Efficient Federated Learning

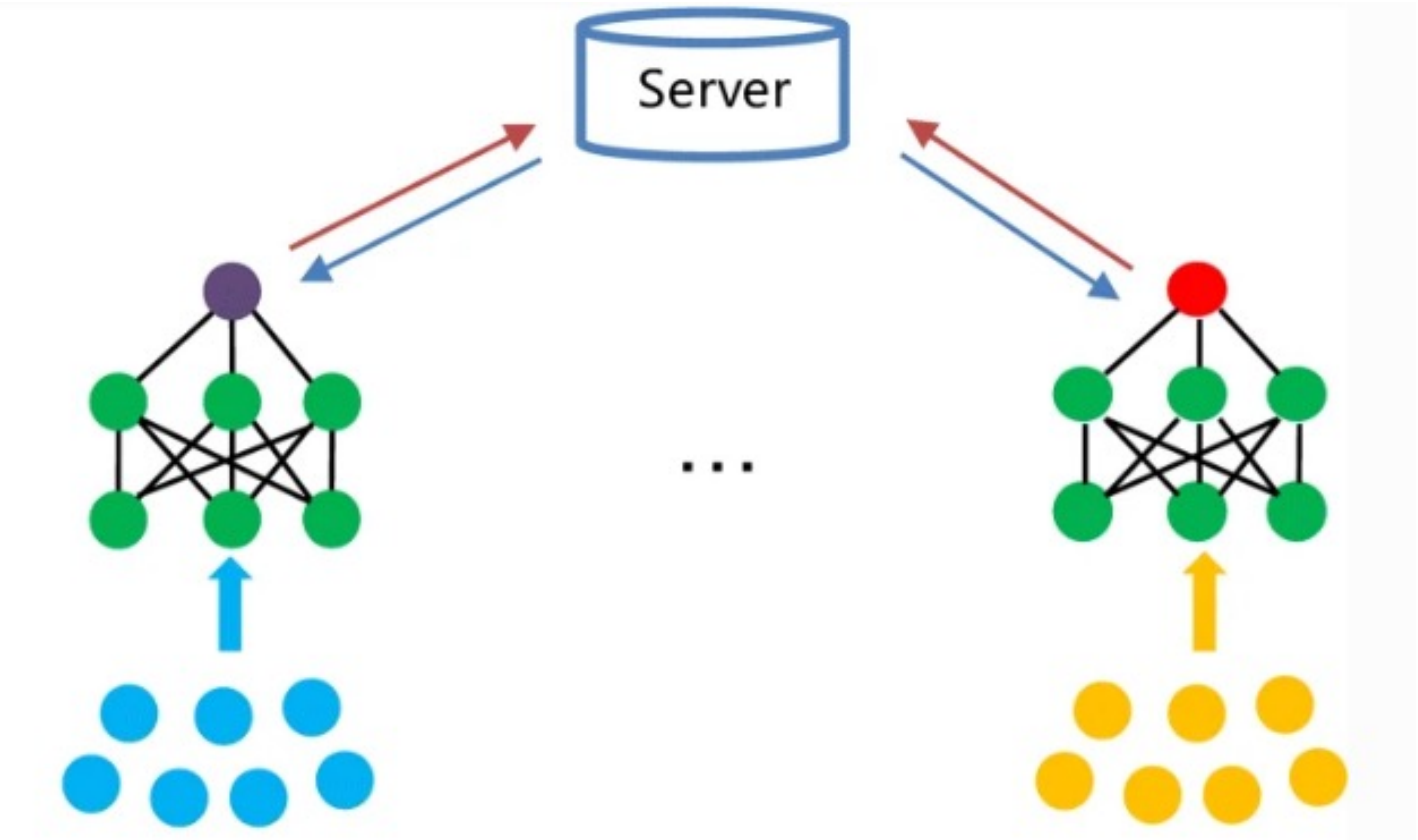
Karolina Bogacka

karolina.bogacka.dokt@pw.edu.pl

21 grudnia 2022

Szkoła Doktorska Politechniki Warszawskiej

Wydział Matematyki i Nauk Informacyjnych

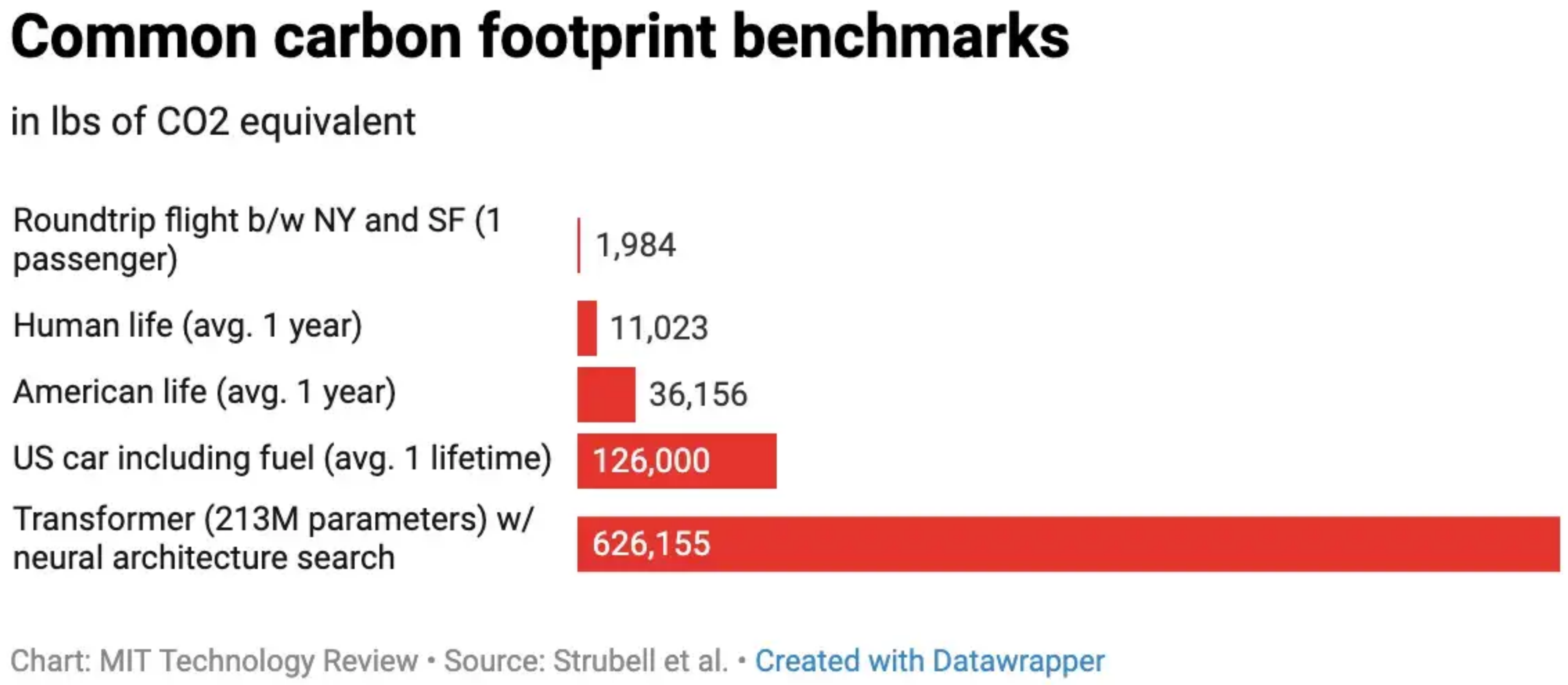


