FedFetch: Faster Federated Learning with Adaptive Downstream Prefetching

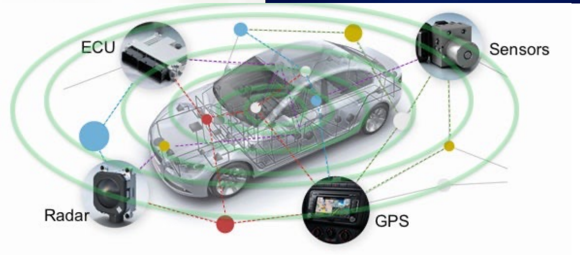
Qifan (Eric) Yan, Andrew Liu, Shiqi He, Mathias Lécuyer, Ivan Beschastnikh



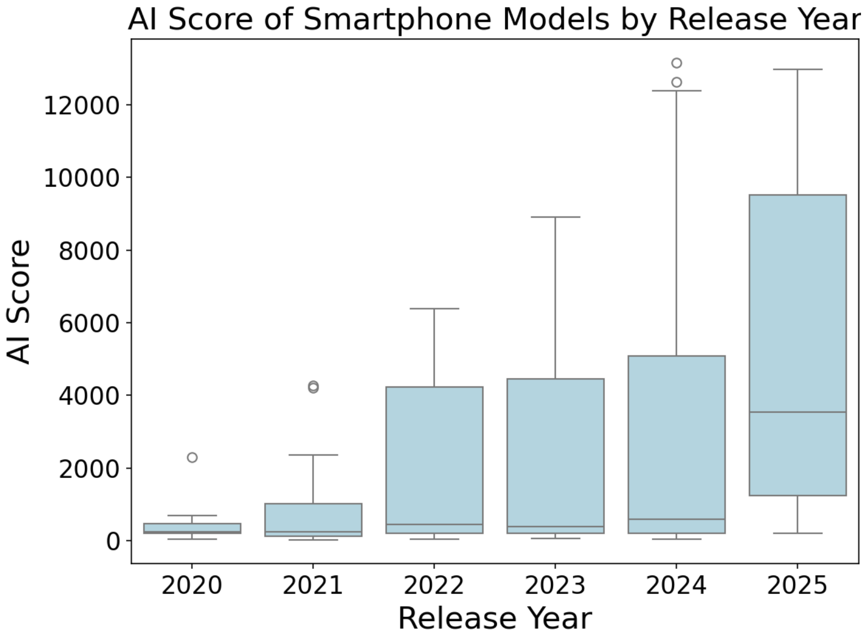


Edge Devices and Machine Learning (ML)

- Generate massive private data



- Increasing ML capabilities



Based on https://ai-benchmark.com/ranking.html

Federated Learning[1]**:** ***ML training*** **with** ***edge devices***

[1] McMahan et al. Communication-Efficient Learning of Deep Networks from Decentralized Data

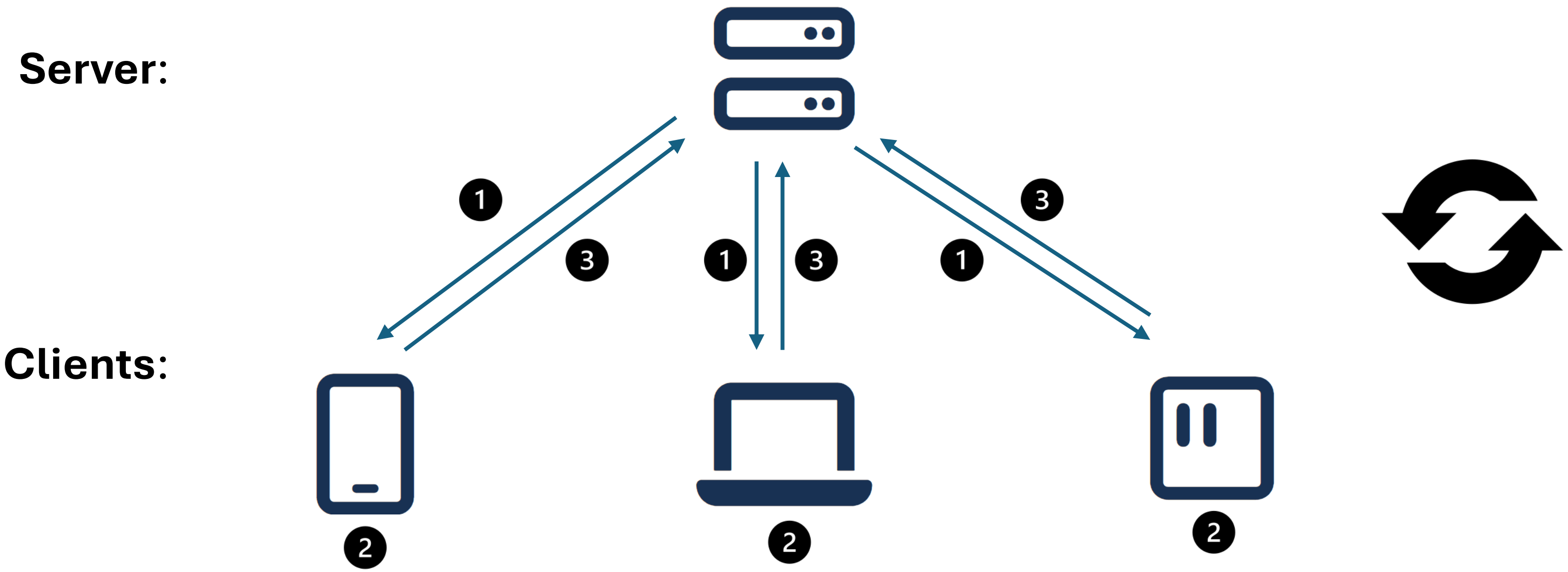
Cross-Device Federated Learning (FL)

- **Clients**
  - are **edge devices** such as phones, laptops, IoT device, intelligent vehicles etc.
  - collaboratively train models under coordination of a **server**

