Research Talk

Efficient Solutions for Machine Learning at the Edge

Saurav Prakash

Assistant Professor

Electrical Engineering, IIT Madras

Slide credits: Prof. Salman Avestimehr (USC), Prof. Sunwoo Lee (Inha University), Dr. Yue Niu (Meta), Dr. Ramy E. Ali (Samsung)

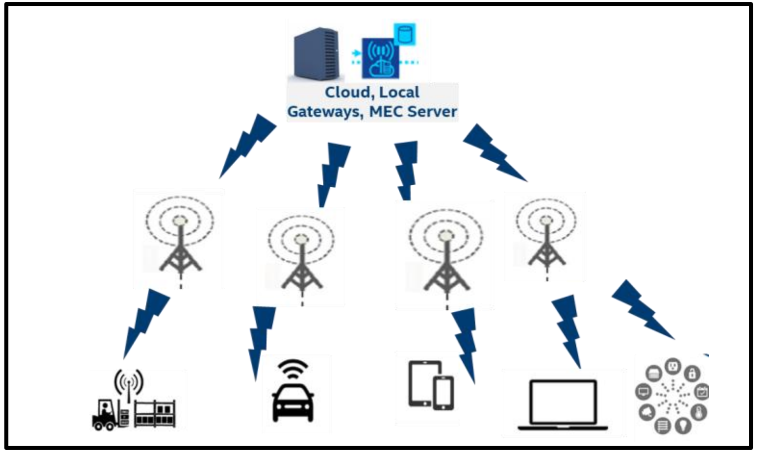
Motivation: AI/ML over Mobile Edge is Gaining Momentum

- **Emerging 6G and beyond** client/edge devices capable of **sensing**, **collecting**, and **processing** data locally.

Data is born at the edge!

- **Privacy**, **latency**, **bandwidth** and **scalability** concerns drive **local processing** of information

- Rich compute, communication, sensing and storage **resources in 6G and beyond** networks suited for **on-device AI and ML,** like **personalized healthcare**

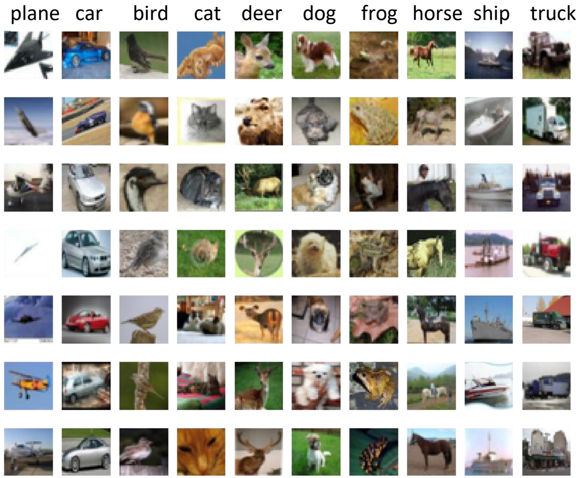


Wireless Mobile Edge

How to enable AI/ML applications at such a massive scale?

Fundamentals

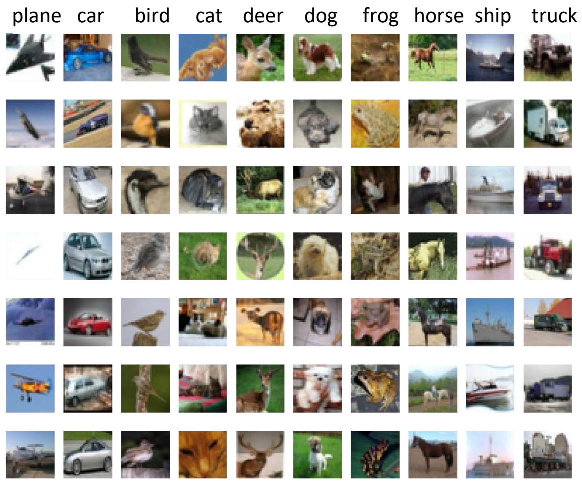
- Machine learning



$$\mathcal{D} = \{\mathbf{x}_j \in \mathbb{R}^{p+1} : j = 1, \cdots, d\}$$

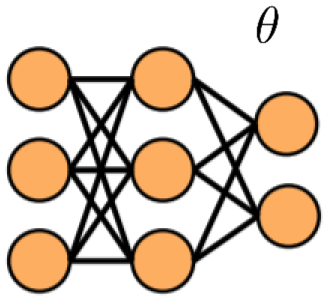
Fundamentals

- Machine learning



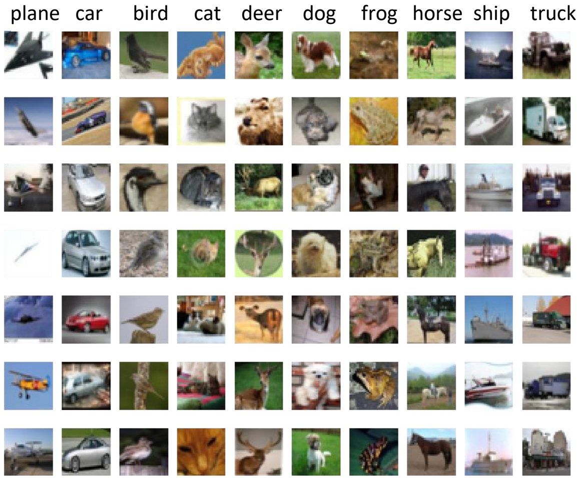
$$\mathcal{D} = \{\mathbf{x}_j \in \mathbb{R}^{p+1} : j = 1, \cdots, d\}$$

Machine Learning



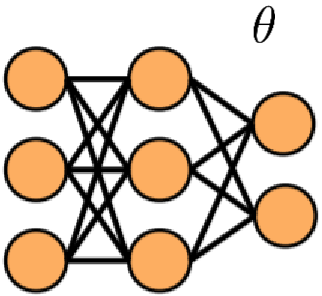
Fundamentals

- Machine learning



$$\mathcal{D} = \{\mathbf{x}_j \in \mathbb{R}^{p+1} : j = 1, \cdots, d\}$$

Machine Learning



Problem: $\theta^* = \arg\min_{\theta \in \mathbb{R}^p} \left(\frac{1}{|\mathcal{D}|} \sum_{\mathbf{x} \in \mathcal{D}} \ell(\theta; \mathbf{x}) + \lambda R(\theta) \right)$

Iterative updates: $\theta^{(t+1)} = \theta^{(t)} - \eta_t \left(\frac{1}{|\mathcal{D}|} \mathbf{g} + \lambda \nabla R\left(\theta^t\right) \right)$

$$\mathbf{g} = \sum_{\mathbf{x} \in \mathcal{D}} \nabla \ell\left(\theta^{(t)}; \mathbf{x}\right)$$

Distributed Machine Learning

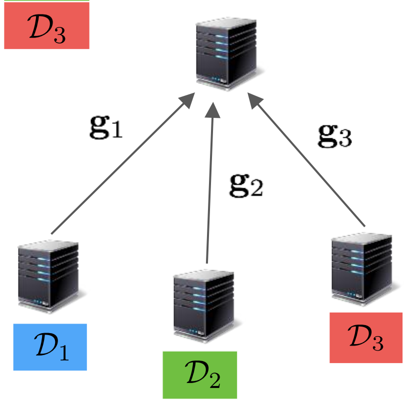
- Data-set D distributedly stored at the cloud

- Training via Gradient Descent

$$\theta^{(t+1)} = \theta^{(t)} - \eta_t \left(\frac{1}{|\mathcal{D}|}\mathbf{g} + \lambda \nabla R\left(\theta^t\right) \right)$$

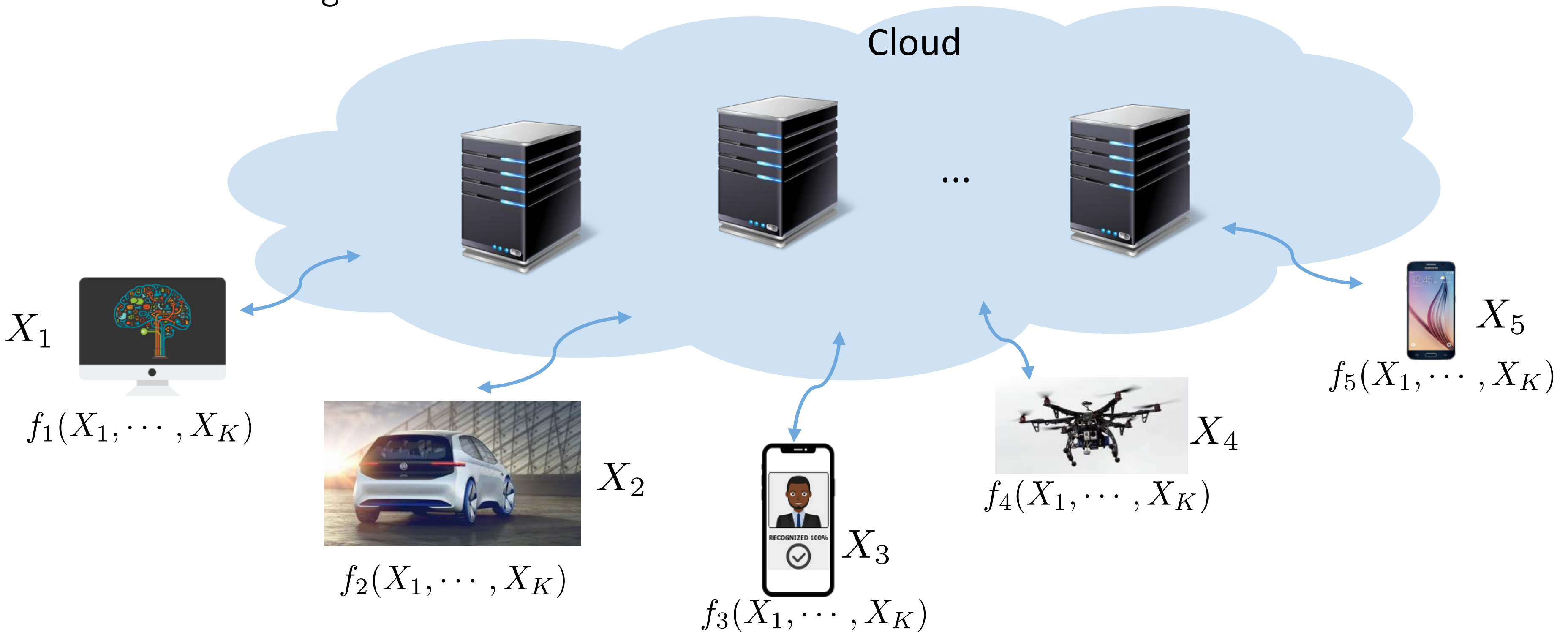
$$\mathcal{D} = \begin{array}{|c|}\hline \mathcal{D}_1 \\ \hline \mathcal{D}_2 \\ \hline \mathcal{D}_3 \\ \hline \end{array}$$

