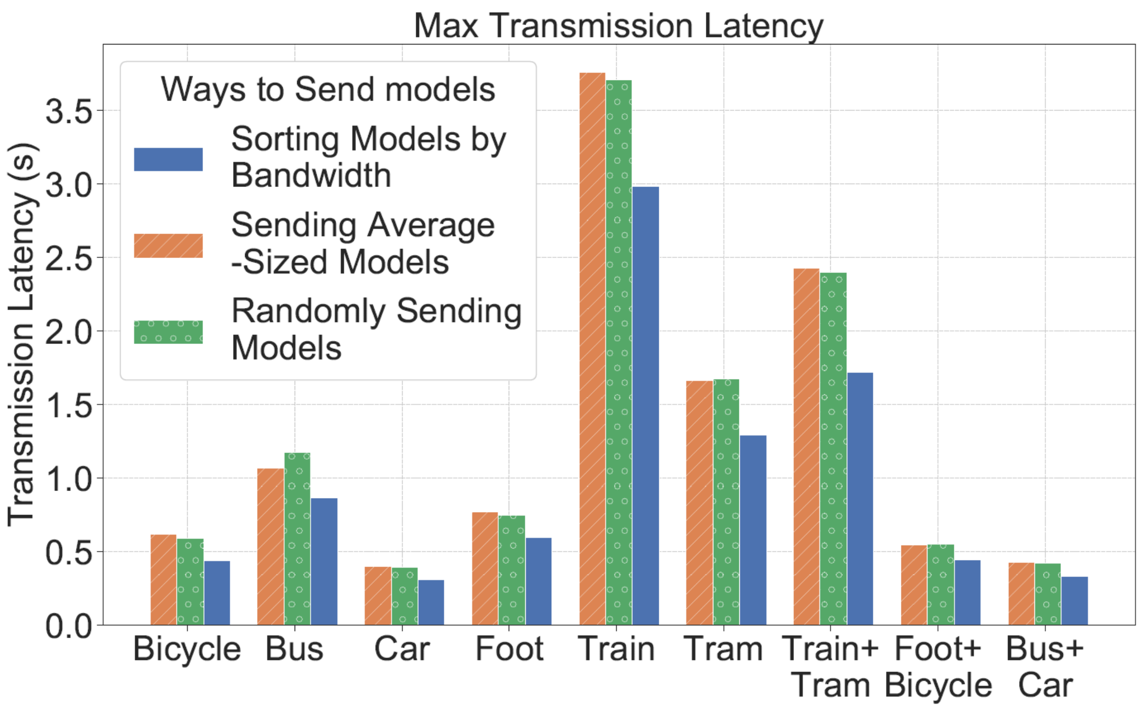
Step 1: sample sub-models:

Step 2: calculate rewards and sub-models weights individually

Step 3: synchronize and update α and θ

Step 4: back to step 1 unless converged

Adaptive Transmission



Participant under worse network conditions get smaller sub-models

Dataset: Hooft J , Petrangeli S , Wauters T , et al. HTTP/2-Based Adaptive Streaming of HEVC Video Over 4G/LTE Networks[J]. IEEE Communications Letters, 2016, 20(11):2177-2180.