  ❶ sync local model with server's latest model

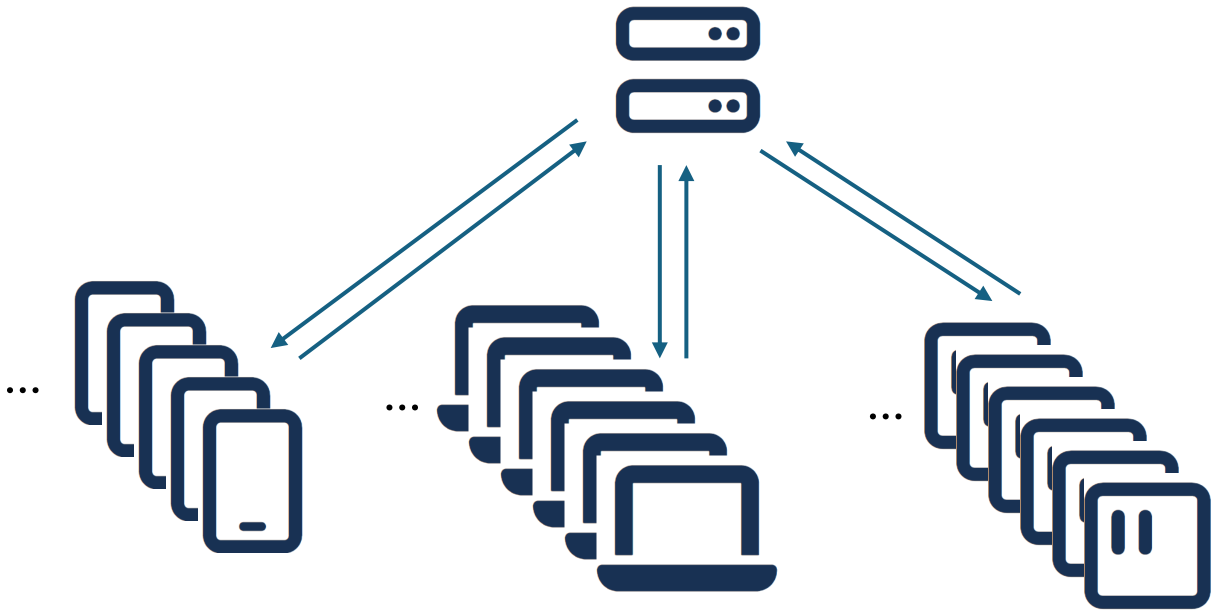
  ❷ perform training steps with optimizer (e.g., SGD) on **local** data

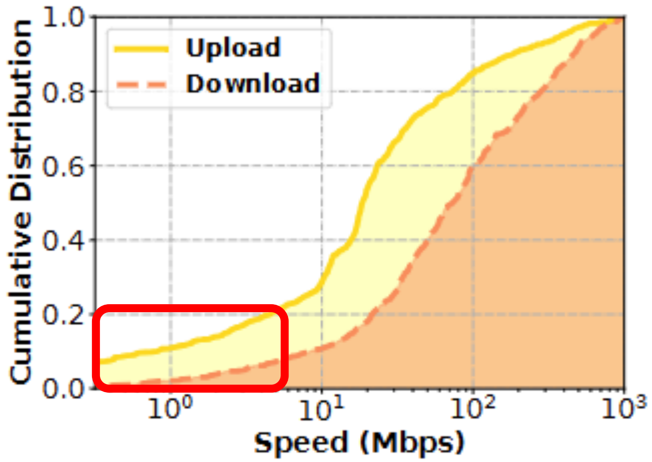
  ❸ transmit model updates to get next round's model through server aggregation



Challenges in Scaling FL

- **Huge** number of clients (production jobs see 10M daily active clients[1])
- System and statistical heterogeneity of clients leading to straggler effect



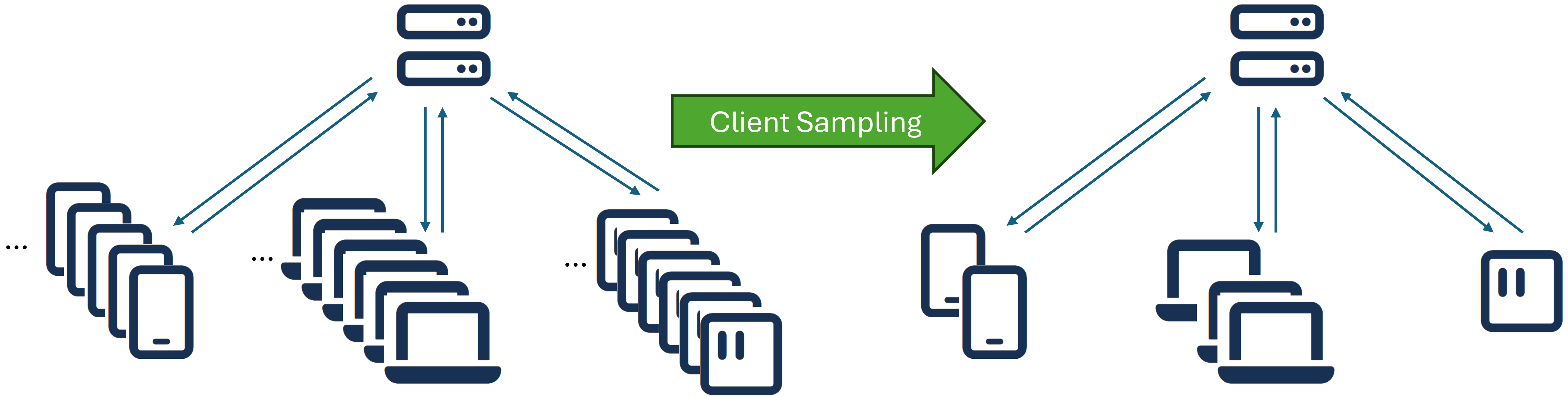


(a) CDF of edge device download and upload bandwidth distribution for North America in Jan 2024 [25].