$$\mathbf{g} = \mathbf{g}_1 + \mathbf{g}_2 + \mathbf{g}_3$$

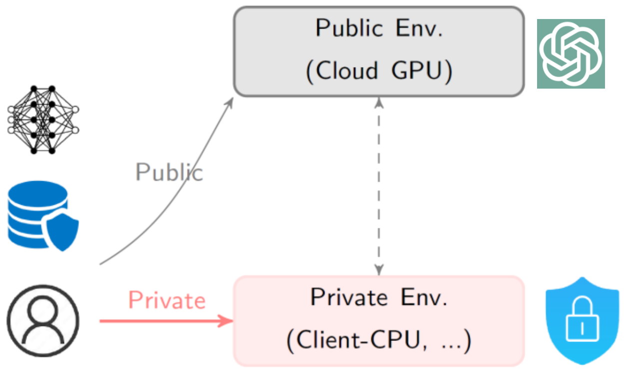


Fundamentals

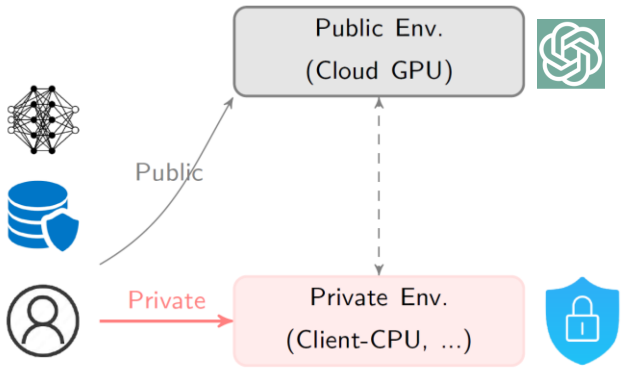
Federated Learning

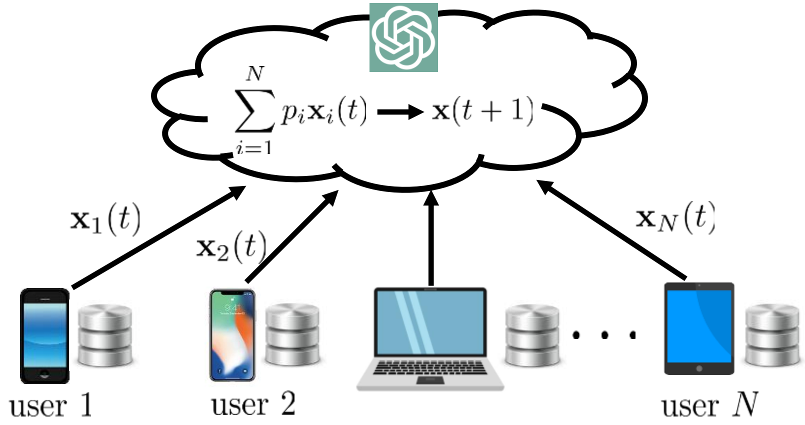


Two Key Edge AI/ML Paradigms

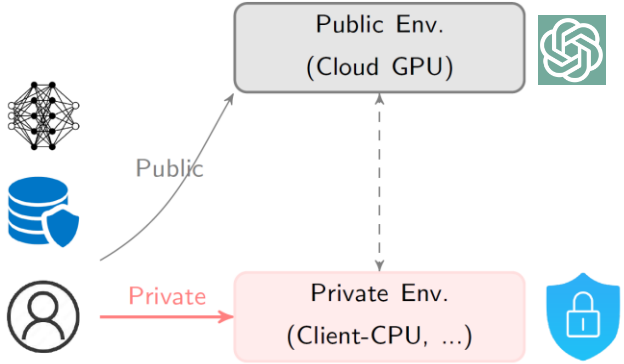


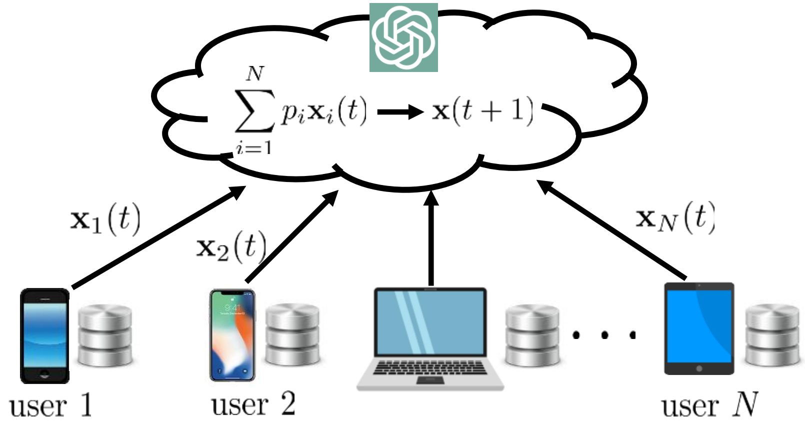
Two Key Edge AI/ML Paradigms





Two Key Edge AI/ML Paradigms





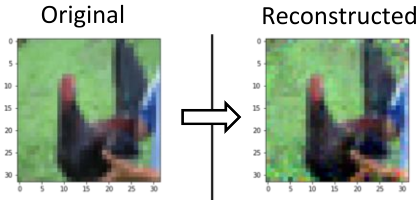
Fundamental Challenges of Edge AI/ML

Resource constraints and heterogeneity

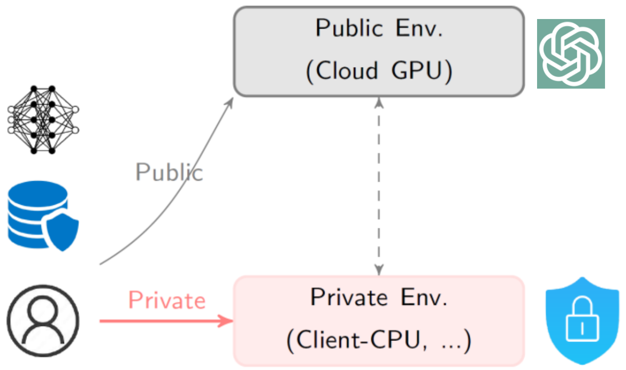
Errors and malicious clients

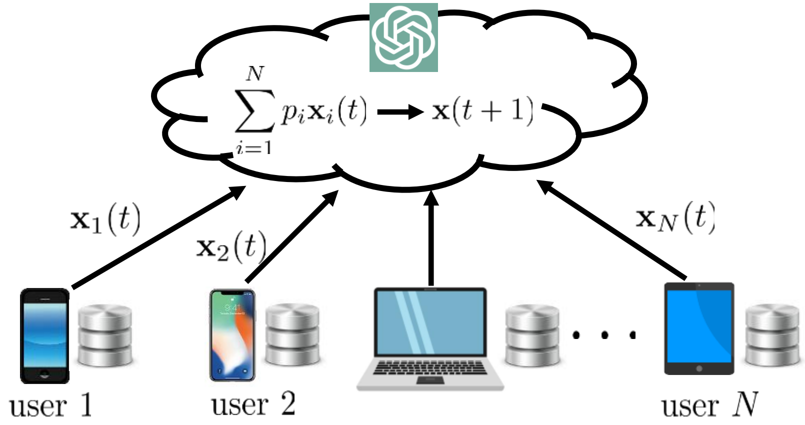
Data leakage through models

Model Inversion



Two Key Edge AI/ML Paradigms





Fundamental Challenges of Edge AI/ML

Resource constraints and heterogeneity

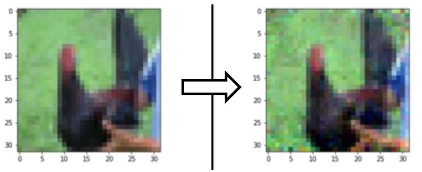
Errors and malicious clients

Data leakage through models

Model Inversion

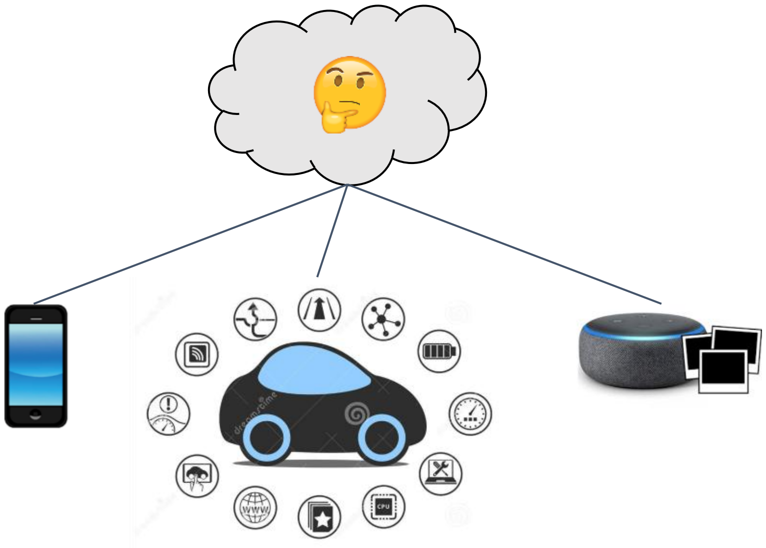
Original

Reconstructed



How to enable Efficient, Robust and Trustworthy solutions for Edge AI/ML?

Training of "Large" Models at the Edge



How to efficiently and effectively **train** large DNN over resource-constrained edge devices?

Key challenges

1. Limited memory and computation (no GPU accelerator)
2. Limited bandwidth and unstable wireless communication
3. Unfair to exclude clients due to limited resources

Training of “Large” Models at the Edge

Samsung
OnePlus 9 Pro

256 GB

Xiaomi
Redmi Note

128 GB

Motorola
Moto G Power

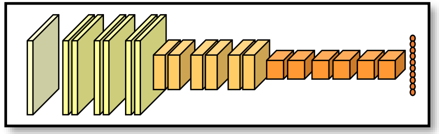
64 GB

The conventional assumption of ‘**homogeneous**’ and ‘**powerful**’ clients **does not hold!!**

Training of “Large” Models at the Edge

Samsung
OnePlus 9 Pro

256 GB



Xiaomi
Redmi Note

128 GB

Motorola
Moto G Power

64 GB

The conventional assumption of ‘**homogeneous**’ and ‘**powerful**’ clients **does not hold!!**

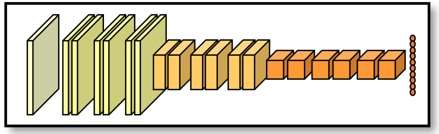
Training of "Large" Models at the Edge

Samsung
OnePlus 9 Pro

Xiaomi
Redmi Note

Motorola
Moto G Power

256 GB



128 GB

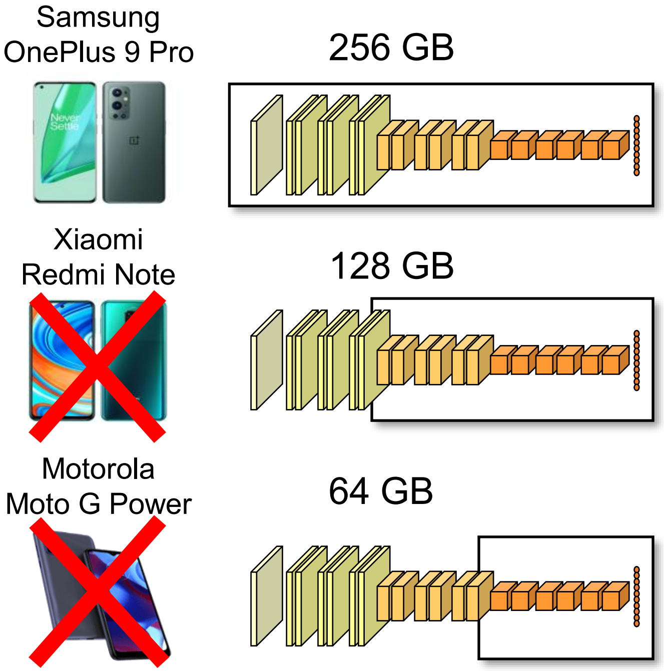
64 GB

The conventional assumption of '**homogeneous**' and '**powerful**' clients **does not hold!!**

What happens when model is so large?

- Small and weak devices cannot even hold the whole model in their memory space!

Training of "Large" Models at the Edge



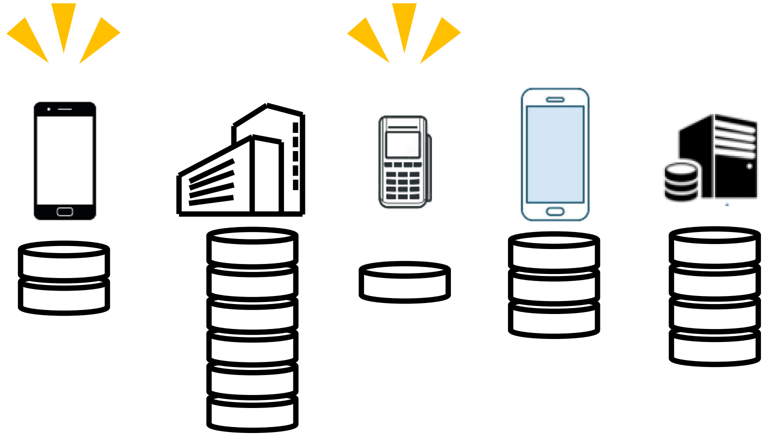
The conventional assumption of '**homogeneous**' and '**powerful**' clients **does not hold!!**

What happens when model is so large?

- Small and weak devices cannot even hold the whole model in their memory space!

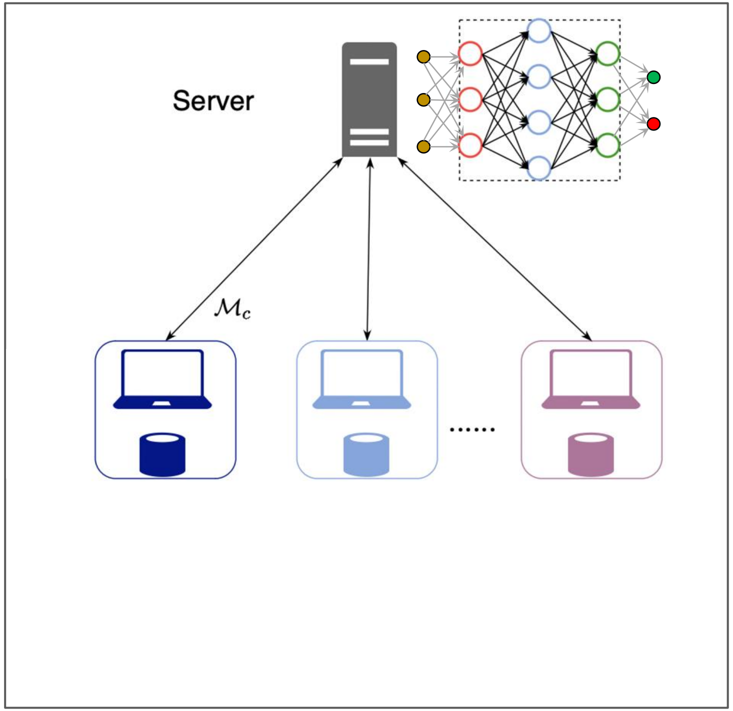
Goal: Enable Weak Client Participation in FL

- We consider ***weak clients*** that cannot effectively train the full model on its own.
  - ➢ Limited memory space
  - ➢ Too weak compute power

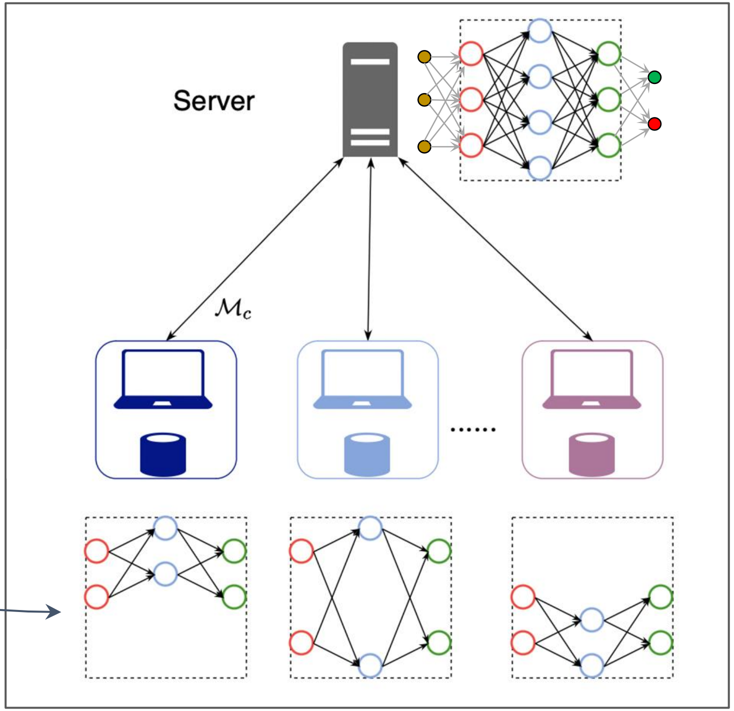


To utilize distributed data in the real-world, we should make all available devices participate in the training!