DELAY-COMPENSATED FEDERATED MODEL SEARCH

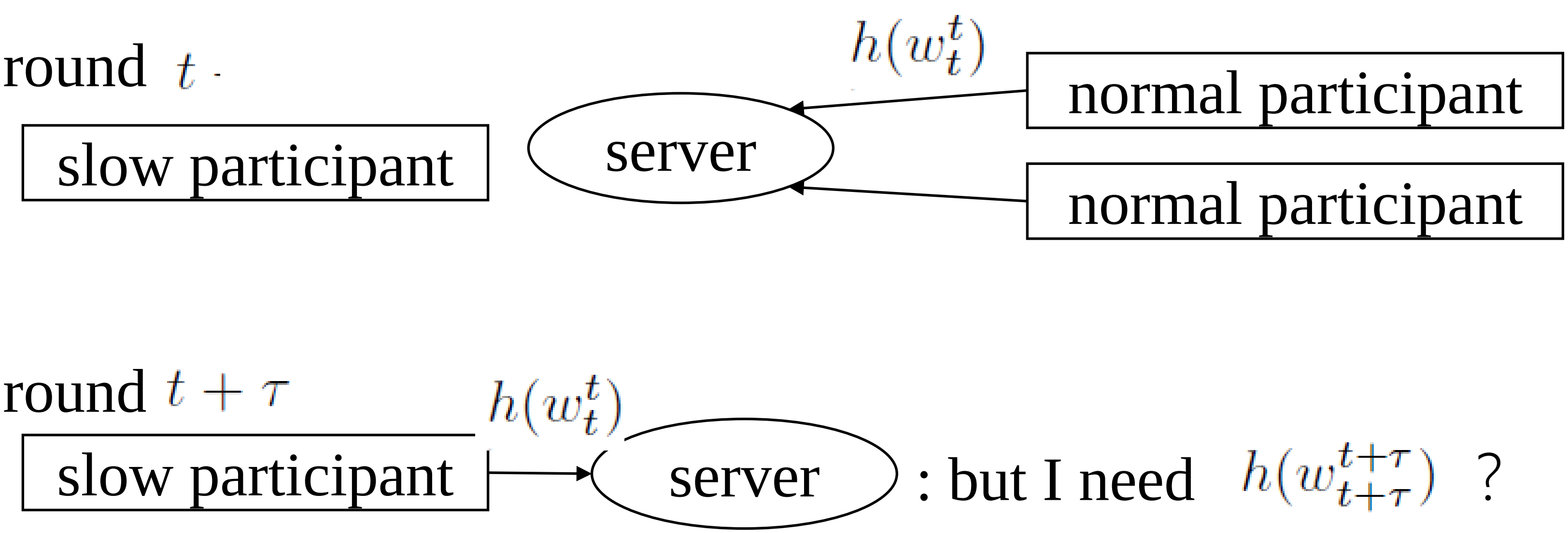
上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Delay-Compensated Federated Model Search

Wait for ~~all~~ participant

- Only wait for most
- Ignore the stragglers for the time being

However, both architecture parameters and models' weights can be stale in the distributed training



unavailable

Stale gradient

$$h(w_{t+\tau}^{t+\tau}) \approx h(w_{t+\tau}^{t})$$
$$\approx h(w_t^t) + \lambda h(w_t^t) \odot h(w_t^t) \odot (w_{t+\tau}^t - w_t^t)$$

$$\nabla_{\alpha_{t+\tau}} \log(p(g_{t+\tau}^m)) \approx \nabla_{\alpha_t} \log(p(g_t^m)) +$$
$$\lambda \nabla_{\alpha_t} \log(p(g_t^m)) \odot \nabla_{\alpha_t} \log(p(g_t^m)) \odot (\alpha_{t+\tau} - \alpha_t)$$

What about $\nabla_\alpha J(\alpha)$?

$$= \frac{1}{M} \sum_{m=1}^{M} R(w_{t+\tau}^{t+\tau}) \nabla_{\alpha_{t+\tau}} \log(p(g_{t+\tau}^m))$$
$$\approx \frac{1}{M} \sum_{m=1}^{M} R(w_t^t) \nabla_{\alpha_{t+\tau}} \log(p(g_{t+\tau}^m)),$$

EXPERIMENT

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Implementation

Dataset: Cifar10, Cifar100, SVHN
Non i.i.d. Data Generation: Dirichlet distribution
Server: Nvidia GTX 1080Ti
Clients: Nvidia Jetson TX2 / Nvidia GTX 1080 Ti
Communication Platform: Pytorch Distributed RPC
Four Phases: Warmup, searching, training from scratch, evaluating

DEFAULT EXPERIMENTAL SETTINGS

Name	Value	Name	Value
batch size	256	# participant (K)	10
learning rate (θ)	0.025	learning rate (P3, centralized)	0.025
momentum (θ)	0.9	momentum (P3, centralized)	0.9
weight decay (θ)	0.0003	weight decay (P3, centralized)	0.0003
gradient clip (θ)	5	gradient clip (P3, centralized)	5
learning rate (α)	0.003	learning rate (P3, FL)	0.1
weight decay (α)	0.0001	momentum (P3, FL)	0.5
gradient clip (α)	5	weight decay (P3, FL)	0.005
baseline decay (α)	0.99	# warm-up steps	10000
cutout [28]	16	# searching steps	6000
random clip	4	# training epochs	600
random horizontal flapping	0.5	# FL training steps	6000