[1] Bonawitz et al. Towards Federated Learning at Scale: System Design. 2019

Client Sampling

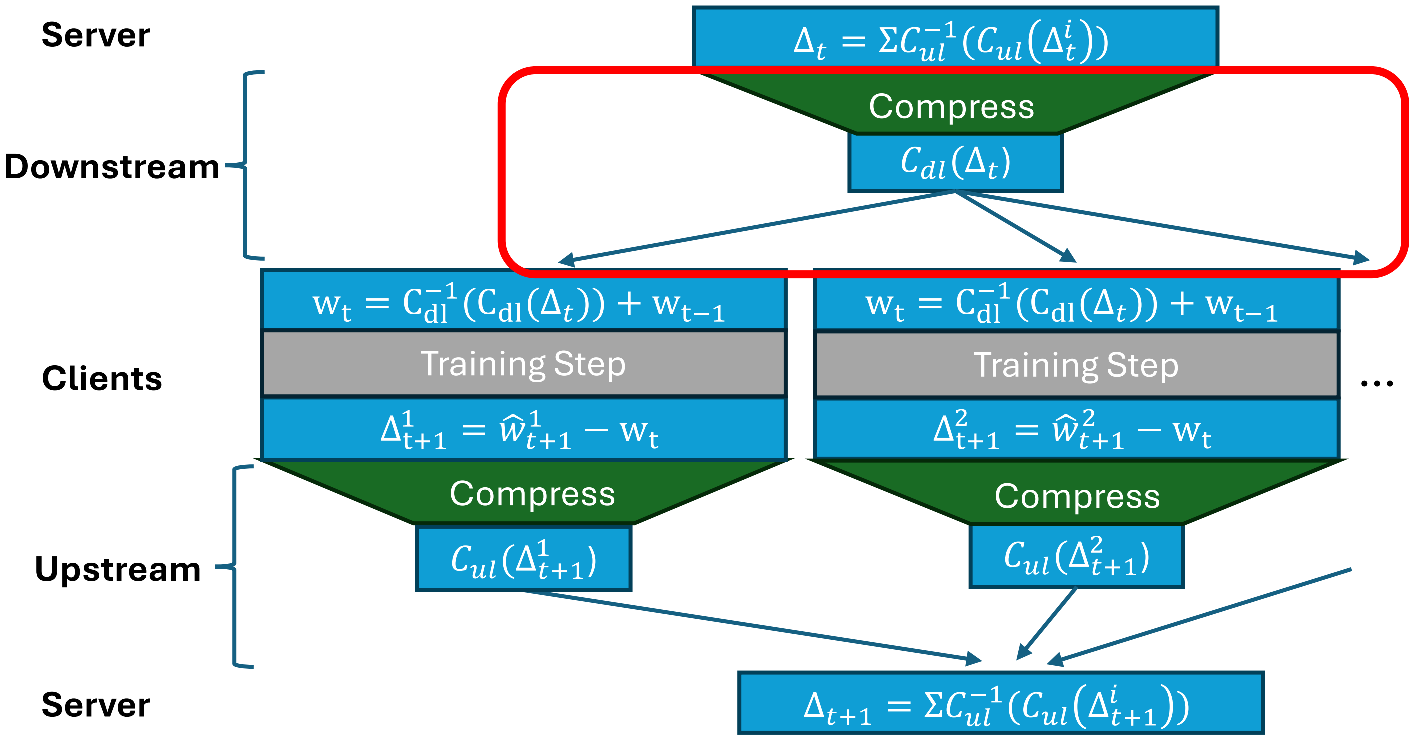
- Chooses K clients from a population of N clients for **partial participation**[1]
- Generally, reduces communication load in the network and on the server
  - Specific methods may have other benefits[2,3]



[1] McMahan et al. Communication-Efficient Learning of Deep Networks from Decentralized Data. 2017
[2] He et al. GlueFL: Reconciling Client Sampling and Model Masking for Bandwidth Efficient Federated Learning. 2023
[3] Lai et al. Oort: Efficient Federated Learning via Guided Participant Selection. 2021

Update Compression

- Compress updates with
  - masking/sparsification[1,2]
  - quantization[3,4,5]
  - low-rank matix factorization[6]
  - and more!
- **Reduce communication costs** but with more training rounds and potential accuracy drop
- Most works address upstream (client → server) compression with C_{ul}
- Few works tackle **downstream (server → client) compression** with $\boldsymbol{C_{dl}}$



[1] He et al. GlueFL: Reconciling Client Sampling and Model Masking for Bandwidth Efficient Federated Learning. 2023
[2] Sattler et al. Robust and Communication-Efficient Federated Learning from Non-IID Data. 2019
[3] Alistar et al. QSGD: Communication-Efficient SGD via Gradient Quantization and Encoding. 2017
[4] Dorfman et al. DoCoFL: Downlink Compression for Cross-Device Federated Learning. 2023
[5] Amiri et al. Federated Learning With Quantized Global Model Updates. 2020
[6] Vogels et al. PowerSGD: Practical Low-Rank Gradient Compression for Distributed Optimization. 2019

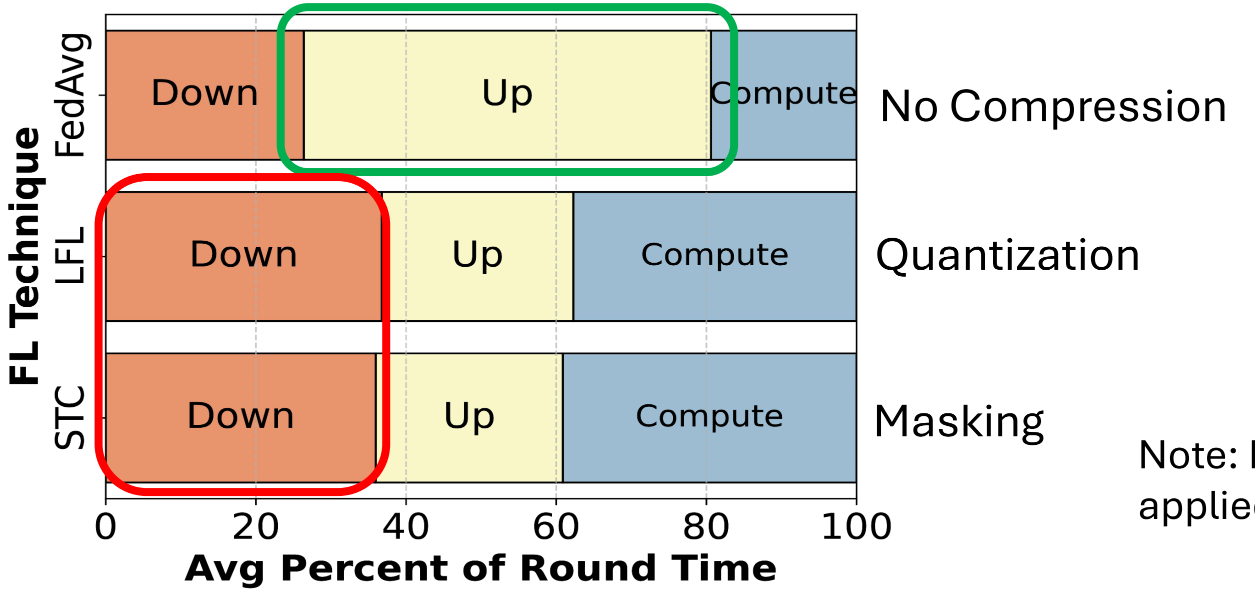
Incompatibility of Sampling and Compression

- Upstream was previously the bottleneck but have become less impactful
- Under client sampling, update compression cannot effectively reduce **downstream (server → client) communication costs**