Idea: Sub-Model Training



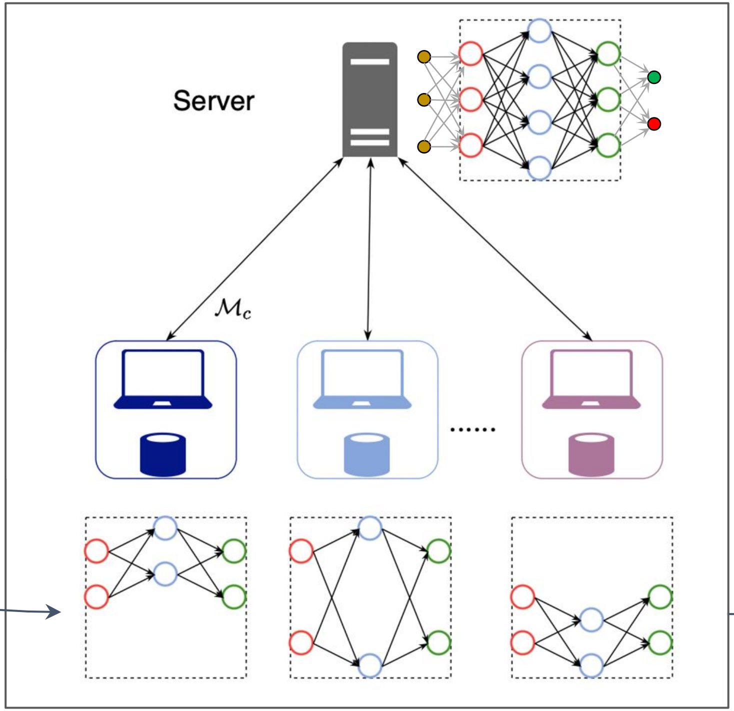
Idea: Sub-Model Training



Each client only
trains a sub-model!!!

Idea: Sub-Model Training

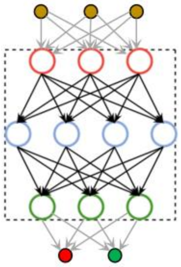
Each client only trains a sub-model!!!

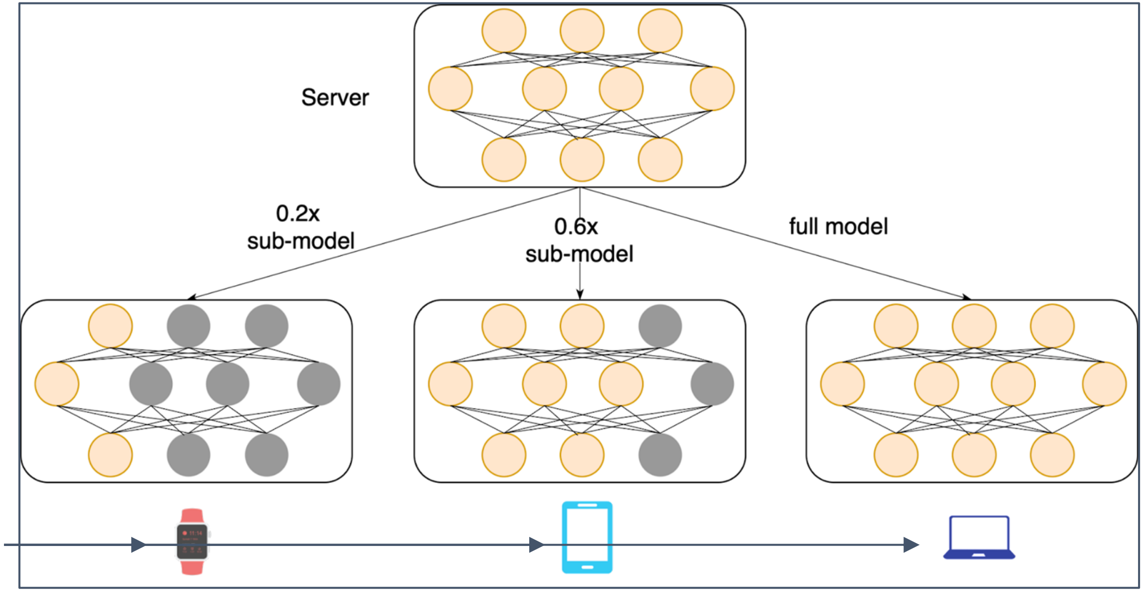


How to create a good sub model???

Related Works on Sub-Model Training

FjORD[1], HeteroFL[2], et al





Different capacity

[1] Horvath, S., etal. *Fjord: Fair and accurate federated learning under heterogeneous targets with ordered dropout.* In *NeurIPS, 2021.*

[2] Diao, E., et al, HeteroFL: Computation and Communication Efficient Federated Learning for Heterogeneous Clients. In *ICLR 2021*.

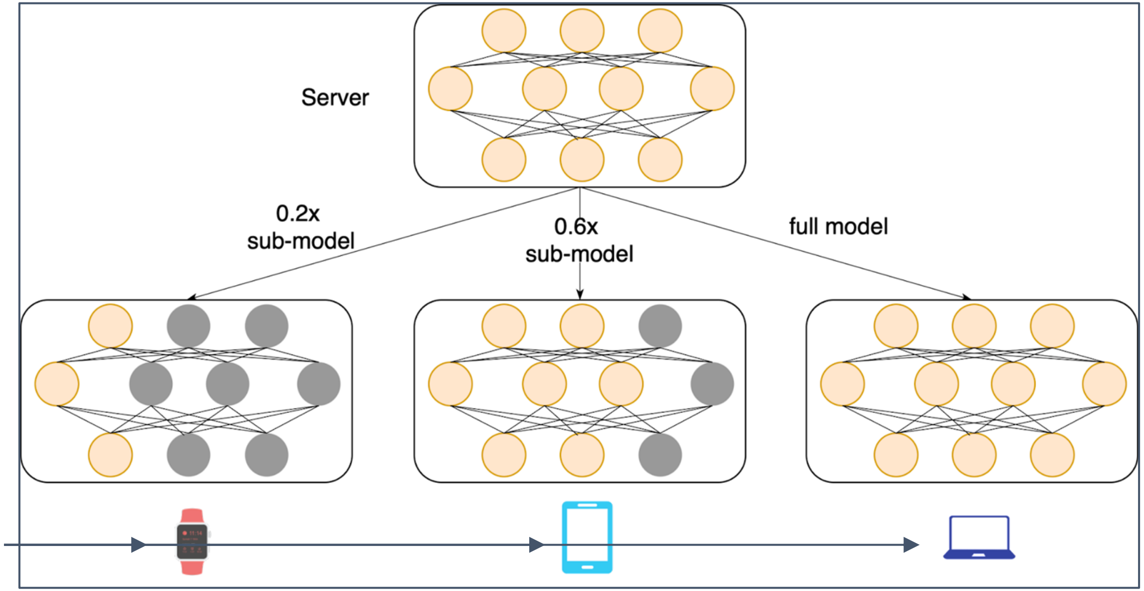
Related Works on Low-Rank Training

FedHM[3], et al

$$W \xrightarrow{SVD} \sum_{i=1}^{N} s_i \cdot \boldsymbol{u}_i \cdot \boldsymbol{v}_i^*$$

$$W \xrightarrow{SVD} \sum_{i=1}^{r} s_i \cdot \boldsymbol{u}_i \cdot \boldsymbol{v}_i^*$$

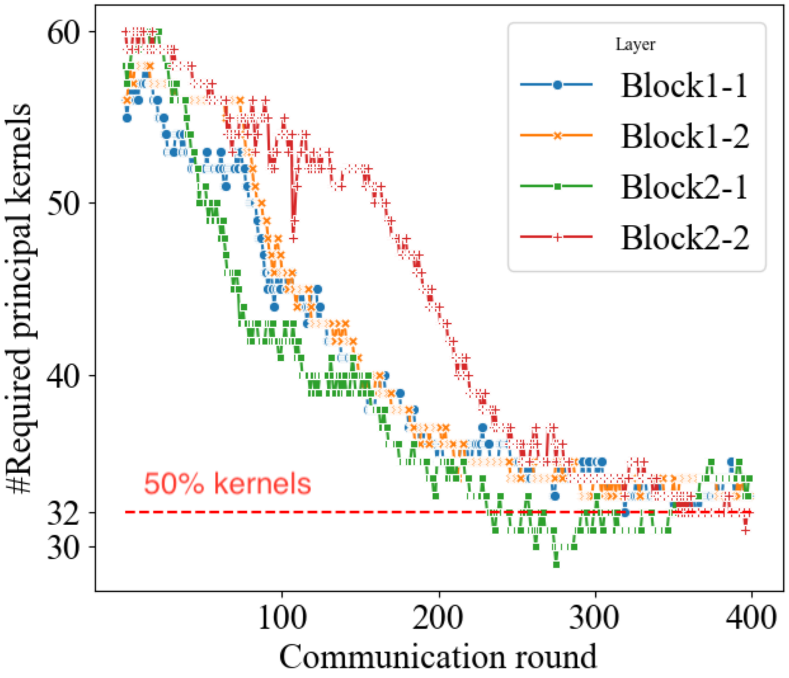
Different capacity



[3] Yao, D., et al. Fedhm: Efficient federated learning for heterogeneous models via low-rank factorization. *arXiv preprint arXiv:2111.14655, 2021*.

But … low-rank compression at clients is not enough

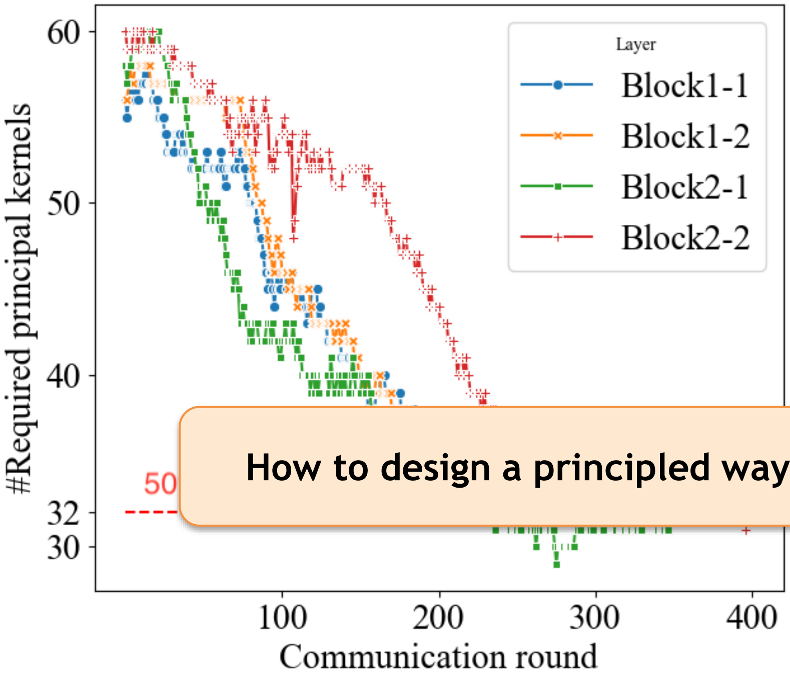
Number of kernels required to approximate the full model (ResNet-18)



Low rank structure is **gradually attained**, a **single low-rank model** incurs **severe reduction** in **model capacity**.

But … low-rank compression at clients is not enough

Number of kernels required to
approximate the full model
(ResNet-18)

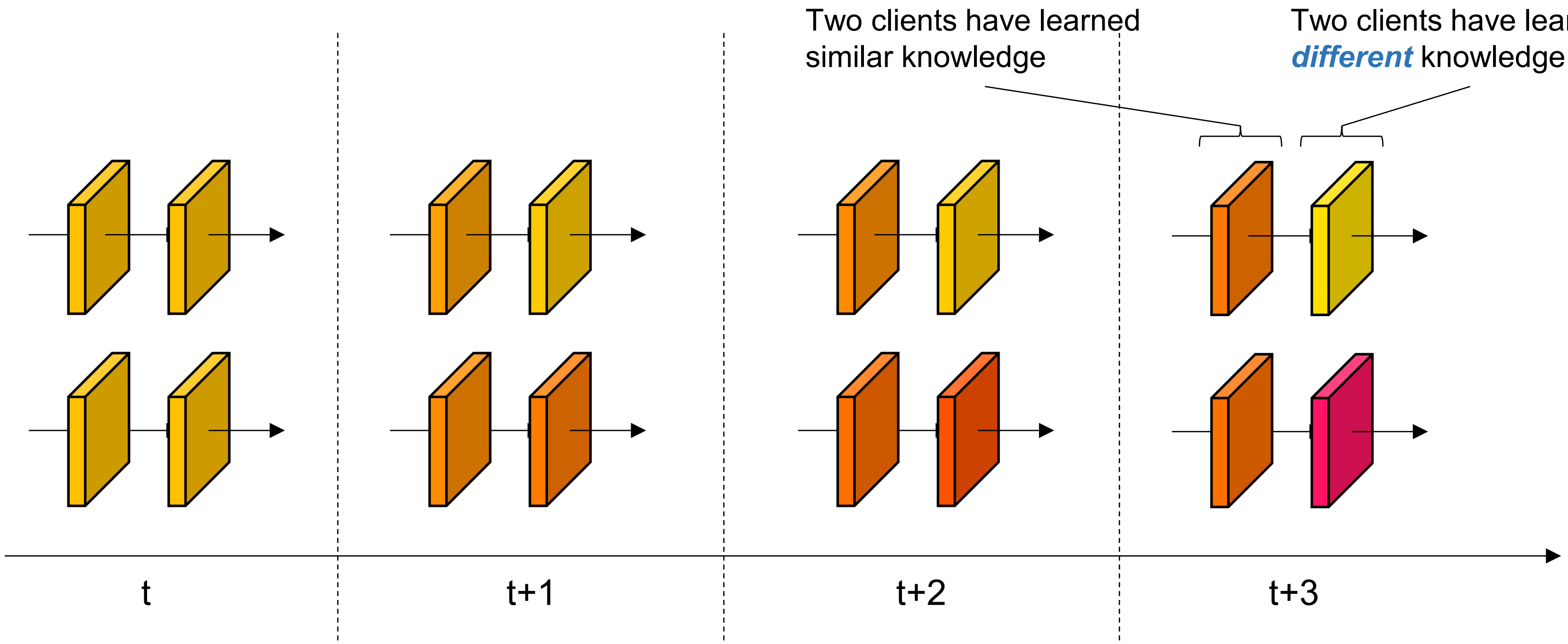


Low rank structure is **gradually attained**, a **single low-rank model** incurs **severe reduction** in **model capacity**.