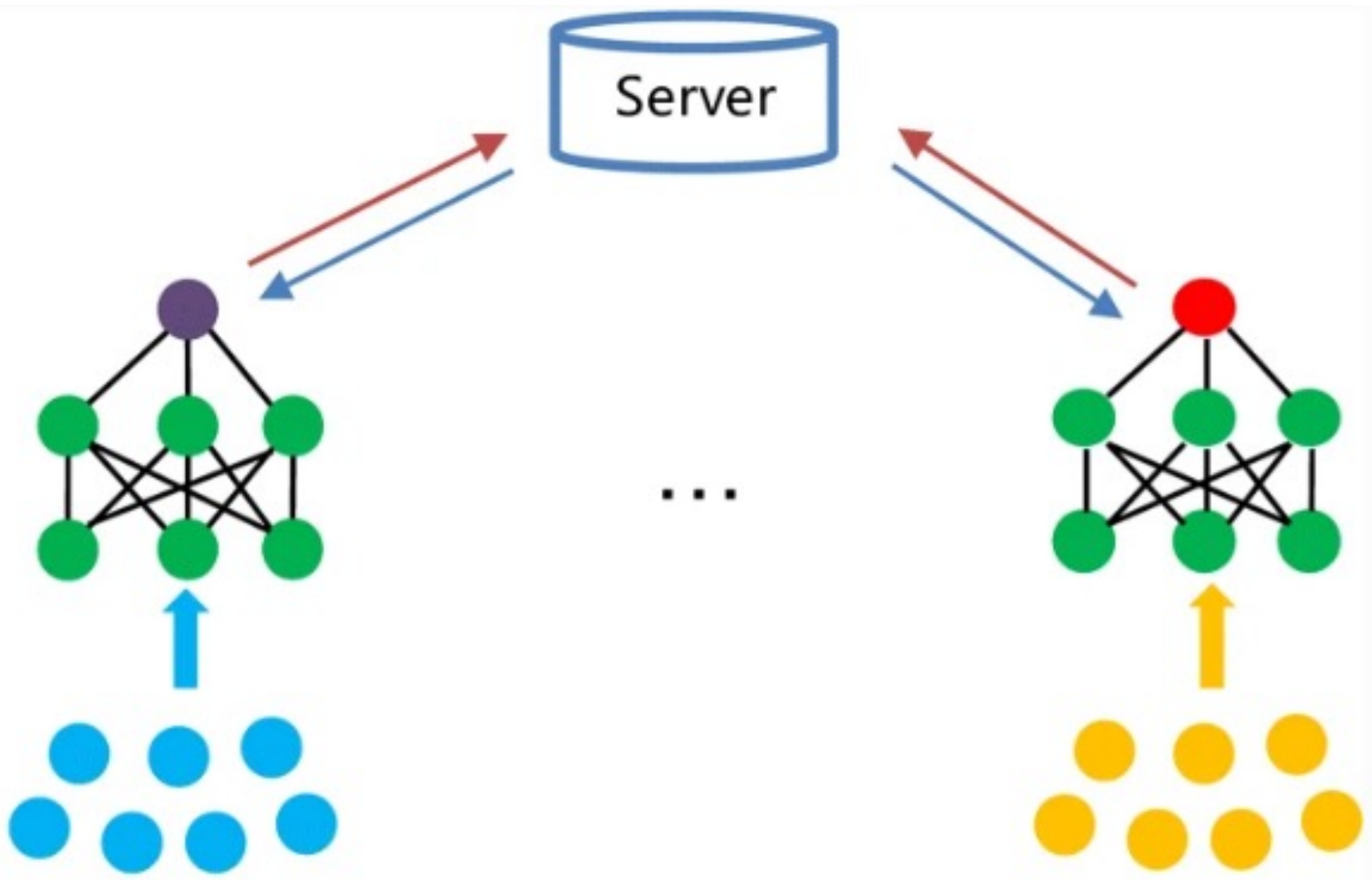
Huang, X., Ding, Y., Jiang, Z.L. *et al.* DP-FL: a novel differentially private federated learning framework for the unbalanced data. *World Wide Web* **23**, 2529–2545 (2020). https://doi.org/10.1007/s11280-020-00780-4

WHY?