Downstream is the **new bottleneck!**

- Our prior work **GlueFL**[1] explores this specifically for sampling + masking

FedFetch generalizes the problem

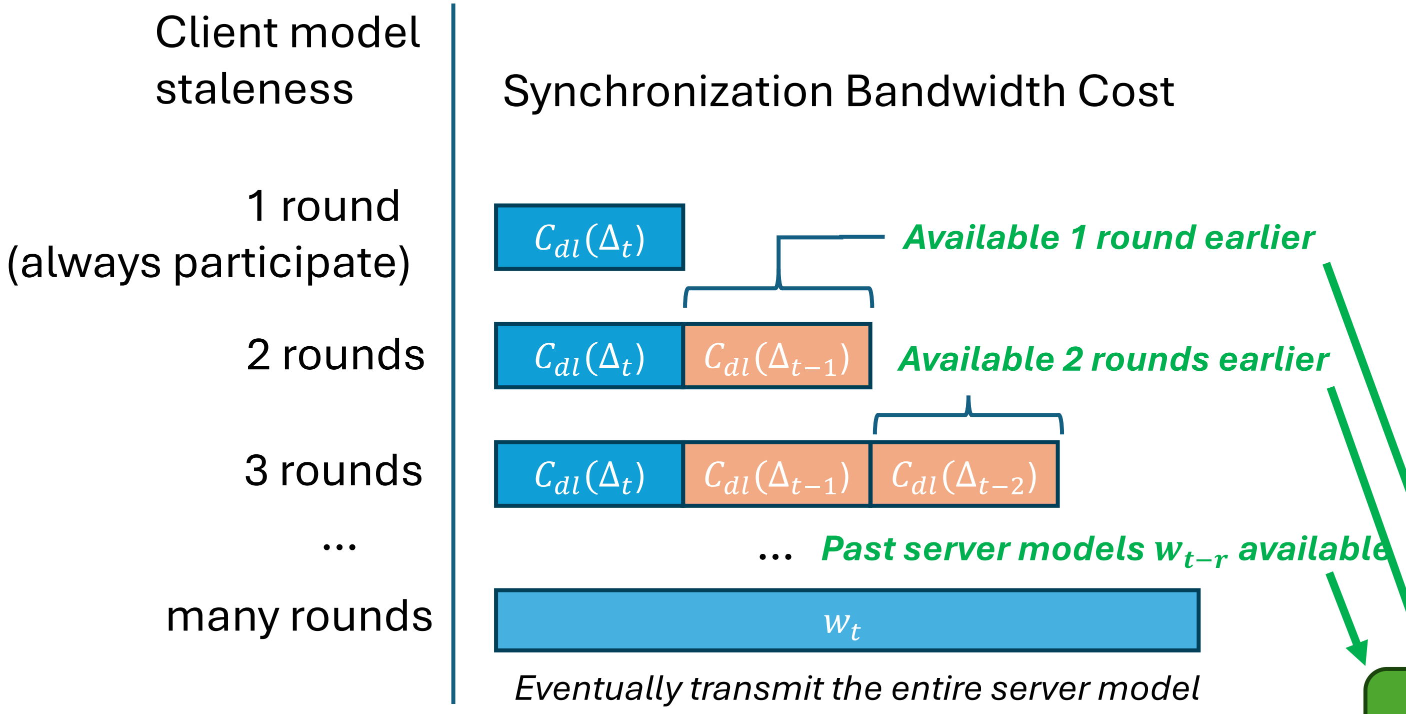


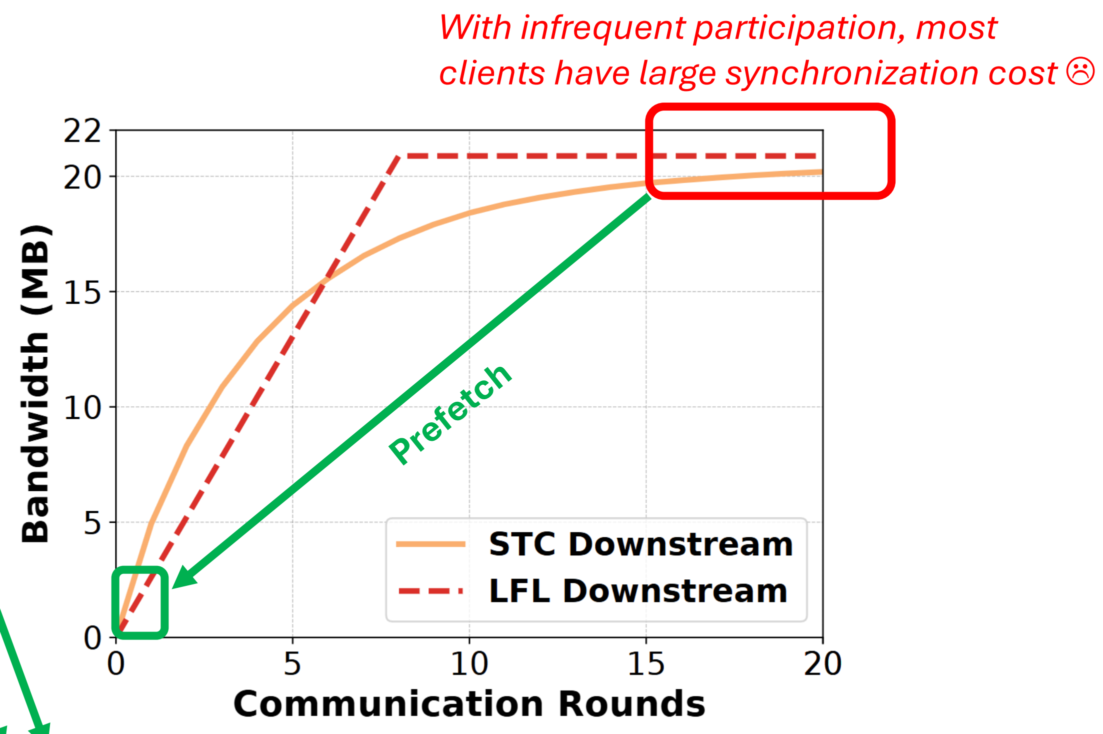
Note: No optimizations applied to compute

[1] He et al. GlueFL: Reconciling Client Sampling and Model Masking for Bandwidth Efficient Federated Learning. 2023

Client Model Staleness is the Culprit

- Client models are increasingly **stale** with every round of non-participation
- Downstream synchronization costs **grow with increasing staleness**