How to design a principled way of splitting and utilizing the 'weak' clients?

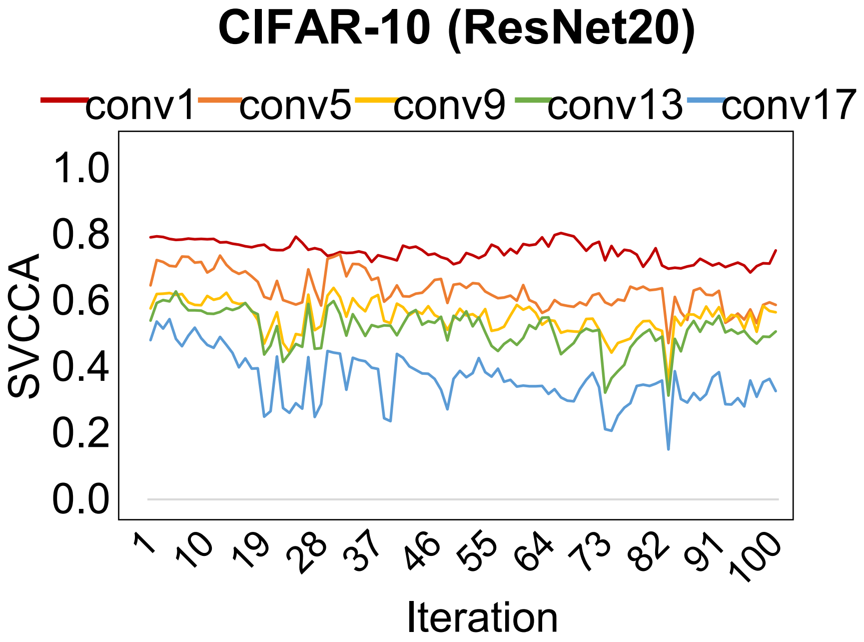
EmbracingFL: Heterogeneous System-Aware FL

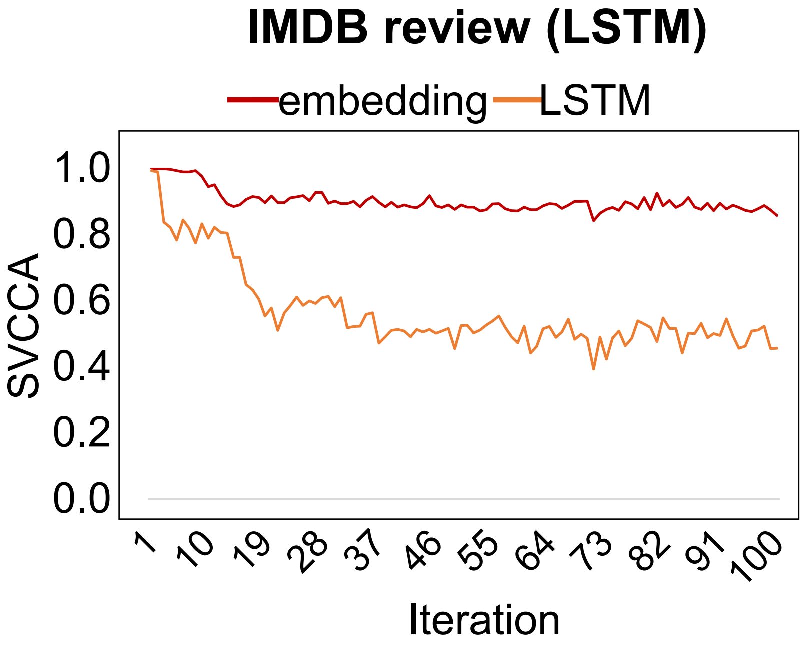
Hypothesis: layers may have different data characteristics



Lee S, Zhang T, **Prakash S**, Niu Y, Avestimehr S. "**Embracing Federated Learning: Enabling Weak Client Participation via Partial Model Training**." *IEEE Transactions on Mobile Computing*, 2024

Empirical Study: Layer-wise Data Representation Analysis

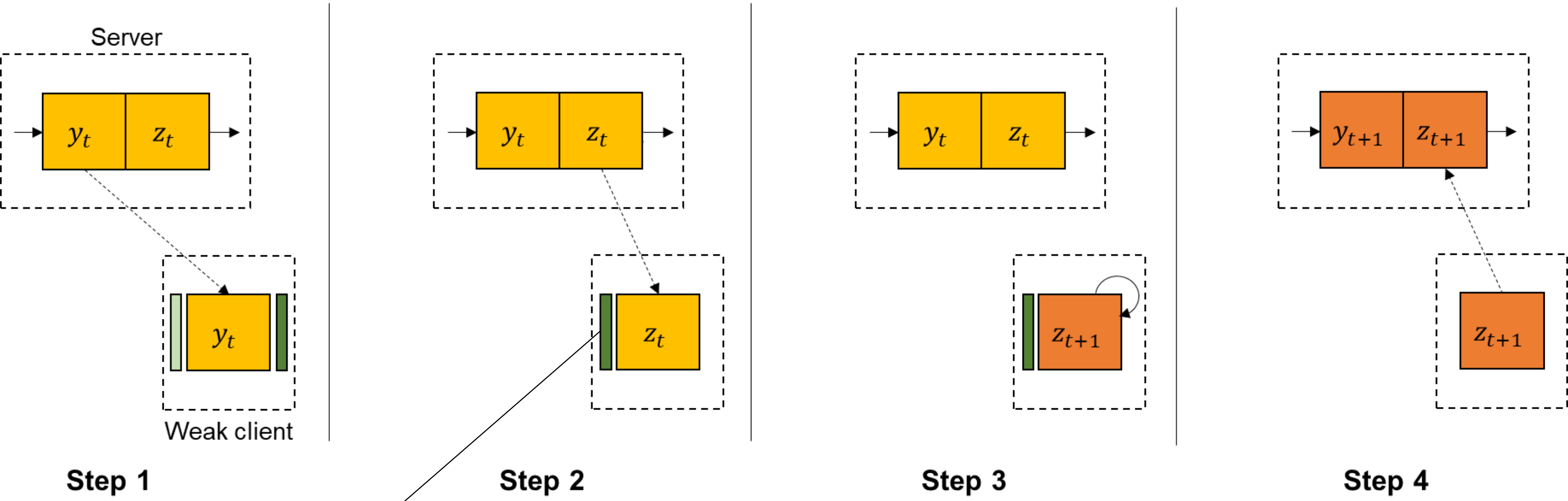




The input-side layers learn similar data across independent clients!

- Singular Vector Canonical Correlation Analysis (SVCCA)
- 128 clients, each client training independently

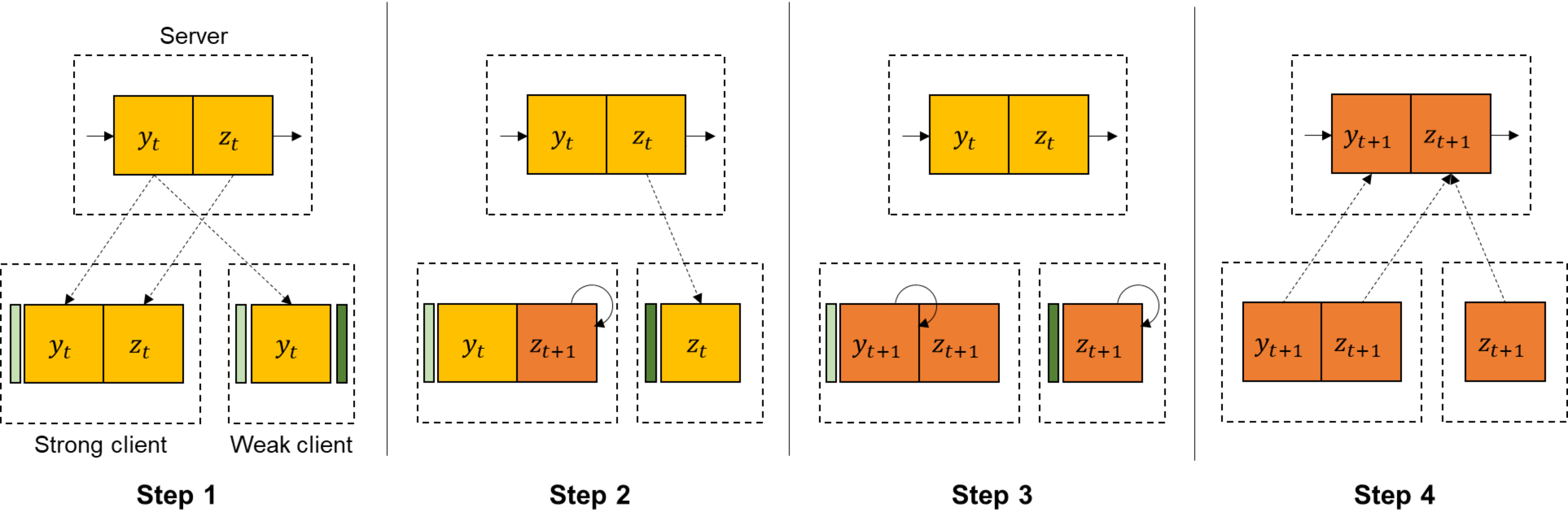
Partial Training at Weak Clients



The intermediate output is recorded and reused multiple steps!

Weak clients contribute only to the output-side subset of layers.

EmbracingFL: Overview of Solution

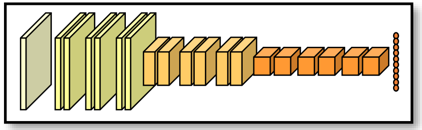


Experimental Settings

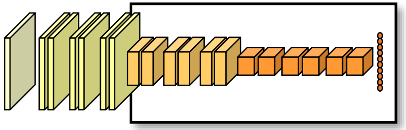
TABLE I
THE MODEL SIZE OF THE THREE DIFFERENT TYPES OF CLIENTS.

Removed layers of Resnet20 (CIFAR-10)	Number of parameters (p)	Number of activations (a)	Capacity
(Strong) -	272,762	6,947,136	1.00
(Moderate) The first conv. layer + the first 3 residual blocks	257,994	2,752,832	0.42
(Weak) The first conv. layer + the first 6 residual blocks	206,346	917,824	0.16

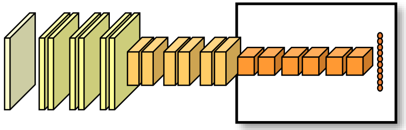
Strong Clients (100%)



Moderate Clients (42%)

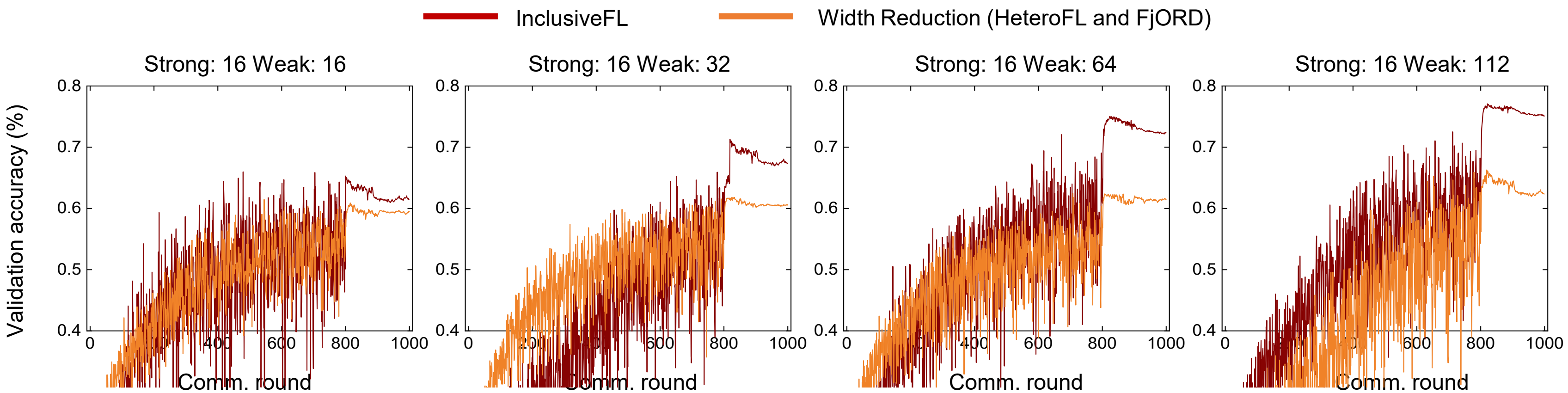


Weak Clients (16%)



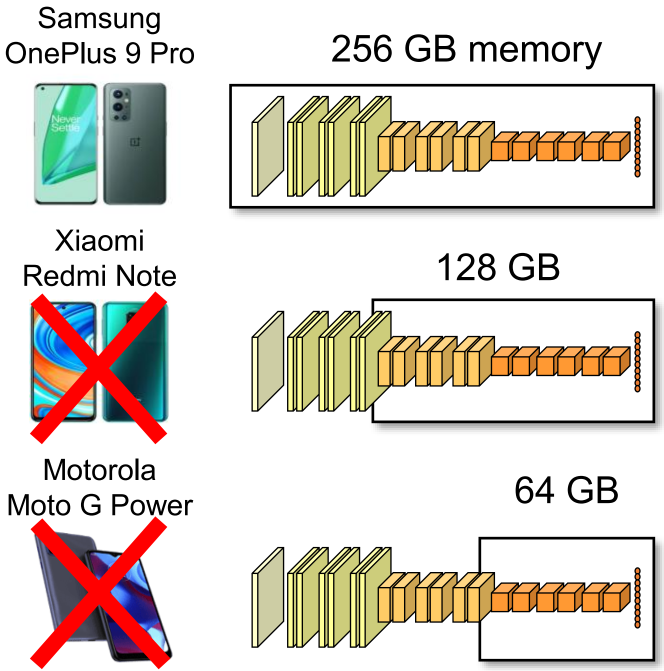
Results: Comparison to SOTA

# of strong	# of weak	Width Reduction	EmbracingFL
16	0	$60.15 \pm 1.5\%$	
16	16	$61.34 \pm 2.1\%$	**66.62** $\pm$ 1.1%
16	32	$62.09 \pm 1.5\%$	**72.60** $\pm$ 1.2%
16	64	$63.68 \pm 3.3\%$	**74.79** $\pm$ 0.8%
16	112	$65.01 \pm 2.9\%$	**77.34** $\pm$ 1.6%



These results empirically demonstrate that EmbracingFL better utilize the ‘weak’ clients than HeteroFL and FjORD!