Privacy

Efficiency


Common carbon footprint benchmarks
in lbs of CO2 equivalent
Roundtrip flight b/w NY and SF (1 passenger)
1,984
Human life (avg. 1 year)
11,023
American life (avg. 1 year)
36,156
US car including fuel (avg. 1 lifetime)
126,000
Transformer (213M parameters) w/ neural architecture search
626,155
Chart: MIT Technology Review • Source: Strubell et al. • Created with Datawrapper


Server
...

Number of Devices

Network Bandwidth

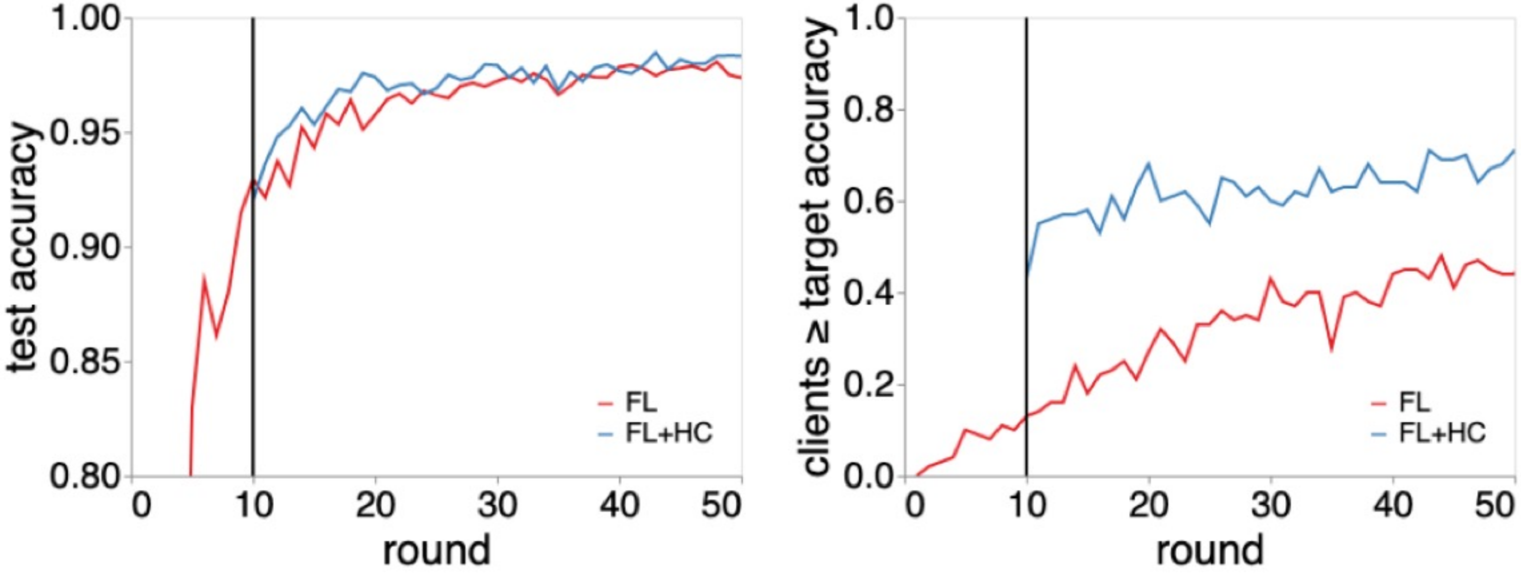
Limited Edge Node Computation

Statistical Heterogeneity

- Local Updating
- Client Selection
- Reducing Model Updates
- Decentralized Training and Different Topologies
- Compression Schemes

Local Updating

Federated learning with hierarchical clustering of local updates to improve training on non-IID data