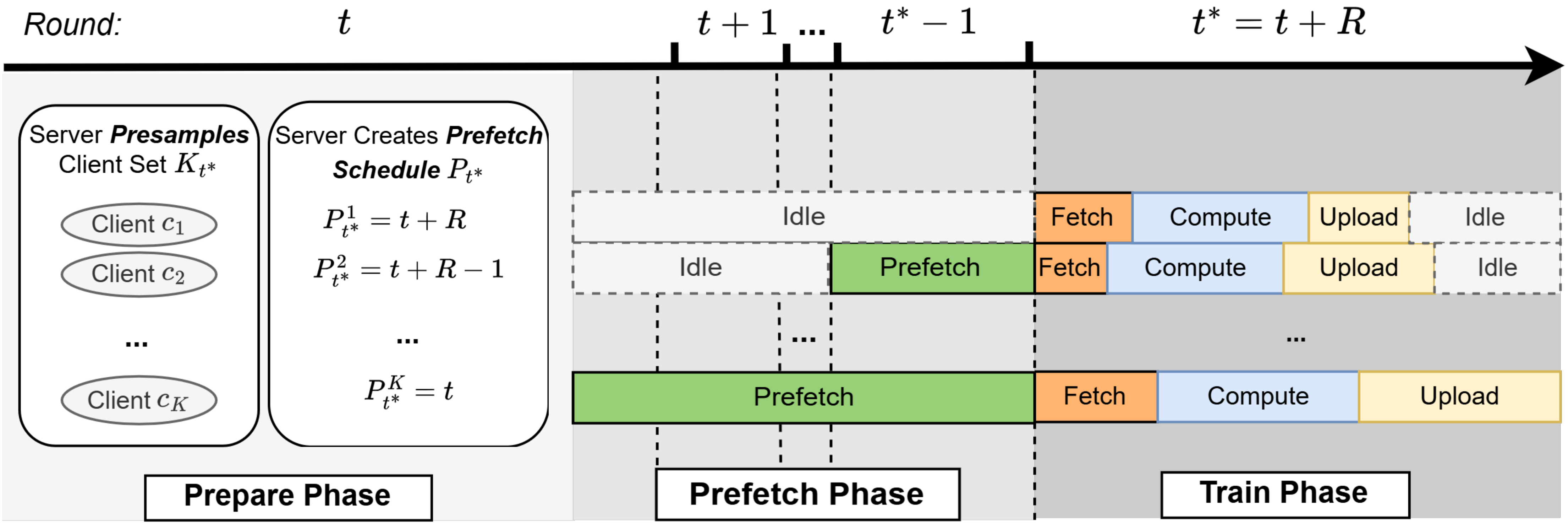




Can we **prefetch** these to reduce client model staleness?

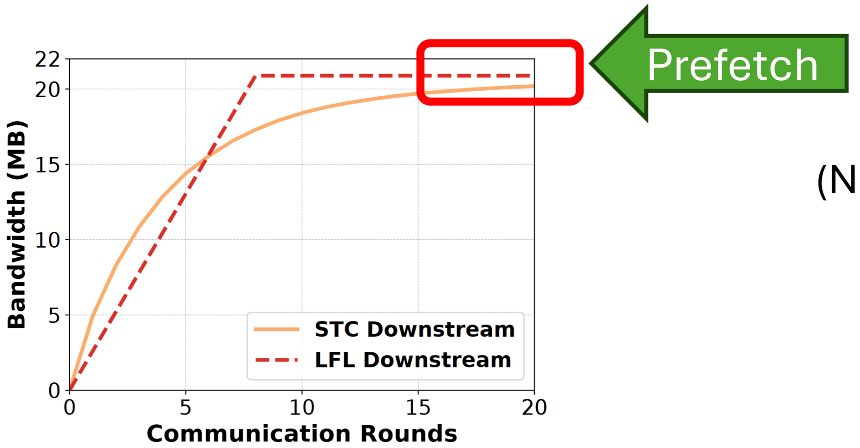
FedFetch Design

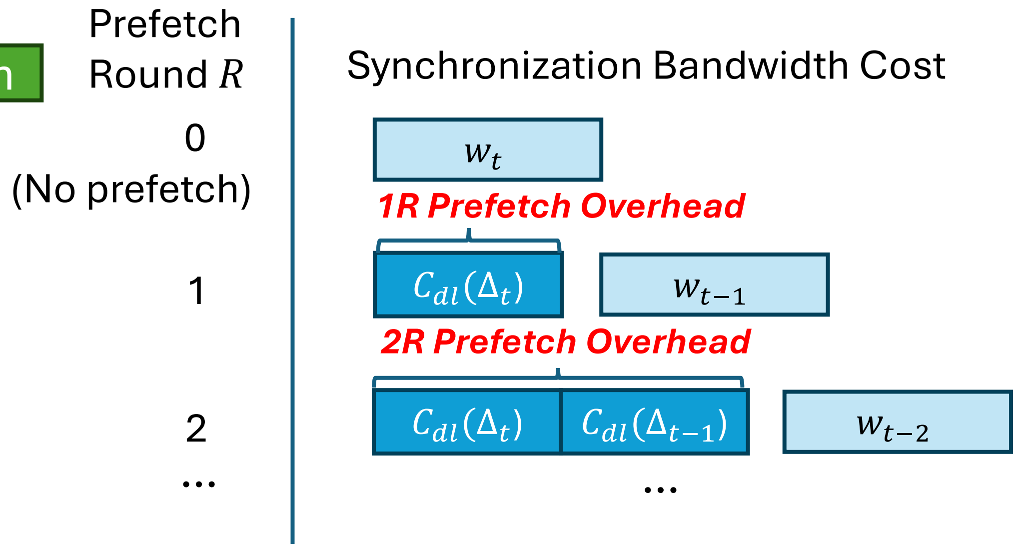
- **Prepare phase:** presample clients to get a stable participation schedule
- **Prefetch phase:** clients download in advance to accelerate Train phase
- **Train phase:** same as standard FL