Training of "Large" Models at the Edge: Is EmbracingFL Enough?

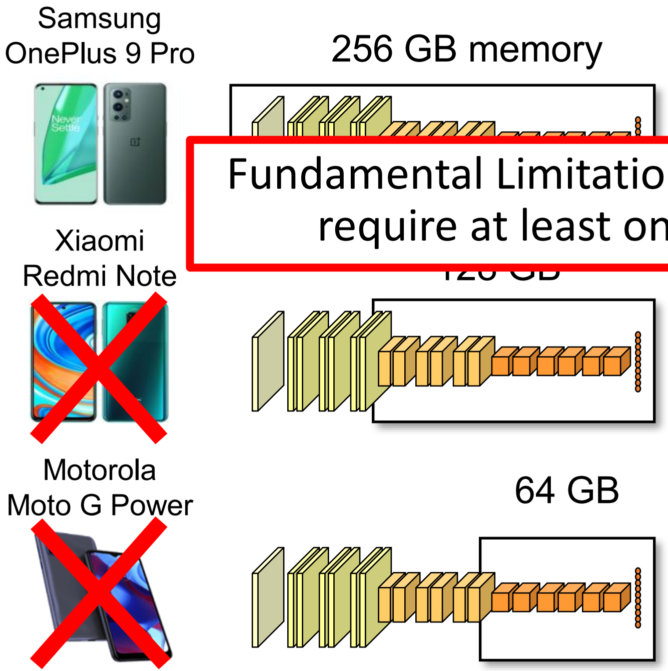


The conventional assumption of 'homogeneous' clients does not hold

What happens when model is so large?

- Small and weak devices cannot even hold the whole model in their memory space!

Training of "Large" Models at the Edge: Is EmbracingFL Enough?



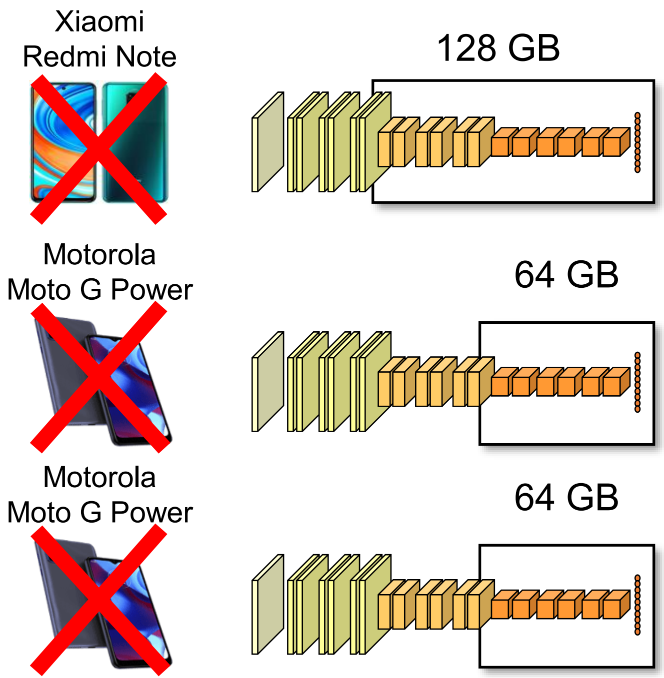
Fundamental Limitation: **All** prior works, including EmbracingFL, require at least one client to train the full-sized model!

What happens when model is so large?

- Small and weak devices cannot even hold the whole model in their memory space!

Training of “Large” Models at the Edge: Is EmbracingFL Enough?

Fundamental Limitation: **All** prior works, including EmbracingFL, require at least one client to train the full-sized model!



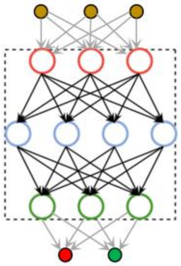
The conventional assumption of ‘homogeneous’ clients does not hold

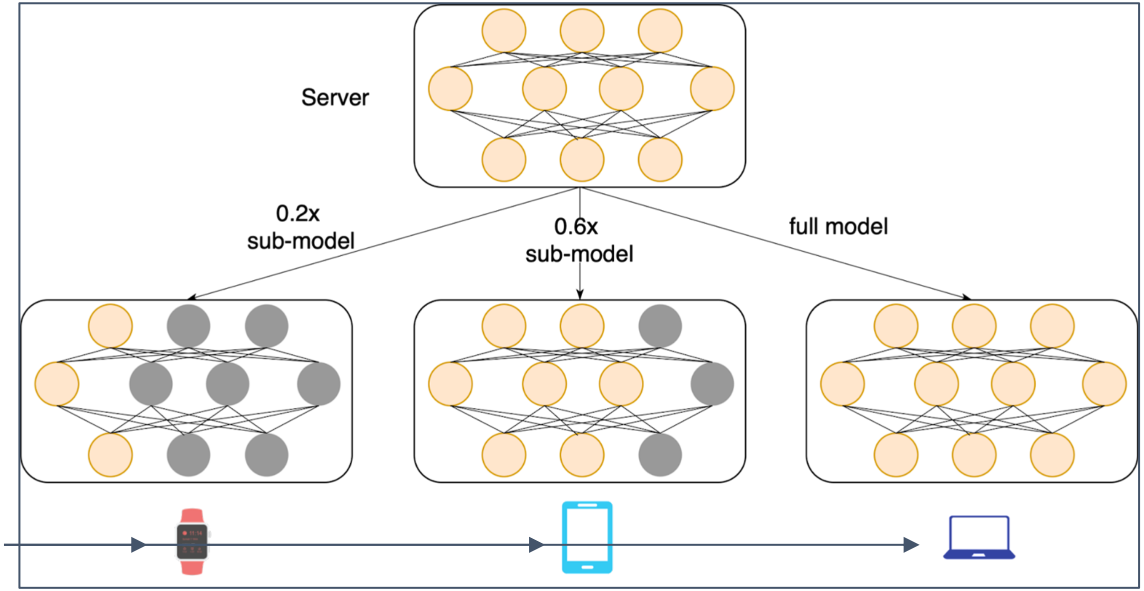
What happens when ALL clients are weak?

- Can the server still train a “large” model?

Related Works on Sub-Model Training

FjORD[1], HeteroFL[2], et al





Different capacity

[1] Horvath, S., etal. *Fjord: Fair and accurate federated learning under heterogeneous targets with ordered dropout.* In *NeurIPS, 2021.*

[2] Diao, E., et al, HeteroFL: Computation and Communication Efficient Federated Learning for Heterogeneous Clients. In *ICLR 2021*.

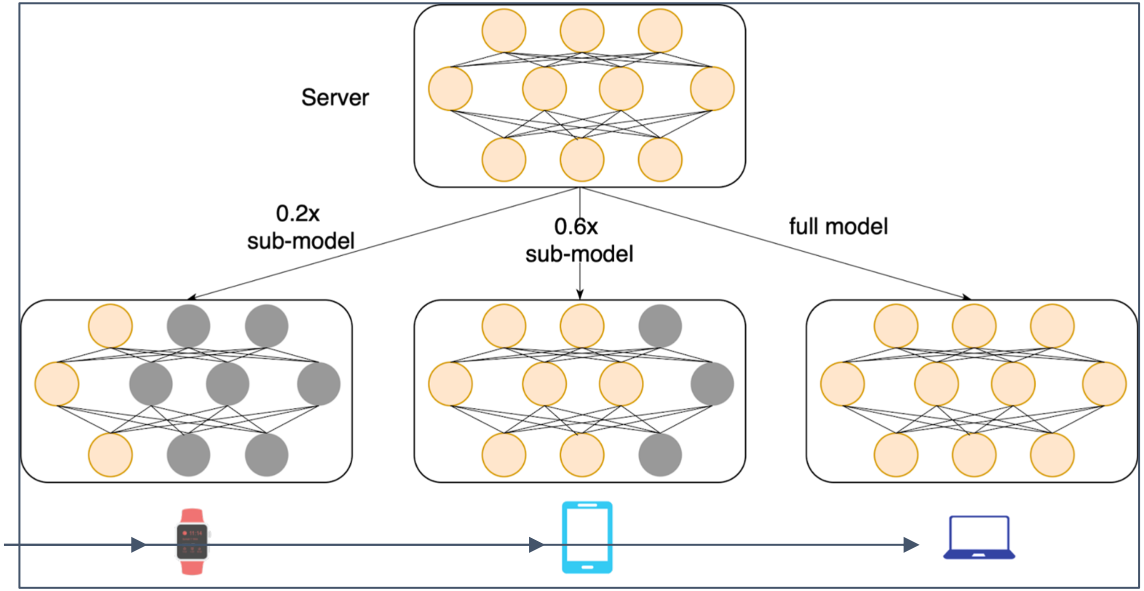
Related Works on Low-Rank Training

FedHM[3], et al

$$W \xrightarrow{SVD} \sum_{i=1}^{N} s_i \cdot \boldsymbol{u}_i \cdot \boldsymbol{v}_i^*$$

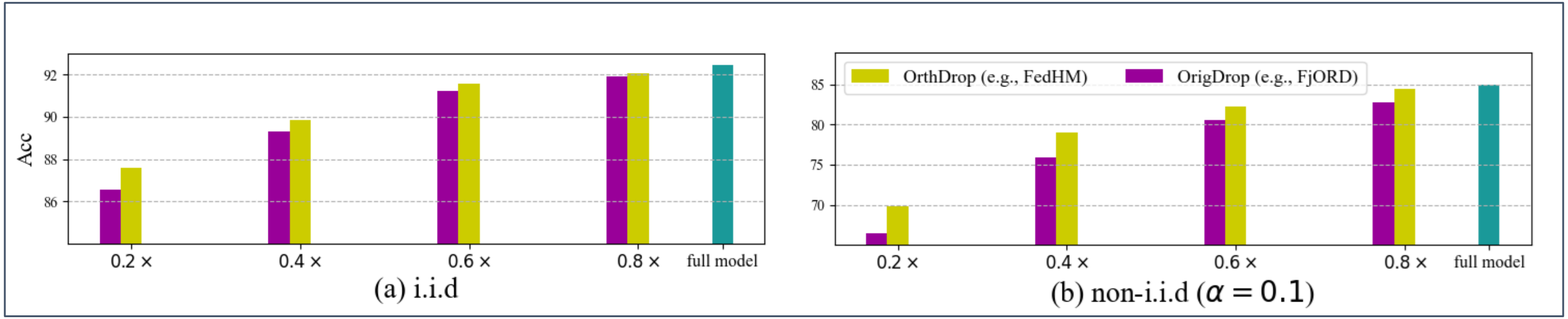
$$W \xrightarrow{SVD} \sum_{i=1}^{r} s_i \cdot \boldsymbol{u}_i \cdot \boldsymbol{v}_i^*$$

Different capacity



[3] Yao, D., et al. Fedhm: Efficient federated learning for heterogeneous models via low-rank factorization. *arXiv preprint arXiv:2111.14655, 2021*.