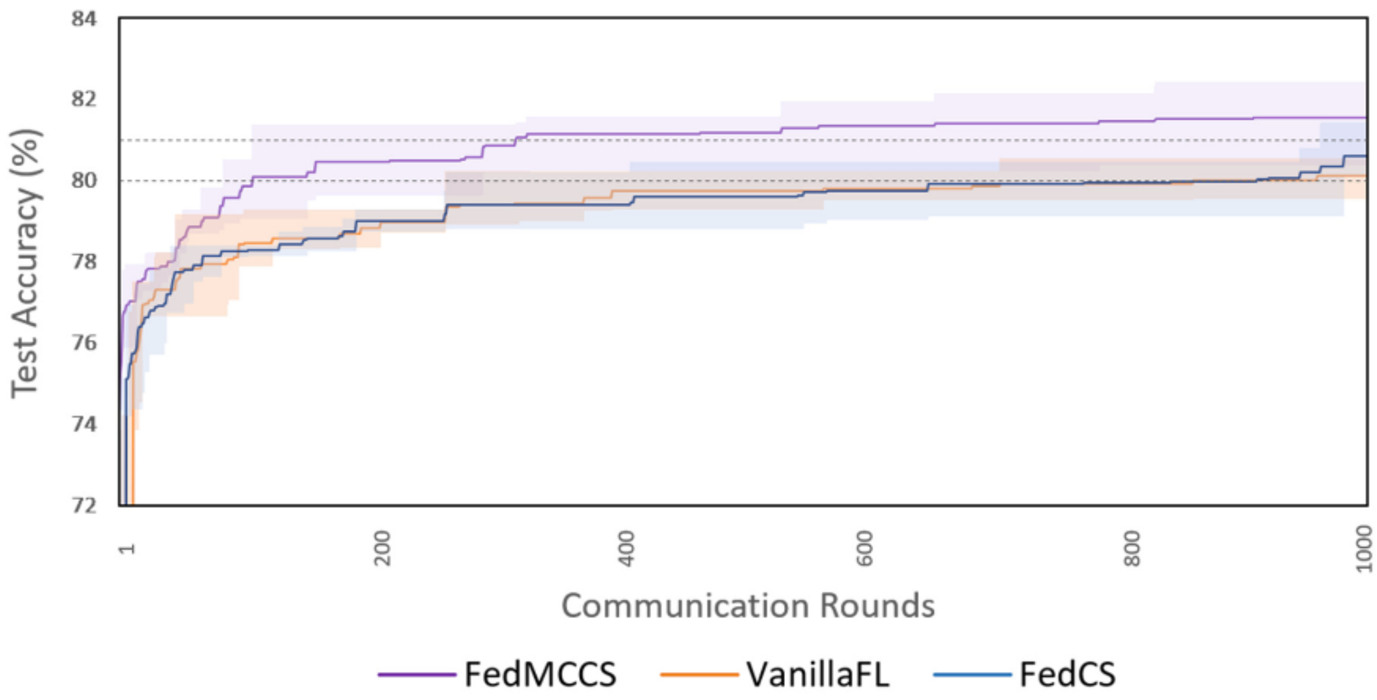
procedure FL+HC ▷ On server
Initialise w_0
for each round $t \in [1, n]$ **do**
$w_{t+1} \leftarrow \textsc{FederatedLearning}(w_t, K)$
end for
$w \leftarrow w_{t+1}$
for each client $k \in K$ **do** ▷ In parallel
$\Delta w^k \leftarrow \textsc{ClientUpdate}(k, w)$
end for
$C \leftarrow$ HierarchicalClusteringAlgorithm$(\Delta w, P)$
for $c \in C$ **do** ▷ In parallel
$w_{c,0} \leftarrow w$
for each round $t = 1, 2, \ldots$ **do**
$w_{c,t+1} \leftarrow \textsc{FederatedLearning}(w_{c,t}, K_c)$
end for
end for
end procedure

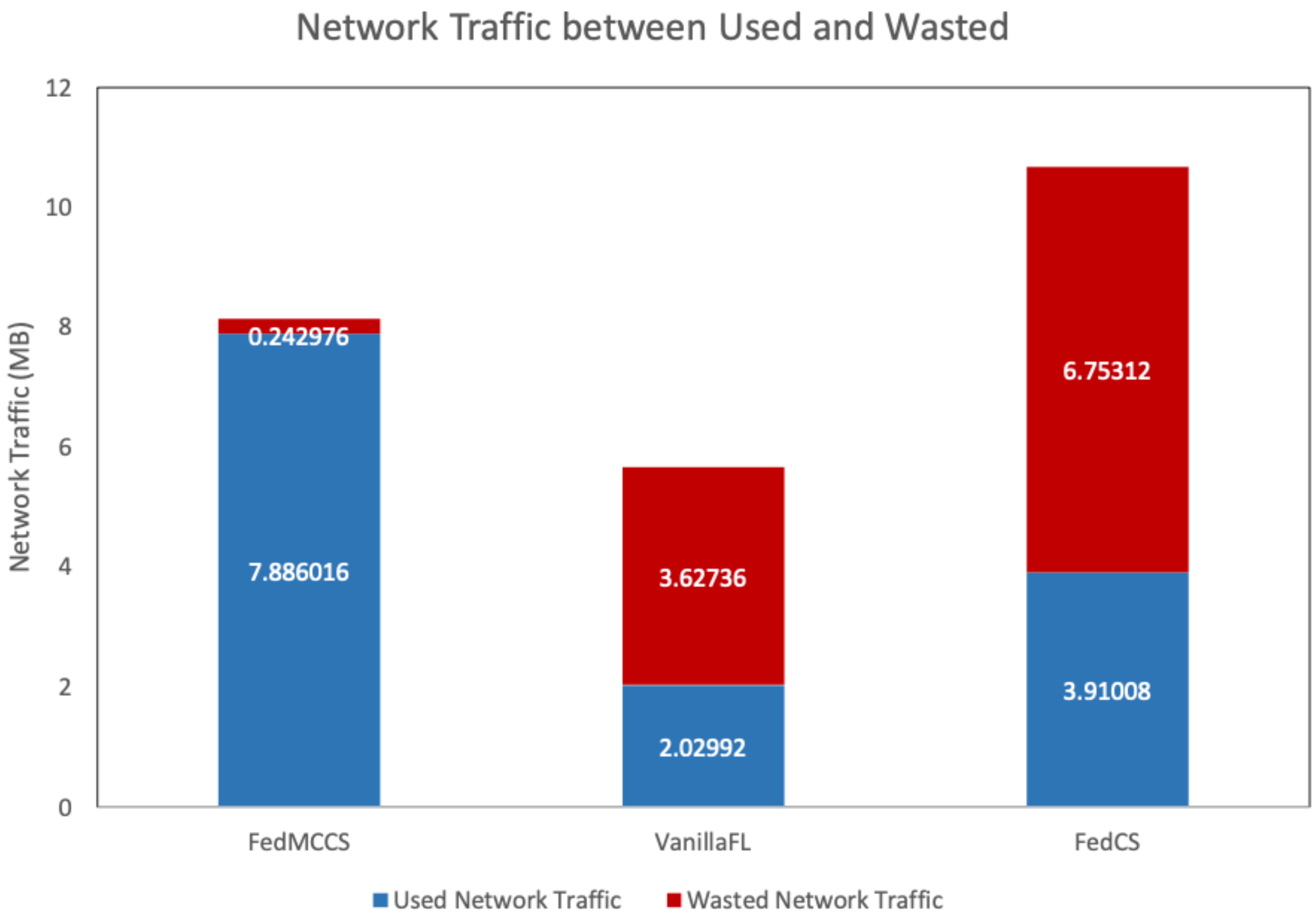
procedure $\textsc{FederatedLearning}(w_t, K)$ ▷ On server
$m \leftarrow \max(\alpha \cdot K, 1)$
$S_t \leftarrow$ (random set of m clients)
for each client $k \in S_t$ **do** ▷ In parallel
$w_{t+1}^k \leftarrow \textsc{ClientUpdate}(k, w_t)$
end for
$w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$
end procedure