FedFetch Prepare Phase

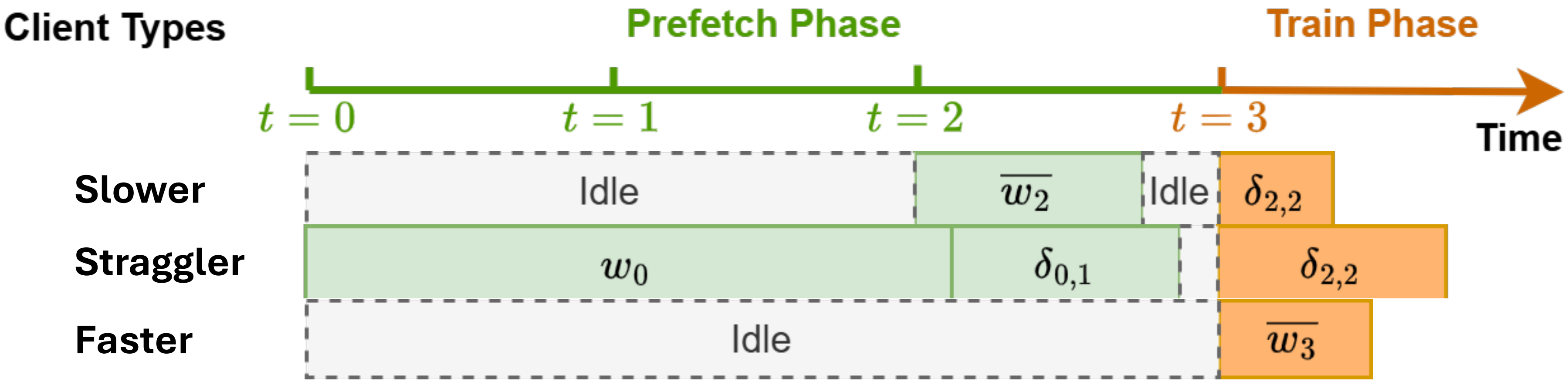
- **Presampling:**
  - Run client sampling function $\boldsymbol{R}$ rounds in advance
  - Typically, **no modifications** to the client sampling function
- **Prefetch Scheduling:**
  - Prefetching comes at a cost of **additional bandwidth overhead** and **grows with $\boldsymbol{R}$**
  - Scheduling to **pick $\boldsymbol{R}$ per-client** for **best time/bandwidth-to-accuracy performance**
  - See paper for prefetch scheduling design





FedFetch Prefetch Phase

- Clients follow **prefetch schedule** (from Prepare phase)
- **Slower** clients, especially **stragglers**, prefetch to save time in Train phase
- **Faster** clients exempted from prefetching to save bandwidth

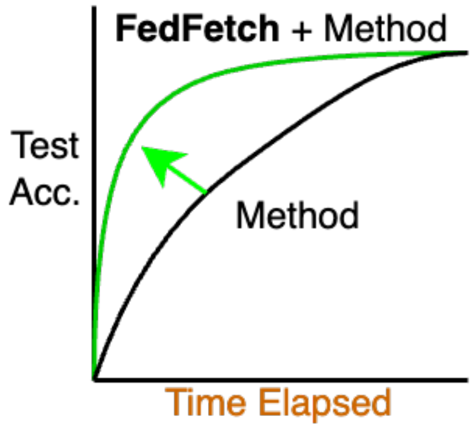


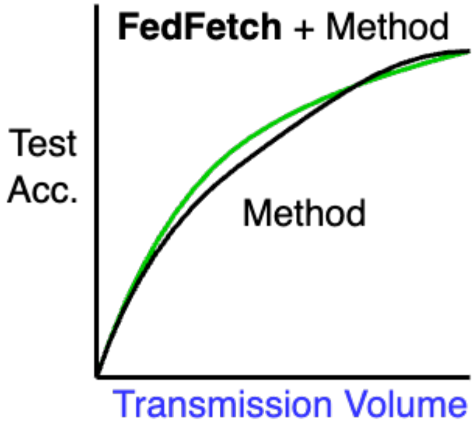
Evaluation Objectives

Q1: **Compatibility** with **client sampling** and **update compression** methods

Method vs **FedFetch +** Method

Q2: Impact on **time**, **bandwidth**, and **accuracy**



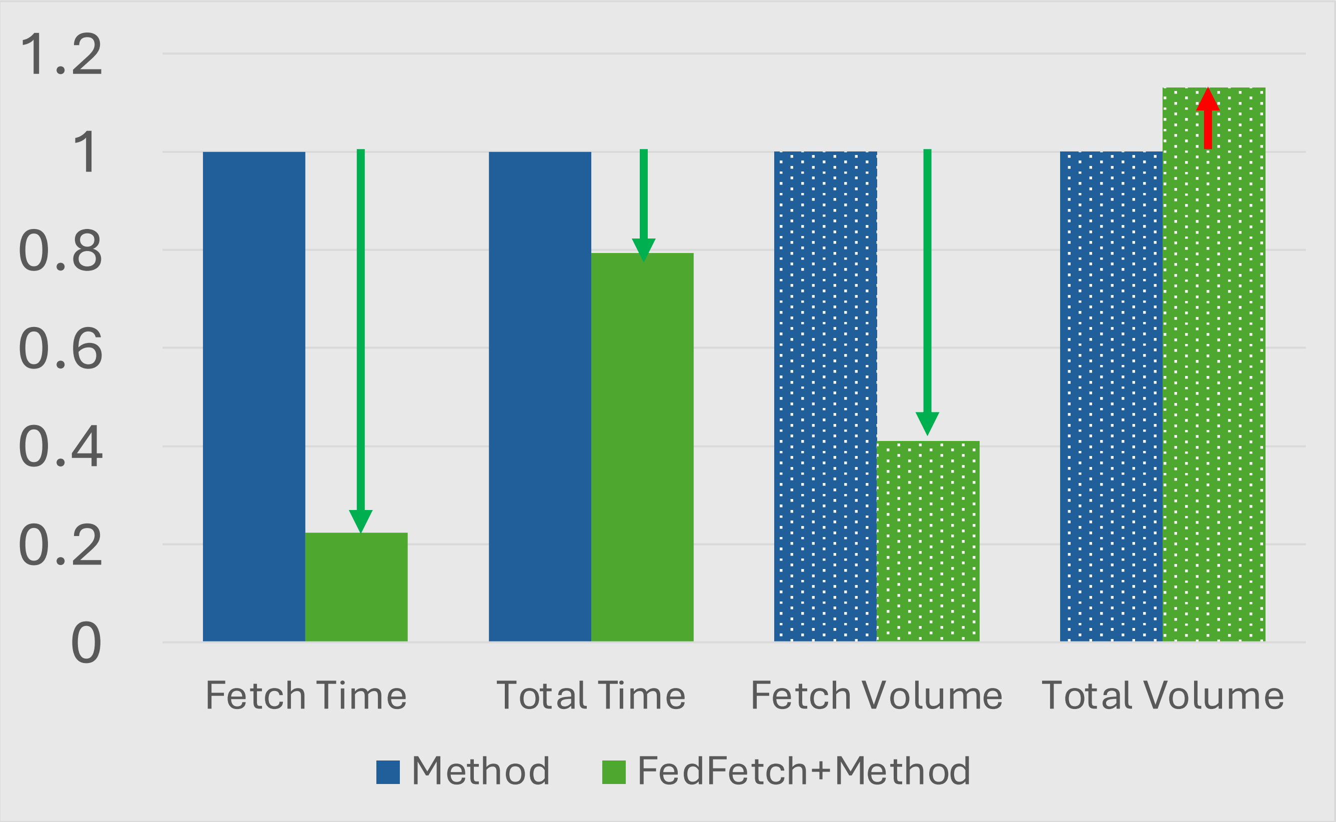


Q3: Impact of **hyperparameter** (full results in paper)