Empirical Evaluations: ResNet-18/CIFAR-10



model: ResNet-18

dataset: CIFAR-10 (i.i.d, non-i.i.d)

clients: 20 (active) / 100 (total)

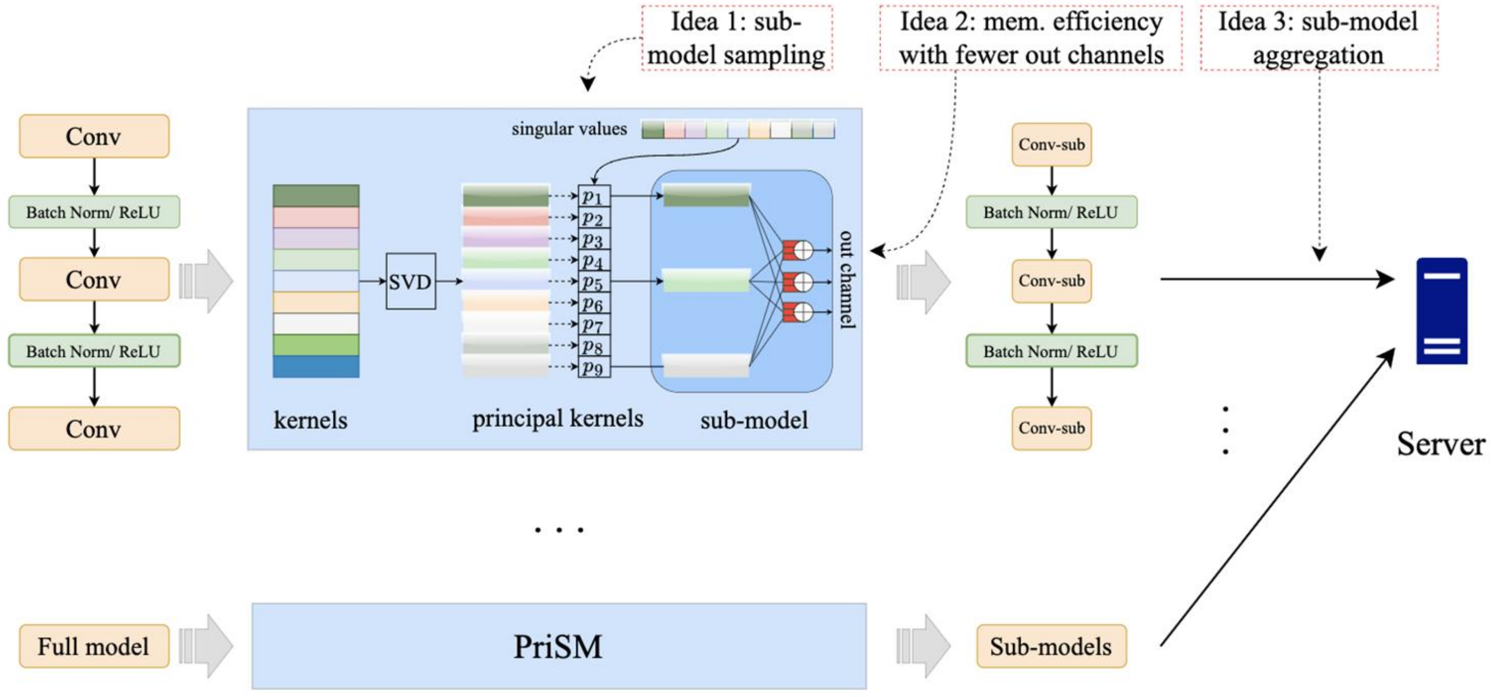
*client capabilities: 0.2*x*, 0.4*x*, 0.6*x*, 0.8*x

total comm. round: 1000

#local epoch/round: 2

- OrigDrop (FjORD): always select a fixed subset of *original* kernels.
- OrthDrop (FedHM): always select a fixed subset of *principal* kernels.

Proposed Solution **PriSM** Overview



Key idea for sub-model generation: **Dropout** in the principal domain!

Niu Y (*), **Prakash S (*)**, Kundu S, Lee S, Avestimehr S. "Overcoming resource constraints in federated learning: Large models can be trained with only weak clients." *Transactions on Machine Learning Research,* 2023.

Proposed Solution **PriSM** Overview

Centralized Training

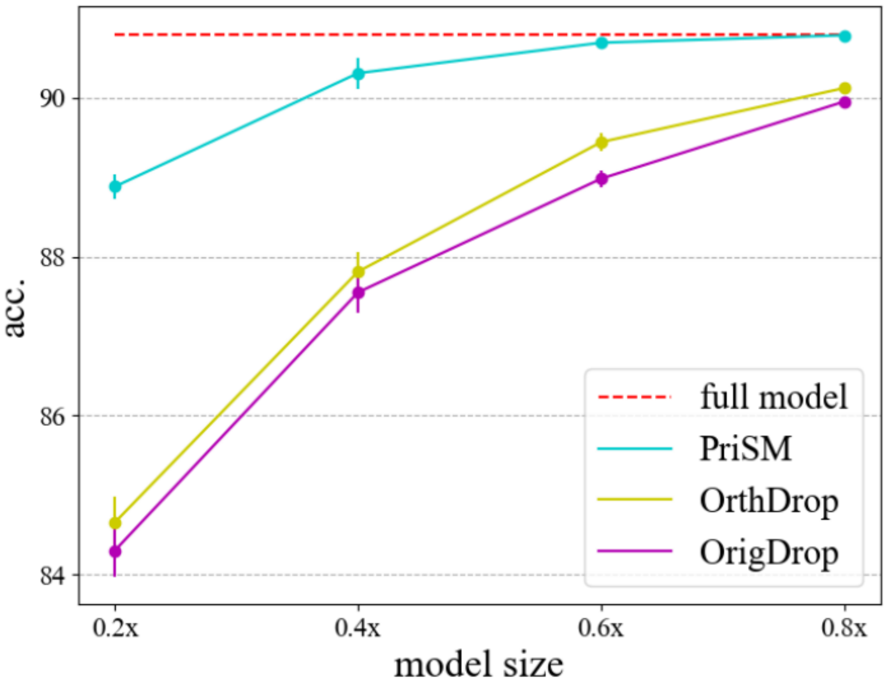
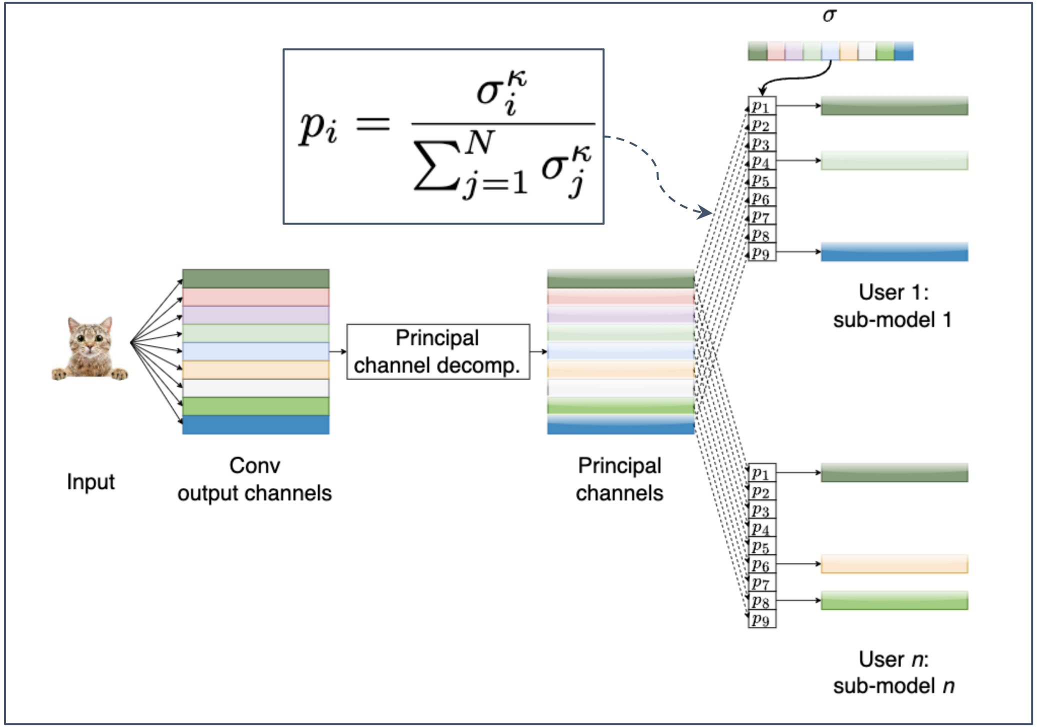


Figure 4: Accuracy of VGG11 on CIFAR-10 with different sub-models. PriSM delivers much better performance under very constrained settings (e.g., training with only 0.2× sub-models) compared to OrthDrop and OrigDrop.

Niu Y (*), **Prakash S (*)**, Kundu S, Lee S, Avestimehr S. "Overcoming resource constraints in federated learning: Large models can be trained with only weak clients." *Transactions on Machine Learning Research,* 2023.

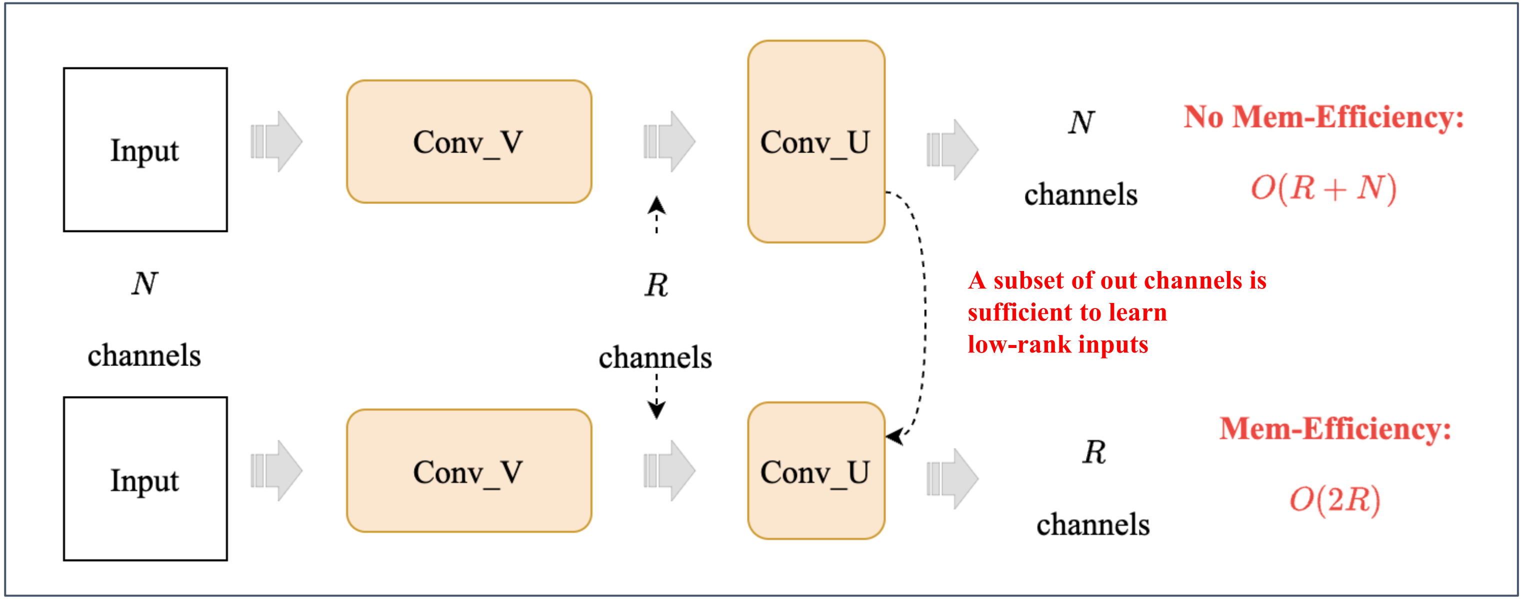
Idea 1: Sub-Model Creation via **Probabilistic** Kernel Sampling



Each client trains a different sub-model (⅓ of the full model in the example above)

→ computation reduction

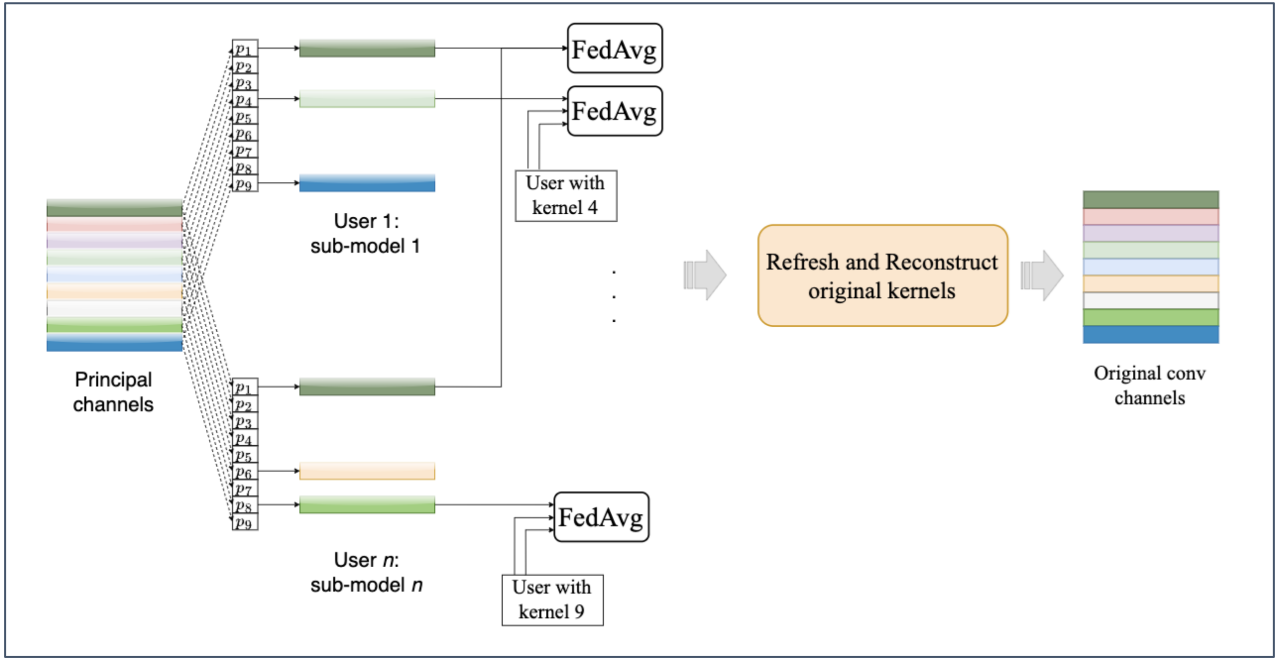
Idea 2: Memory-Efficient Sub-Model Training



Each client only keep a subset of output channel

→ computation & memory reduction

Idea 3: Sub-Model Aggregation

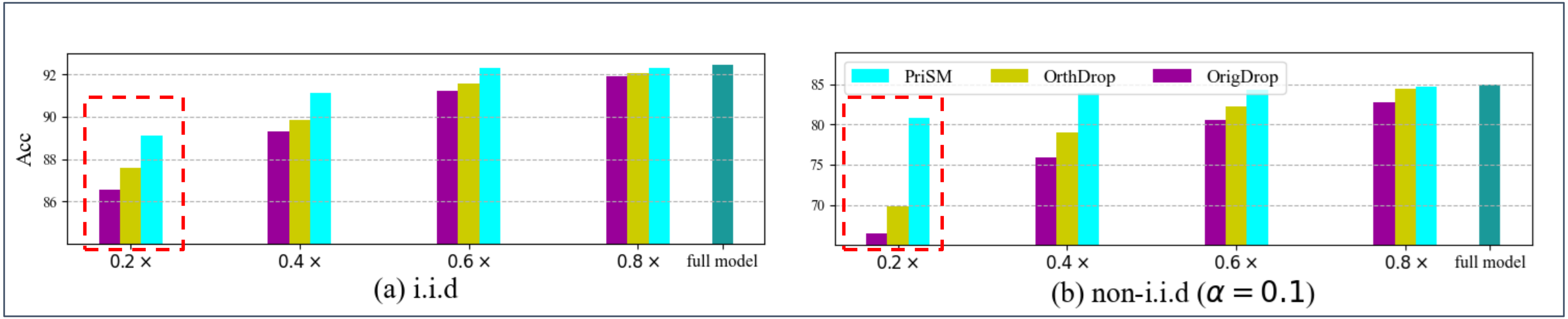


Each kernel get chance to be trained and aggregated

→ full-model capacity is preserved

Empirical Evaluations: ResNet-18/CIFAR-10

Empirical Evaluations: ResNet-18/CIFAR-10



model: ResNet-18

dataset: CIFAR-10 (i.i.d, non-i.i.d)

clients: 20 (active) / 100 (total)

*client capabilities: 0.2*x, *0.4*x, *0.6*x, *0.8*x

total comm. round: 1000

#local epoch/round: 2

Baselines:

- OrigDrop (FjORD): always select a fixed subset of *original* kernels.
- OrthDrop (FedHM): always select a fixed subset of *principal* kernels.