procedure $\textsc{ClientUpdate}(k, w)$ ▷ On client k
$\mathcal{B} \leftarrow$ (Split $\mathcal{P}_k$ into batches of size B)
for each local epoch i from 1 to E **do**
for batch $b \in \mathcal{B}$ **do**
$w \leftarrow w - \eta \nabla \mathcal{L}(w; b)$
end for
end for
return w to server
end procedure

Client Selection

FedMCCS: Multicriteria Client Selection Model for Optimal IoT Federated Learning




Network Traffic between Used and Wasted
Network Traffic (MB)
12
10
8
6
4
2
0
0.242976
7.886016
3.62736
2.02992
6.75312
3.91008
FedMCCS
VanillaFL
FedCS
Used Network Traffic
Wasted Network Traffic

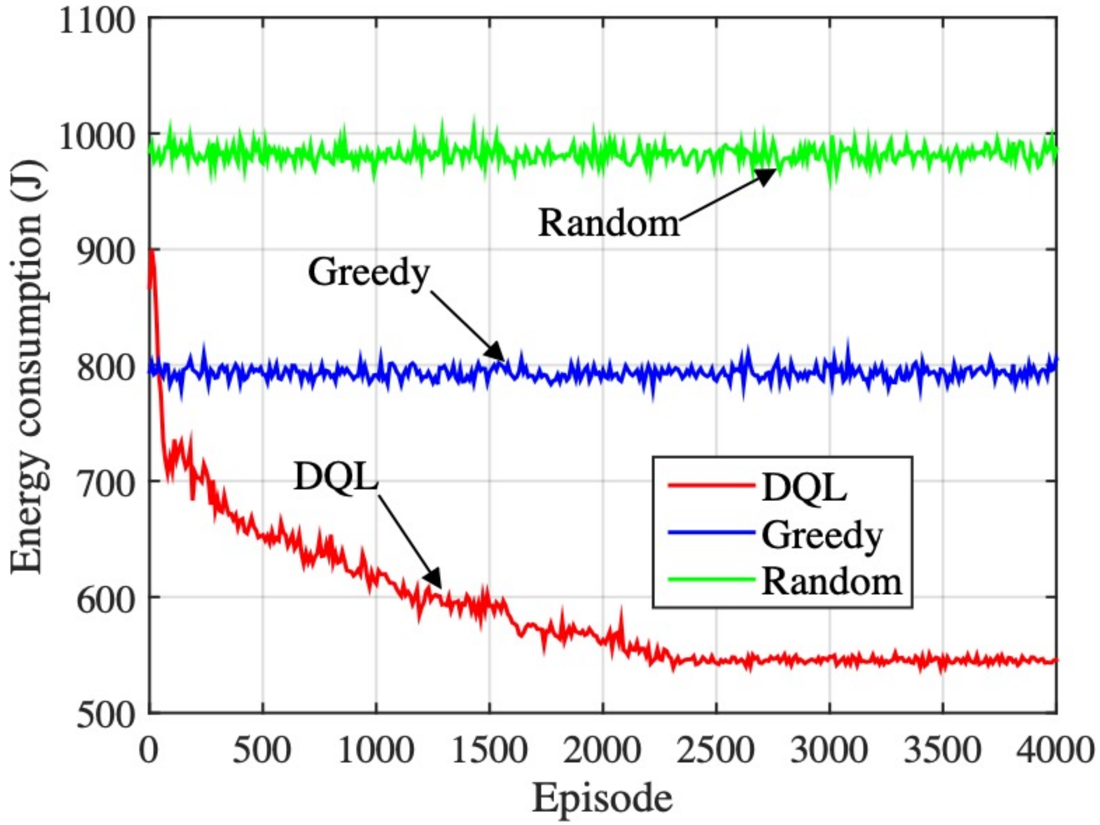
1: `Initialization` in Protocol 1.
2: `Client Filtering` : The server applies Stratified-based filtering to select clients according to their metadata, avoiding communications with irrelevant clients.
3: `Resource Request` : The server requests resource information from the filtered clients.