Q4: Impact of **client availability and overcommitment** (full results in paper)

Main Performance Evaluation

- Combined with 7 techniques(including GlueFL)
- Speeds up fetch (blocking download) time by **4.49x**
- Reduces training time by **1.26x**
- Incurs extra bandwidth of 13%, shifts from fetch (download) to prefetch

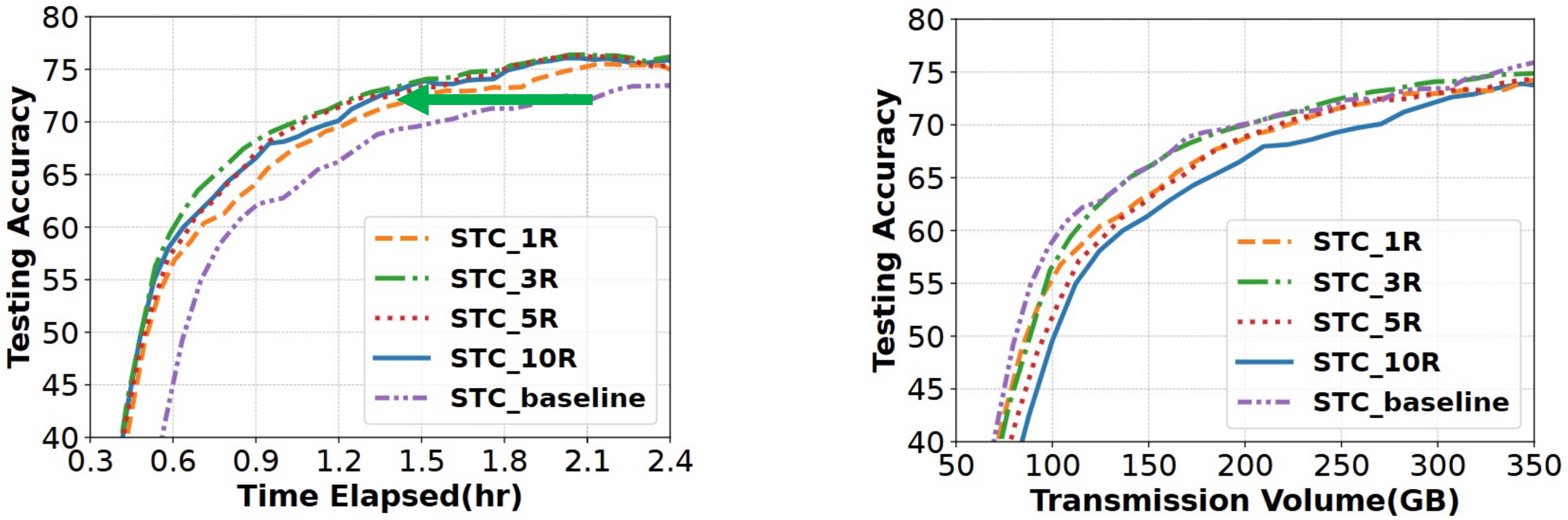


		...age Trg 68%		
			FV	**TV**
Baseline			10.8	19.1
Masking		7	11.5	**15.0**
		0	**8.85**	18.1
		5	14.0	**20.7**
	Fe	**6**	**8.84**	24.6
Quantization		8	10.8	**12.0**
	F	**7**	**4.45**	13.7
		5	10.4	**11.6**
		9	**4.34**	13.7
		4	10.2	**11.7**
	F	**2**	**5.48**	14.1
Low-rank		1	8.01	**9.24**
	FedFetc	**3**	**4.28**	11.0
Quantization (Full model)		6	**1.15**	5.52
	Fed	7	2.50	**5.38**

FT: Fetch Time
TT: Total Time
FV: Fetch Volume
TV: Total Volume

Hyperparameter Evaluation

- The presample/prefetch round $\boldsymbol{R}$ is the main hyperparameter of FedFetch
- FedFetch + STC (STC_1R, STC_3R, STC_5R, STC_10R)[1] consistently perform better than STC by itself (STC_baseline) in terms of **time-to-accuracy**
- Small Rs have similar bandwidth-to-accuracy performance as STC by itself
- We recommend a small $R = 3$ work for better time/bandwidth-to-accuracy



[2] Sattler et al. Robust and Communication-Efficient Federated Learning from Non-IID Data. 2019

Summary of Contributions

- **FedFetch**
  - reveals conflicts between **client sampling** and **update compression**
  - accelerates **downstream communication** by **4.49x** with low bandwidth overhead
  - is a **general design** compatible with many sampling and compression techniques
  - introduces new research direction of **short-term stable client participation schedules**

- Code available at https://github.com/DistributedML/FedFetch
  - as a FedScale[1] module using real client bandwidth[2], compute performance[3], online/offline behavior[4] data

Thank you!

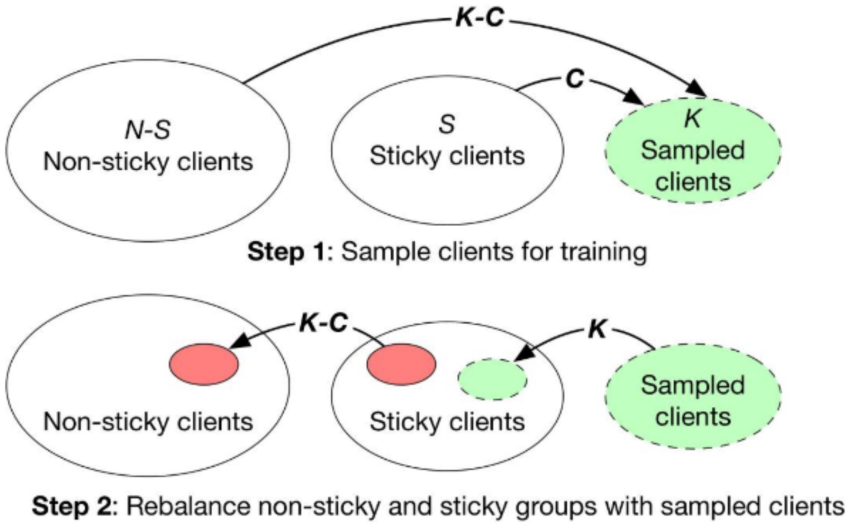
SYSTOPIA

[1] Lai et al. FedScale: Benchmarking Model and System Performance of Federated Learning at Scale. 2022
[2] https://www.measurementlab.net/
[3] https://ai-benchmark.com/index.html
[4] Yang et al. Characterizing impacts of heterogeneity in federated learning upon large-scale smartphone data. 2021