Empirical Evaluations: ResNet-18/CIFAR-10

Training time breakdown for ResNet-18 on CIFAR-10.

stage	sub-model create	local train	aggregate	SVD
time	3.27 s	36.82 s	0.11 s	0.96 s

model: ResNet-18

dataset: CIFAR-10 (i.i.d, non-i.i.d)

clients: 20 (active) / 100 (total)

*client capabilities: 0.2*x, *0.4*x, *0.6*x, *0.8*x

total comm. round: 1000

#local epoch/round: 2

Another Topic 1: Machine Unlearning

Data Leakage through models

- New privacy regulations (e.g., GDPR, PDPB) ensure **Right to be Forgotten**
- Permanent **data removal** is possible: remove it from the dataset
- **Is this enough?**

Another Topic 1: Machine Unlearning

Data Leakage through models

- New privacy regulations (e.g., GDPR, PDPB) ensure **Right to be Forgotten**
- Permanent **data removal** is possible: remove it from the dataset
- **Is this enough?**
- For example, deep neural networks are known to memorize training samples [Feldman et al., 2020]
  - Model inversion attacks [Fredrikson et al., 2015]

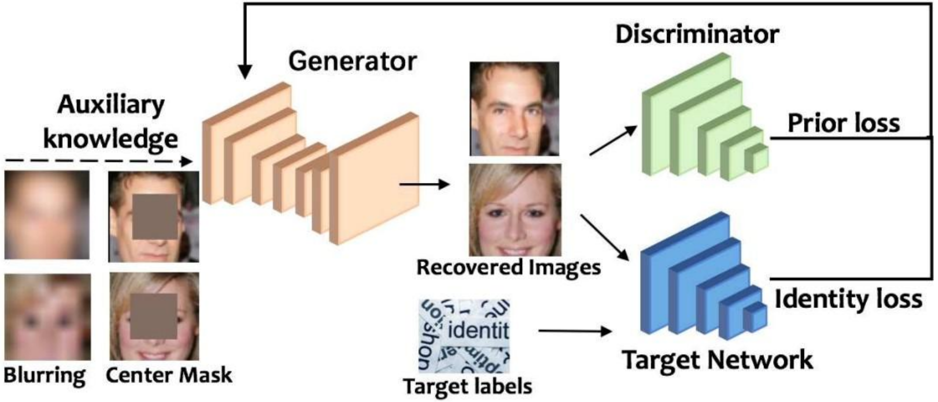


Figure: Generative model inversion attacker frameworks [Zhang et al., 2020], [Dèjá Vu, 2023].

Another Topic 1: Machine Unlearning

Data Leakage through models

- New privacy regulations (e.g., GDPR, PDPB) ensure **Right to be Forgotten**
- Permanent **data removal** is possible: remove it from the dataset
- **Is this enough?**
- For example, deep neural networks are known to memorize training samples
  - ➢ Model inversion attacks [Carlini et al., 2020]

Figure: Extracting Training Data from Large Language Models [Carlini et al., 2020].

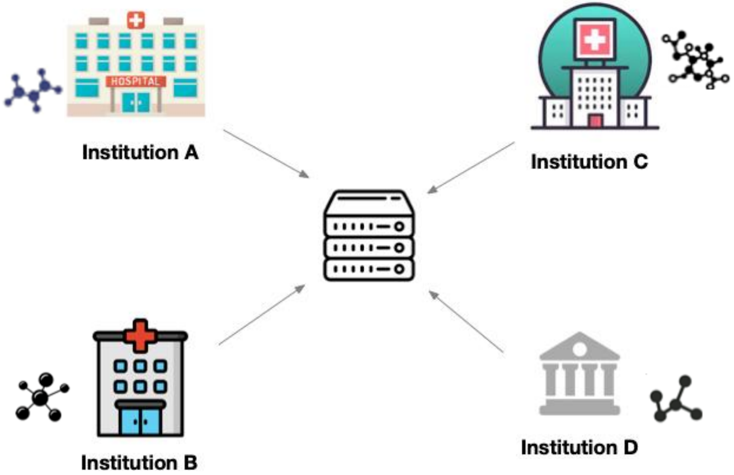
Another Topic 2: Hyperbolic Machine Learning

Genome sequences

Large

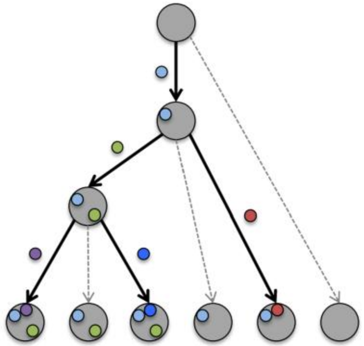
- E.g., Human genome 3 billion base pairs; ~750 MB of data

Private



Hierarchical

- E.g., Clonal Evolution Theory of Cancer [Nowell, 1976]



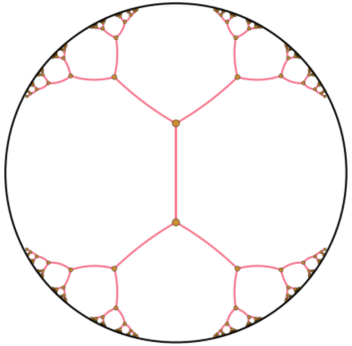
Phylogenetic Tree

How to do federated learning with high dimensional **hierarchical** data?

Another Topic 2: Hyperbolic Machine Learning

Hyperbolic space

Negatively curved space: Great for tree-like data



Poincare disc: Red curves are equidistant **geodesics**

- **Distance**

$$d(\boldsymbol{u}, \boldsymbol{v}) = \operatorname{arcosh}\left(1 + 2\frac{\|\boldsymbol{u}-\boldsymbol{v}\|^2}{(1-\|\boldsymbol{u}\|^2)(1-\|\boldsymbol{v}\|^2)}\right)$$

- **Linear hyperplane**

$$H_{\boldsymbol{w},\boldsymbol{p}} \triangleq \{\boldsymbol{x} \in \mathbb{B}^2 : \langle(-\boldsymbol{p} \oplus \boldsymbol{x}), \boldsymbol{w}\rangle = 0\}$$

$$\boldsymbol{x} \oplus \boldsymbol{y} = \frac{(1 + 2\langle \boldsymbol{x}, \boldsymbol{y}\rangle + \|\boldsymbol{y}\|^2)\boldsymbol{x} + (1 - \|\boldsymbol{x}\|^2)\boldsymbol{y}}{1 + 2\langle \boldsymbol{x}, \boldsymbol{y}\rangle + \|\boldsymbol{x}\|^2\|\boldsymbol{y}\|^2}, \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{B}^2$$

For FL, naive model aggregation is **not** applicable!

All Rivers Run to the Sea: Private Learning with Asymmetric Flows (CVPR 2024)

Yue Niu[1], Ramy E. Ali[2], **Saurav Prakash**[3], Salman Avestimehr[1]

[1]University of Southern California (USC)

[2]Samsung

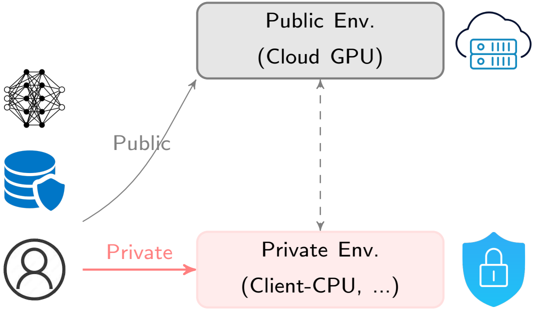
[3]Indian Institute of Technology Madras

January, 2024

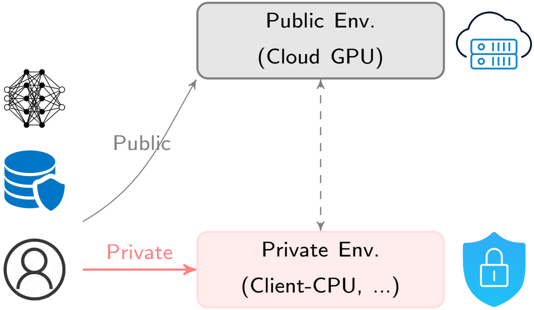
Outline

Outline

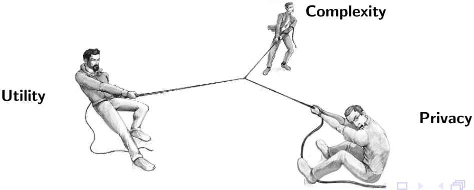
How to leverage cloud ML while ensuring privacy?



How to leverage cloud ML while ensuring privacy?

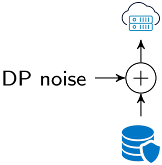


The Utility-Privacy-Complexity Trilemma



Privacy-Preserving ML Approaches

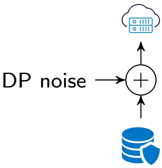
(Naive) DP-based ML



- Provable guarantee
- Severe accuracy drop

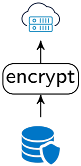
Privacy-Preserving ML Approaches

(Naive) DP-based ML



- Provable guarantee
- Severe accuracy drop

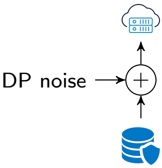
Crypto-based ML



- Strong protection
- High complexity

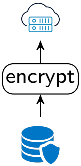
Privacy-Preserving ML Approaches

(Naive) DP-based ML



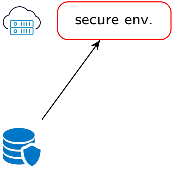
- Provable guarantee
- Severe accuracy drop

Crypto-based ML



- Strong protection
- High complexity

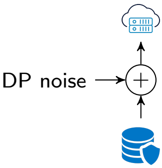
Secure Enclaves



- Hardware security
- Long running time

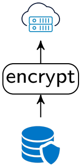
Privacy-Preserving ML Approaches

(Naive) DP-based ML



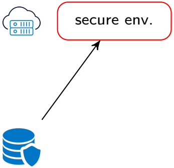
- Provable guarantee
- Severe accuracy drop

Crypto-based ML



- Strong protection
- High complexity

Secure Enclaves



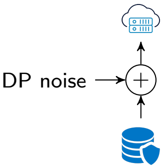
- Hardware security
- Long running time

Our work: Leverage both DP & Trusted hardware (local CPU, ...)

$\rightarrow$ Overcome accuracy drop of naive-DP & long running time of TEEs

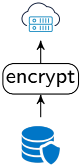
Privacy-Preserving ML Approaches

(Naive) DP-based ML



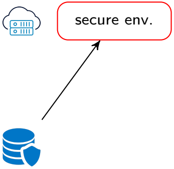
- Provable guarantee
- Severe accuracy drop

Crypto-based ML



- Strong protection
- High complexity

Secure Enclaves



- Hardware security
- Long running time

Related works leveraging TEEs

- Slalom'18: Inference only $\rightarrow$ This work: Inference and Training
- 3LegRace'21: Layerwise TEE-GPU communication $\rightarrow$ This work: No layer-wise communication

Outline

Delta: Private Learning with Asymmetric Flows

What does Delta do?

Decompose model & data into a low-dimensional part & a residual part

1. Lightweight model (client-side, TEEs, ...)
   - Fed with the low-dimensional information-sensitive part of the data
   - Confidential computing (no DP noise needed)

Delta: Private Learning with Asymmetric Flows

What does Delta do?

Decompose model & data into a low-dimensional part & a residual part

1. Lightweight model (client-side, TEEs, ...)
   - Fed with the low-dimensional information-sensitive part of the data
   - Confidential computing (no DP noise needed)
2. Large model (offloaded to cloud)
   - Fed with the quantized residual part of the data
   - The residual data is protected by a DP noise

What does Delta do?