4: `Multi-criteria Client Selection` : Based on the clients responses, the server uses Multi-Criteria selection approach to determine a maximum of $\lceil K \times C \rceil$ clients to participate in the remaining steps.

5: `Distribution` : The server disseminates the global model parameters to the selected clients.
6: `Update and Upload` in Protocol 1.
7: `Aggregation` : The server averages the parameters, when more than 70% of the requested updates are received.
8: All steps but `Initialization` are iterated as in Protocol 2.

Client Selection

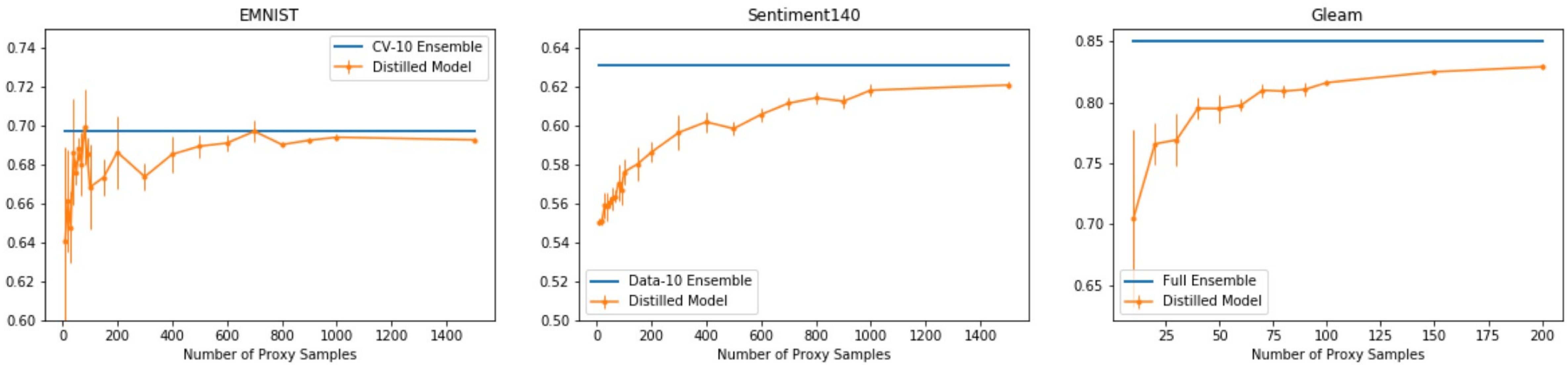
Efficient Training Management for Mobile Crowd-Machine Learning: A Deep Reinforcement Learning Approach



Client Selection

Initialize: $\boldsymbol{\theta}, \boldsymbol{\theta}^{-}$;
for episode $i = 1$ to N **do**
  for iteration $t = 1$ to T **do**
    Select action a according to the ϵ-greedy policy;
    Execute action a and observe reward r and next state s';
    Store experience $e = < s, a, r, s' >$ in $\mathcal{M}$;
    Select N_b experiences $e_k = < s_k, a_k, r_k, s'_k >$ from $\mathcal{M}$;
    for $k = 1$ to N_b **do**
      Determine $a^{\max} = \arg\max_{a' \in \mathcal{A}} Q(s'_k, a'; \boldsymbol{\theta})$;
      Calculate $y_k = r_k + \gamma Q\left(s'_k, a^{\max}; \boldsymbol{\theta}^{-}\right)$;
    end for
    Define $\overline{L}(\boldsymbol{\theta}) = \frac{1}{N_b} \sum_{k=1}^{N_b} \Big(y_k - Q(s_k, a_k; \boldsymbol{\theta})\Big)^2$;
    Perform a gradient descent step on $\overline{L}(\boldsymbol{\theta})$ to update $\boldsymbol{\theta}$;
    Reset $\boldsymbol{\theta}^{-} = \boldsymbol{\theta}$ in every N^{-} iteration;
  end for
end for