Extra slides

Presampling and Client Sampling

- Presampling means that client sampling cannot use information available at start of a client's training round
- Example: Sticky sampling from GlueFL[1]
  - strongly benefits from recent client participation schedules



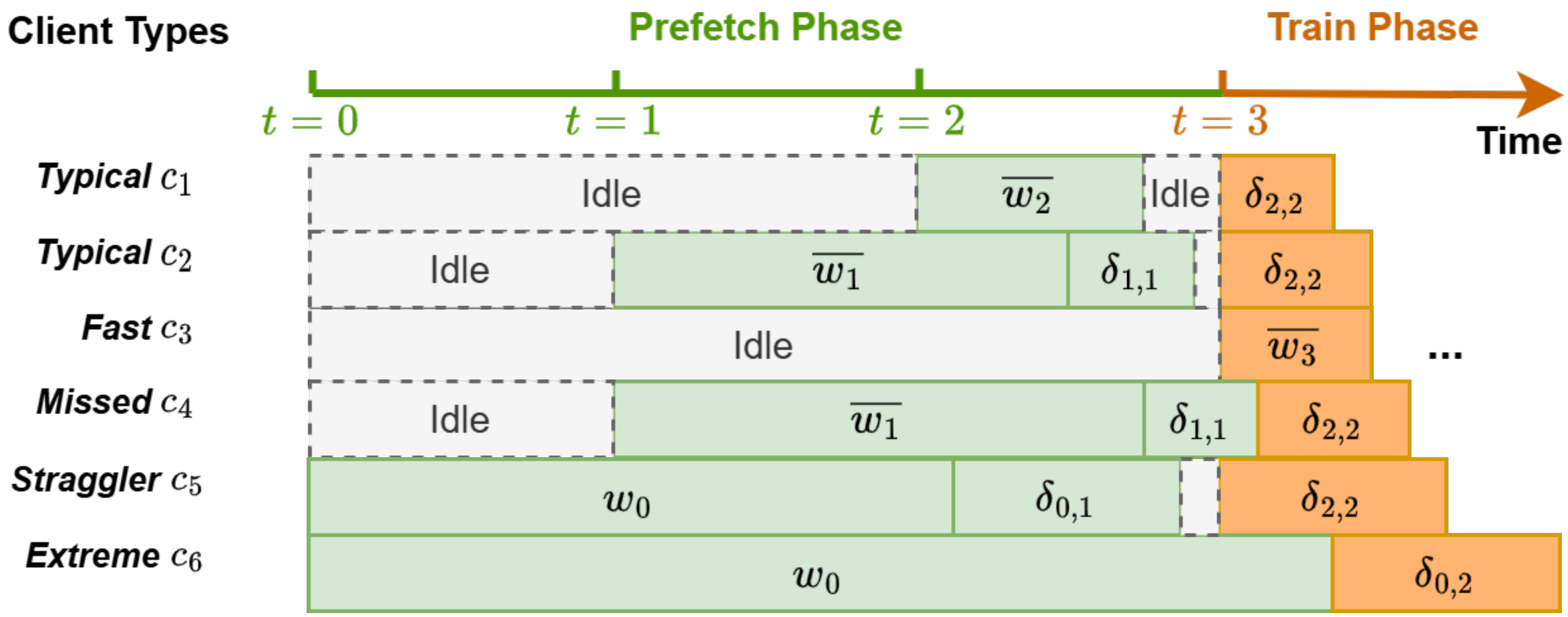
- However, **FedFetch** + GlueFL is 1.13x faster than GlueFL by itself

[1] He et al. GlueFL: Reconciling Client Sampling and Model Masking for Bandwidth Efficient Federated Learning

Full Result Table

		FEMNIST Trg 75%				*Google Speech Trg 61%*				*OpenImage Trg 68%*			
		FT	**TT**	**FV**	**TV**	**FT**	**TT**	**FV**	**TV**	**FT**	**TT**	**FV**	**TV**
Baseline	FedAvg	0.64	2.56	1.56	2.76	5.54	21.7	12.2	21.6	1.14	5.4	10.8	19.1
Masking	STC	0.85	2.46	2.58	**3.07**	9.96	25.0	13.7	**18.0**	0.97	4.47	11.5	**15.0**
	FedFetch + STC	**0.21**	**1.67**	**1.31**	3.14	**2.71**	**13.4**	**10.4**	21.6	**0.44**	**4.00**	**8.85**	18.1
	GlueFL	0.30	2.11	2.58	**3.32**	2.28	16.7	12.9	**18.9**	0.38	2.55	14.0	**20.7**
	FedFetch + GlueFL	**0.18**	**1.84**	**1.49**	4.01	**1.26**	**14.3**	**11.2**	25.0	**0.23**	**2.36**	**8.84**	24.6
Quantization	QSGD	0.46	1.35	1.56	**1.73**	3.41	8.62	10.49	**11.6**	0.72	3.68	10.8	**12.0**
	FedFetch + QSGD	**0.06**	**0.90**	**0.58**	1.83	**0.84**	**6.26**	**4.87**	14.3	**0.13**	**3.07**	**4.45**	13.7
	LFL	0.43	1.27	1.45	**1.60**	3.03	10.9	10.78	**11.9**	0.67	3.55	10.4	**11.6**
	FedFetch + LFL	**0.06**	**0.90**	**0.58**	1.83	**0.58**	**8.88**	**4.7**	14.7	**0.12**	**3.09**	**4.34**	13.7
	EDEN	0.42	1.26	1.43	**1.60**	4.02	12.7	11.7	**13.7**	0.74	3.54	10.2	**11.7**
	FedFetch + EDEN	**0.06**	**0.89**	**0.60**	1.75	**1.12**	**10.1**	**6.92**	16.5	**0.16**	**3.12**	**5.48**	14.1
Low-rank	POWERSGD	1.49	4.23	5.25	**5.62**	6.87	26.7	28.1	**29.8**	0.58	2.81	8.01	**9.24**
	FedFetch + POWERSGD	**0.14**	**3.02**	**1.47**	6.54	**0.67**	**21.0**	**6.89**	35.5	**0.11**	**2.43**	**4.28**	11.0
Quantization (Full model)	DoCoFL	0.04	0.96	**0.13**	0.72	**0.48**	8.79	**1.45**	6.97	0.12	3.26	**1.15**	5.52
	FedFetch + DoCoFL	0.04	**0.93**	0.28	**0.56**	0.50	**8.47**	2.95	**6.02**	0.12	**2.87**	2.50	**5.38**

Full Prefetch Process



Downstream and Upstream Bandwidth Across Rounds