Accuracy

CENTRALIZED EVALUATION ACCURACIES OF SEARCHED MODELS ON CIFAR10

Method	Error(%)	Param(M)	Strategy	FL	NAS
RL-based Federated Model Search					
DARTS (1st order) [7]	3.00	3.3	grad		✓
DARTS (2nd order)	2.81	3.3	grad		✓
ENAS [13]	2.89	4.6	RL		✓
Ours	**2.62**	3.6	RL	✓	✓
Delay-Compensated Federated Model Search					
use (70% staleness)	2.84	3.2	RL	✓	✓
throw (70% staleness)	3.00	4.0	RL	✓	✓
Ours(70% staleness)	**2.72**	3.2	RL	✓	✓
Ours (10% staleness)	**2.59**	**2.7**	RL	✓	✓

FEDERATED EVALUATION ACCURACIES OF SEARCHED MODELS ON CIFAR10

Method	Error(%)	Param(M)	Strategy	FL	NAS
RL-based Federated Model Search					
FedAvg [20]	15.00	-	hand	✓	
EvoFedNAS(big) [16]	13.32	-	evol	✓	✓
EvoFedNAS(small)	16.64	-	evol	✓	✓
Ours	13.36	3.6	RL	✓	✓
Delay-Compensated Federated Model Search					
Ours (10% staleness)	**13.25**	2.7	RL	✓	✓

FEDERATED EVALUATION ACCURACIES OF SEARCHED MODELS ON NON-I.I.D. DATASETS

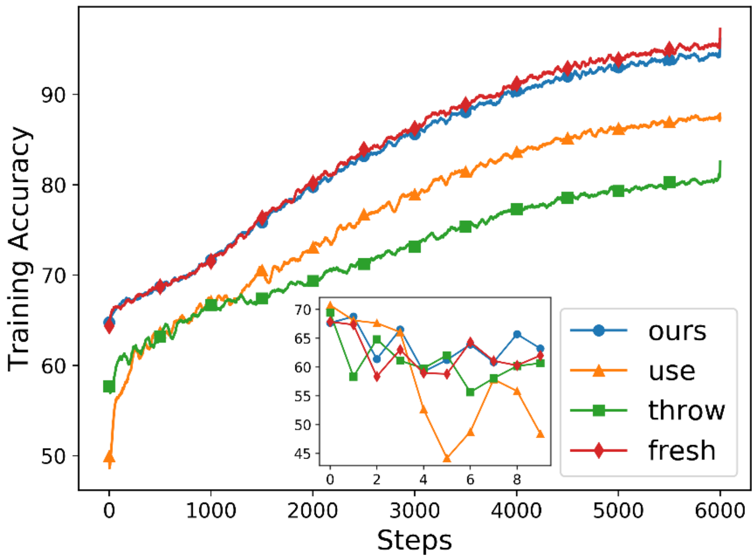
Method	Error(%)	Param(M)	Strategy	NAS
Non-i.i.d. CIFAR10				
FedAvg * [20]	22.40	58.2	hand	
FedNAS [17]	18.76	4.2	grad	✓
EvoFedNAS(big) [16]	18.73	-	evol	✓
EvoFedNAS(small)	21.06	-	evol	✓
Ours (non i.i.d.)	**18.56**	**3.9**	RL	✓
Non-i.i.d. SVHN				
FedAvg *	10.78	58.2	hand	
Ours (non i.i.d.)	**10.23**	**2.5**	RL	✓

* Using ResNet152 [29] as the base model.

- Competitive to centralized NAS
- Better performance than other FLNAS
- Much smaller model size

Efficiency

Delay Compensation



Fresh: hard synchronization
Ours: delay compensation
Use: directly use stale data
Throw: throw all stale data away

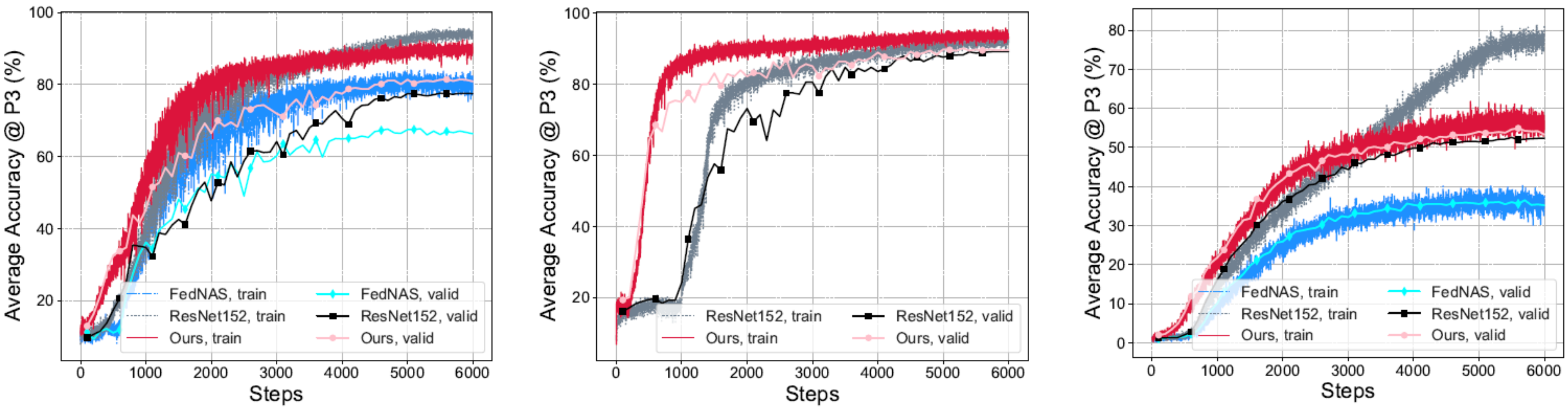
Comparative Performance as fresh, but solving the problem of stragglers