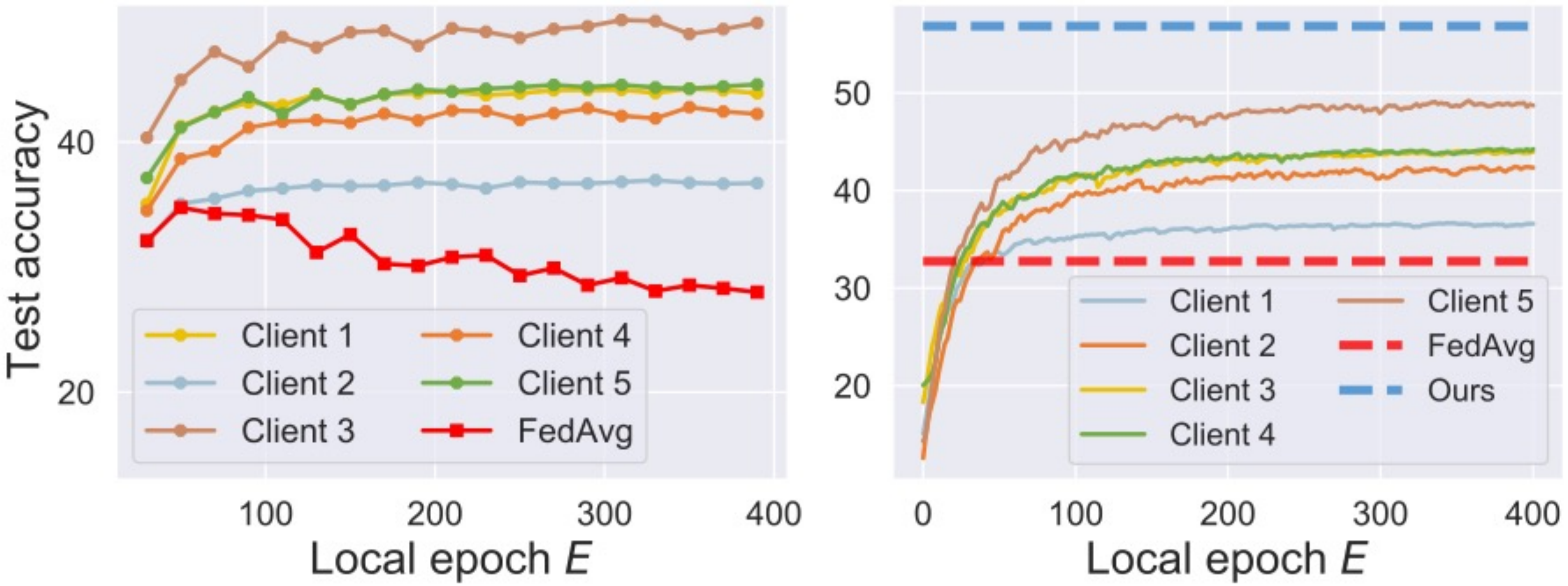
Reducing Model Updates

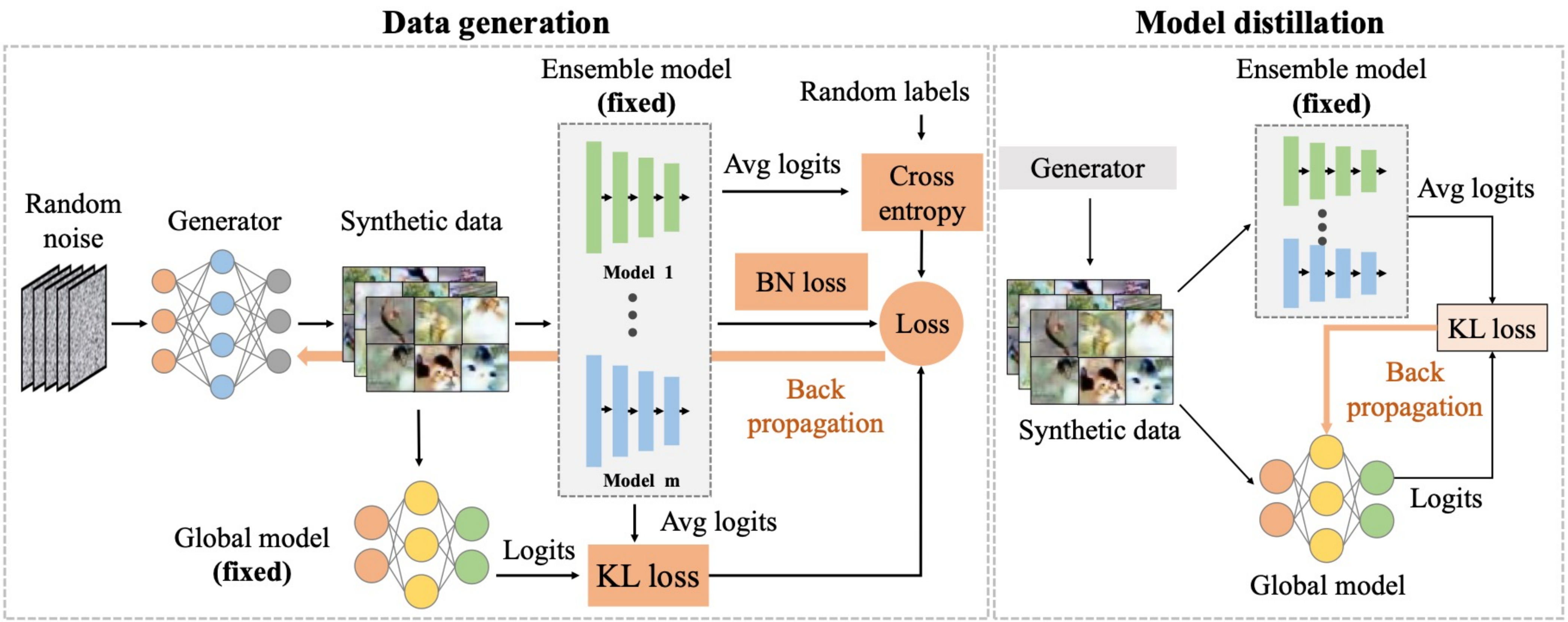


One-Shot Federated Learning

Reducing Model Updates

DENSE: Data-Free One-Shot Federated Learning



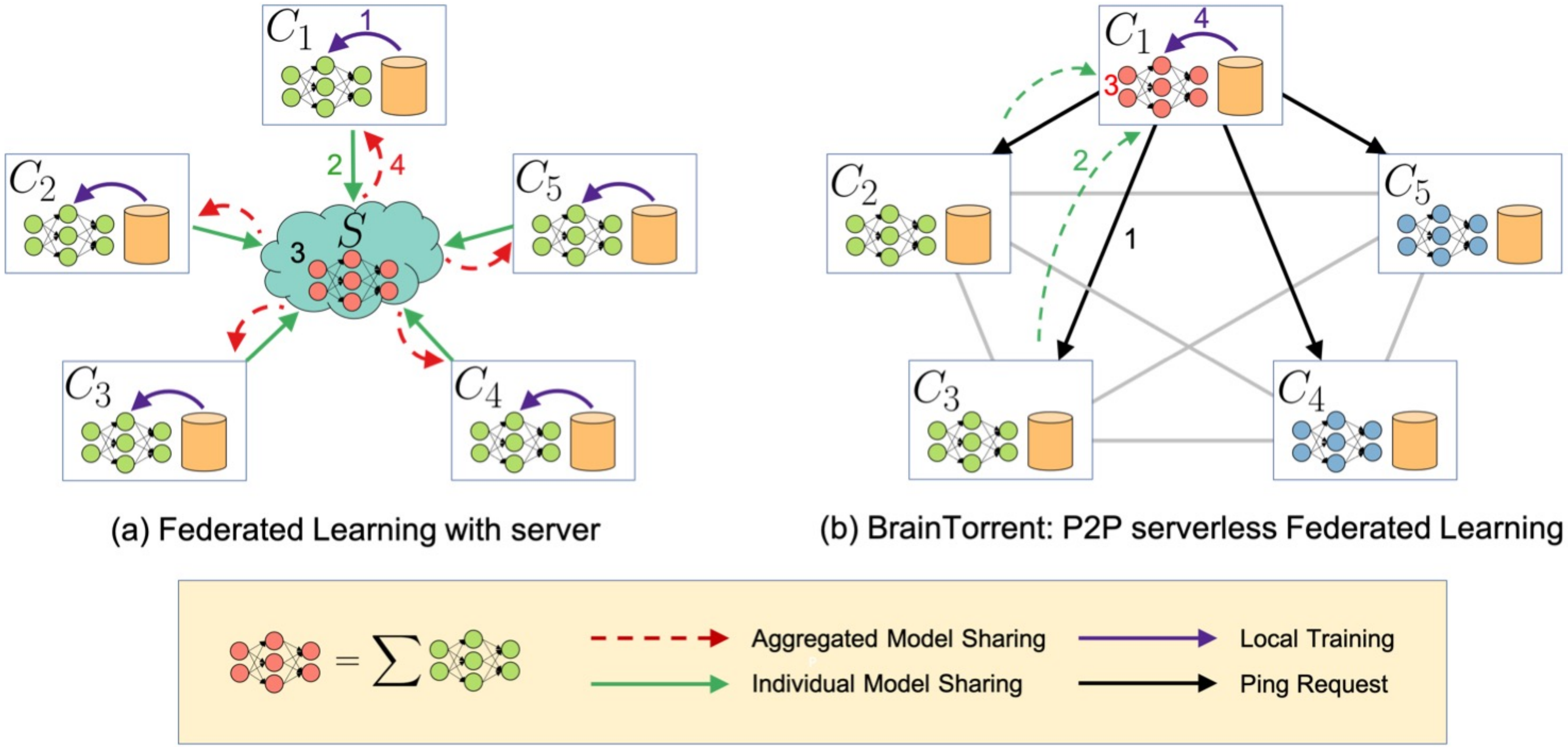


Data generation
Model distillation
Random noise
Generator
Synthetic data
Ensemble model
(fixed)
Model 1
Model m
Random labels
Avg logits
Cross entropy
BN loss
Loss
Back propagation
Global model
(fixed)
Logits
Avg logits
KL loss
Generator
Ensemble model
(fixed)
Avg logits
KL loss
Back propagation
Synthetic data
Logits
Global model

Decentralized Training

BrainTorrent: A Peer-to-Peer Environment for Decentralized Federated Learning

# Clients	Scans/client	Avg. Dice over Clients		Aggregated Model	
		FLS	BrainTorrent	FLS	BrainTorrent
5	4	0.812	0.851	0.845	**0.863**
7	3	0.753	0.837	0.843	0.861
10	2	0.792	0.807	0.842	0.850
20	1	0.570	0.578	0.687	0.728
Pooled Model			**0.866**		



C_1
C_2
C_3
C_4
C_5
S
1
2
3
4