Decompose model & data into a low-dimensional part & a residual part

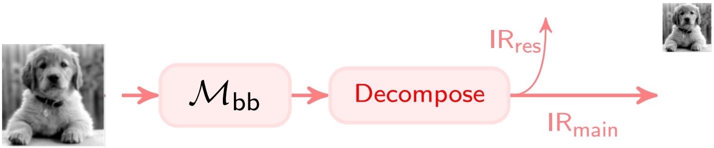
1. Lightweight model (client-side, TEEs, ...)
   - Fed with the low-dimensional information-sensitive part of the data
   - Confidential computing (no DP noise needed)
2. Large model (offloaded to cloud)
   - Fed with the quantized residual part of the data
   - The residual data is protected by a DP noise

$\Rightarrow$ Delta provides better utility-privacy trade-off than naive-DP methods

Delta Overview

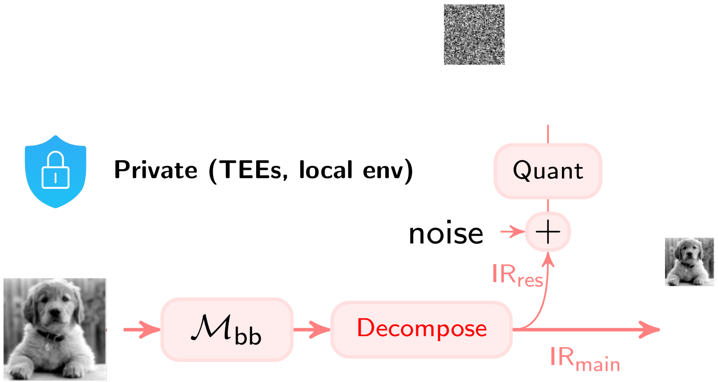
Forward Propagation: Asymmetric Data Decomposition

Private (TEEs, local env)



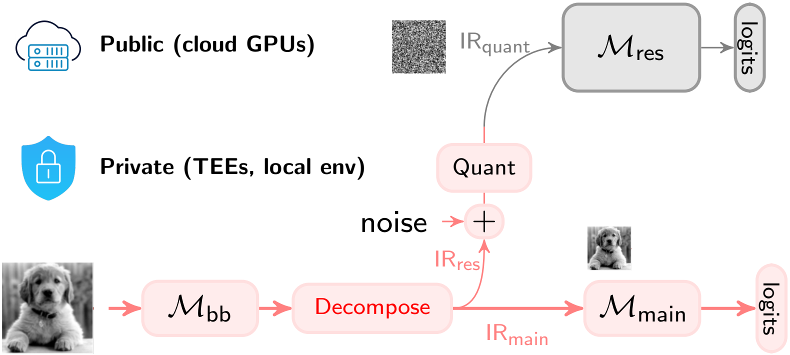
$\rightarrow$ To leverage the low-rank structure of the data

Forward Propagation: Perturbation & Binary Quantization



$\rightarrow$ To ensure privacy and reduce communication cost

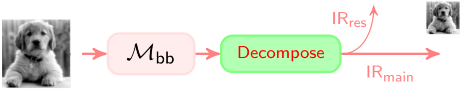
Forward Propagation: Model Decomposition



$\rightarrow$ To ensure low complexity in the private environment

Delta: Detailed Procedure

Asymmetric Data Decomposition



$\mathbf{IR} \xrightarrow{\text{SVD}} \begin{cases} \mathbf{IR}_{\text{SVD main}} \xrightarrow{\text{DCT}} \begin{cases} \mathbf{IR}_{\text{main}} \\ \mathbf{IR}_{\text{DCT res}} \end{cases} \\ \mathbf{IR}_{\text{SVD res}} \end{cases}$

$\mathbf{IR}_{\text{DCT res}} + \mathbf{IR}_{\text{SVD res}} \rightarrow \mathbf{IR}_{\text{res}}$

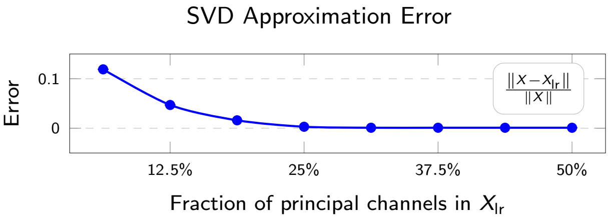
few channels & smaller feature size

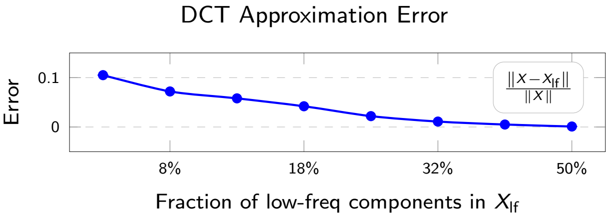
few info & large dimension

- SVD $\rightarrow$ asymmetric decomposition along channel dimension
- DCT $\rightarrow$ asymmetric decomposition along spatial dimension

Delta: Detailed Procedure

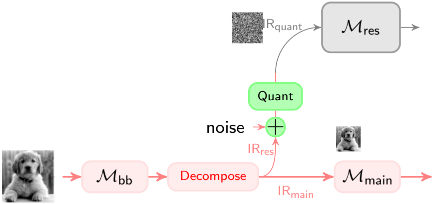
Why asymmetric decomposition?





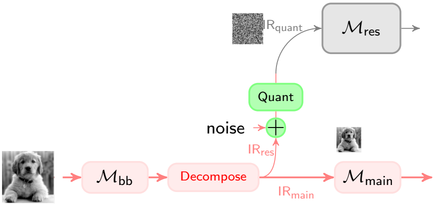
Delta: Detailed Procedure

Random Binary Quantization



$$\mathrm{IR}_{\mathrm{quant}}(\cdot) = \mathrm{BinQuant}(\mathrm{IR}_{\mathrm{noisy}}(\cdot)) = \begin{cases} 0 & \mathrm{IR}_{\mathrm{noisy}}(\cdot) < 0 \\ 1 & \mathrm{IR}_{\mathrm{noisy}}(\cdot) \geq 0 \end{cases}$$

Random Binary Quantization

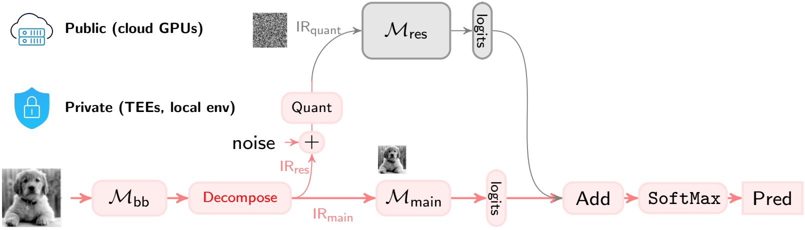


$$\mathrm{IR}_{\mathrm{quant}}(\cdot) = \mathrm{BinQuant}(\mathrm{IR}_{\mathrm{noisy}}(\cdot)) = \begin{cases} 0 & \mathrm{IR}_{\mathrm{noisy}}(\cdot) < 0 \\ 1 & \mathrm{IR}_{\mathrm{noisy}}(\cdot) \geq 0 \end{cases}$$

Theorem: `Delta` ensures that any operation in the public environment satisfy (ϵ, δ)-DP given noise $\mathcal{N}(0, p\Delta/\epsilon \cdot \sqrt{2\log(1.25/\delta)})$ and mini-batch size b, where $p = b/N$ is the sampling probability.

Delta: Detailed Procedure

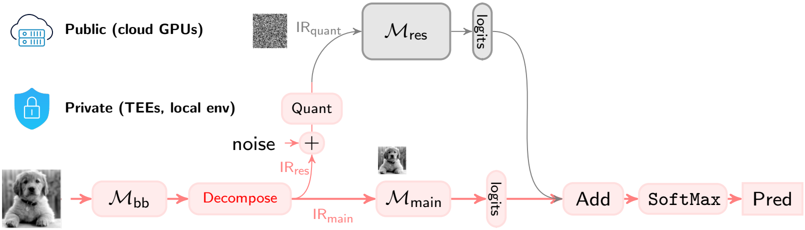
Private Backpropagation



$$\mathcal{M}_{\mathrm{main}} : \boldsymbol{o}_{\mathrm{tot}}(i) = \frac{e^{\boldsymbol{z}_{\mathrm{main}}(i)+\boldsymbol{z}_{\mathrm{res}}(i)}}{\sum_{j=1} e^{\boldsymbol{z}_{\mathrm{main}}(j)+\boldsymbol{z}_{\mathrm{res}}(j)}} \quad \text{for} \quad i = 1, \cdots, L$$

$$\mathcal{M}_{\mathrm{res}} : \boldsymbol{o}_{\mathrm{res}}(i) = \frac{e^{\boldsymbol{z}_{\mathrm{res}}(i)}}{\sum_{j=1} e^{\boldsymbol{z}_{\mathrm{res}}(j)}} \quad \text{for} \quad i = 1, \cdots, L,$$

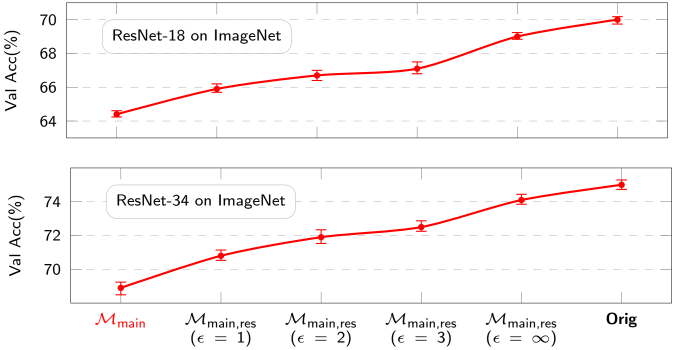
Delta: Full Picture



- Asymmetric data decomposition
- Efficient model design
- Random binary quantization
- Private backpropagation

Outline

Experiment Highlights: Model Utility



$\rightarrow$ Lightweight model achieves good accuracy, but still residuals are useful

Experiment Highlights: Model Utility

Setting: ResNet-18 with $\epsilon = 1$

	`Delta`: perturb IR_{res}	naive-DP: perturb IR
CIFAR-10	92.4%	69.6% ($\downarrow -22.8$)
CIFAR-100	71.4%	48.3% ($\downarrow -23.1$)
ImageNet	65.9%	34.4% ($\downarrow -31.5$)

$\rightarrow$ Delta improves accuracy by up to 31.5%

Experiment Highlights: Model Complexity

MACs of the modules in `Delta`

	$\mathcal{M}_{\text{bb}}+\mathcal{M}_{\text{main}}$	SVD	DCT	$\mathcal{M}_{\text{res}}$
ResNet-18	48.3 M	0.52 M	0.26 M	547M
ResNet-34	437 M	1.6 M	0.7 M	3.5G

- Small model $\mathcal{M}_{\text{main}}$ only costs 10% complexity of $\mathcal{M}_{\text{res}}$
- Costs of SVD and DCT are marginal

Experiment Highlights: Speedup

Running time with one single input

	Priv-only	3LegRace	Slalom	Delta
Train (ms/speedup)	1372	237 (6×)	-	62 (22×)
Inference (ms/speedup)	510	95 (5×)	84 (6×)	20 (25×)

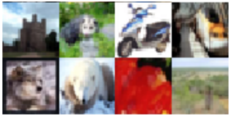
3LegRace [Niu, et al, PETs 2022]: layer-wise feature decomposition on linear layers
Slalom [Tramer, et al, ICLR 2019]: layer-wise computation distribution on linear layers

- Significant speedup compared to solely using private envs
- Faster compared to baselines due to reduced communication

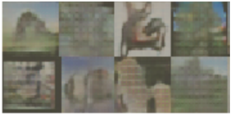
Experiment Highlights: Protection Against Attacks

Procedure: Train a GAN with the quantized residuals
Setting: ResNet-18, CIFAR-100

Against model inversion attack [`SecretRevealer`, CVPR'20]



Original samples



Reconstruction (no noise)

Reconstruction ($\epsilon = 1$)

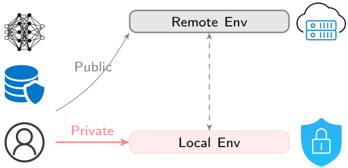
- Attack can succeed on certain samples (e.g., row 1, col 3)
- Random quantization provide further protection

Outline

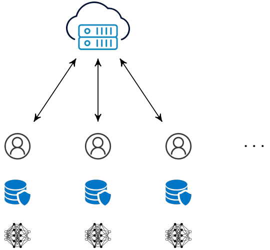
Future Works

Extend to More General Settings

User-Server Setting

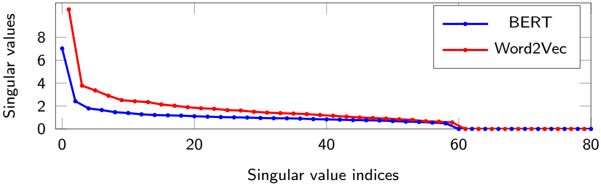


Federated Setting



Extend to LMs

LMs' embedding also exhibits a low-ranks structure



Original text (top) and approximated (bottom) text with 1/5 principal vectors.

Large Language Models are foundational machine learning models that use deep learning algorithms to process and understand natural language. These models are trained on massive amounts of text data to learn patterns and entity relationships in the language.

Large Language Models can perform many types of language tasks, such as translating languages, analyzing sentiments, chatbot conversations, and more. They can understand complex textual data, identify entities and relationships between them, and generate new text that **is** coherent and grammatically accurate.

Large Language Models are foundational machine learning models that use deep learning algorithms to process and understand natural language. These models are trained on massive amounts of text data to learn patterns and entity relationships in the language.

Large Language Models can perform many types of language tasks, such and translating languages, analyzing sentiments, chatbot conversations, and more. They can understand complex textual data, identify entities and relationships between them, and generate new text that **are** coherent and grammatically accurate.

Ongoing Work

How to leverage the low-rank structure of inputs in self-attention to reduce quadratic complexity?

- Preliminary results – Niu Y, **Prakash S**, Avestimehr S. "ATP: Enabling Fast LLM Serving via Attention on Top Principal Keys." *arXiv preprint arXiv:2403.02352*. 2024.

Thanks! Questions?