Emulated Running time

SEARCH TIME ON CIFAR10

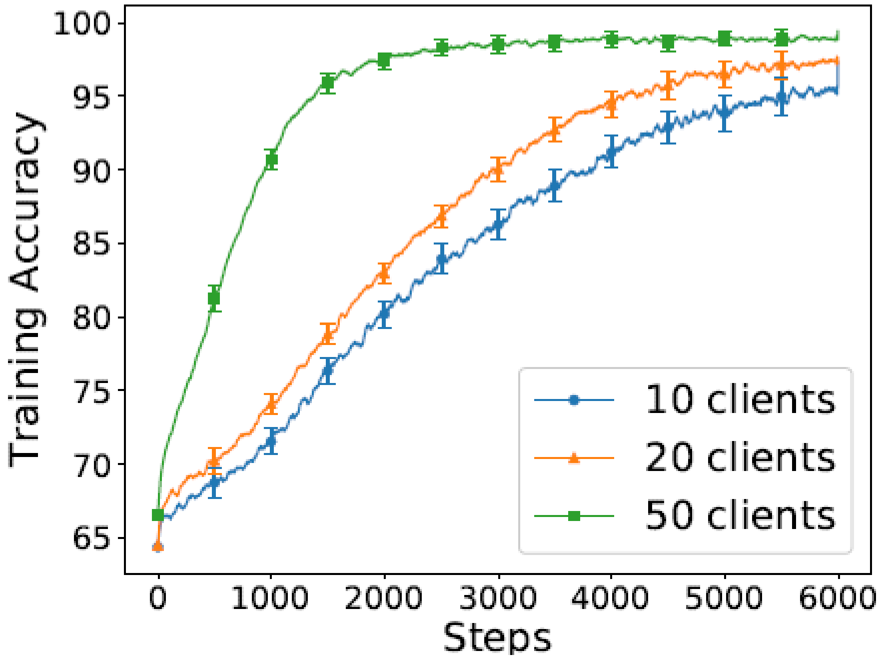
Method	Search Time (hours)	Sub-net Size (M)
FedNAS * [17]	<5	1.93 [1]
EvoFedNAS [16]	16.1	4.23 [1]
Ours (1080Ti)	**<2.5**	**0.27**
Ours (TX2)	**<10**	0.27

Convergence of searched model



Faster convergence and higher accuracies on non i.i.d data

Number of Participants and Transferability



RESULTS OF TRANSFERRING FROM I.I.D. CIFAR10 TO I.I.D. CIFAR100

Method	Acc(%)	Para(M)
DARTS	82.99	3.4
FedNAS	80.28	4.2
Ours	**83.31**	3.6

RESULTS OF TRANSFERRING FROM NON-I.I.D. CIFAR10 TO NON-I.I.D. CIFAR100

Method	Acc(%)	Para(M)
FedAvg	52.57	58.2
FedNAS	36.01	4.2
Ours	**54.63**	3.9

- more participants → less fluctuation
- Almost the same accuracy performance
- Performs well in large-scale settings

perform over a small dataset and later transfer the model to a larger dataset.

Conclusion

- A reinforcement learning based federated model search → Automatically search for a best-fit model

Adaptively distributes the training tasks of sub-models to participants, → Highly efficient in communication and computation.

- A soft synchronization scheme and delay-compensated optimizer → Alleviate the staleness

- Abundant experiments, → Outperform the state-of-the-art methods in terms of efficiency and model accuracy, particularly on non-i.i.d. data.

Thanks