(a) Federated Learning with server
(b) BrainTorrent: P2P serverless Federated Learning
Aggregated Model Sharing
Individual Model Sharing
Local Training
Ping Request

Initialize N clients models, $\mathbf{C} = \{C_1, \ldots, C_N\}$ with random weights;
Initialize N version vectors $\mathbf{V} = [\mathbf{v}^1, \ldots, \mathbf{v}^N]$ with all zero entries;
for *round* r *in 1, 2, . . .* **do**
  Randomly select a client i from $\{1, \ldots, N\}$;
  $\mathbf{v}_{\text{old}} \leftarrow \mathbf{v}^i$;
  $\mathbf{v}_{\text{new}} \leftarrow \text{ping\_request}(C_i \rightarrow \mathbf{C})$;
  $\mathbf{W} \leftarrow \frac{a_i}{a}\mathbf{W}^i$;
  for $j \in \{1, \ldots, i-1, i+1, \ldots, N\}$ **do**
    if $v^j_{\text{new}} > v^j_{\text{old}}$ **then**
      Receive updated $\mathbf{W}^j$ and a_j from C_j ;
    end
    $\mathbf{W} \leftarrow \mathbf{W} + \frac{a_j}{a}\mathbf{W}^j$;
  end
  $\mathbf{W}^i \leftarrow \text{FineTune}(\mathbf{W}, \mathcal{D}_i)$;
  Increment $\mathbf{v}^i(i)$;
end

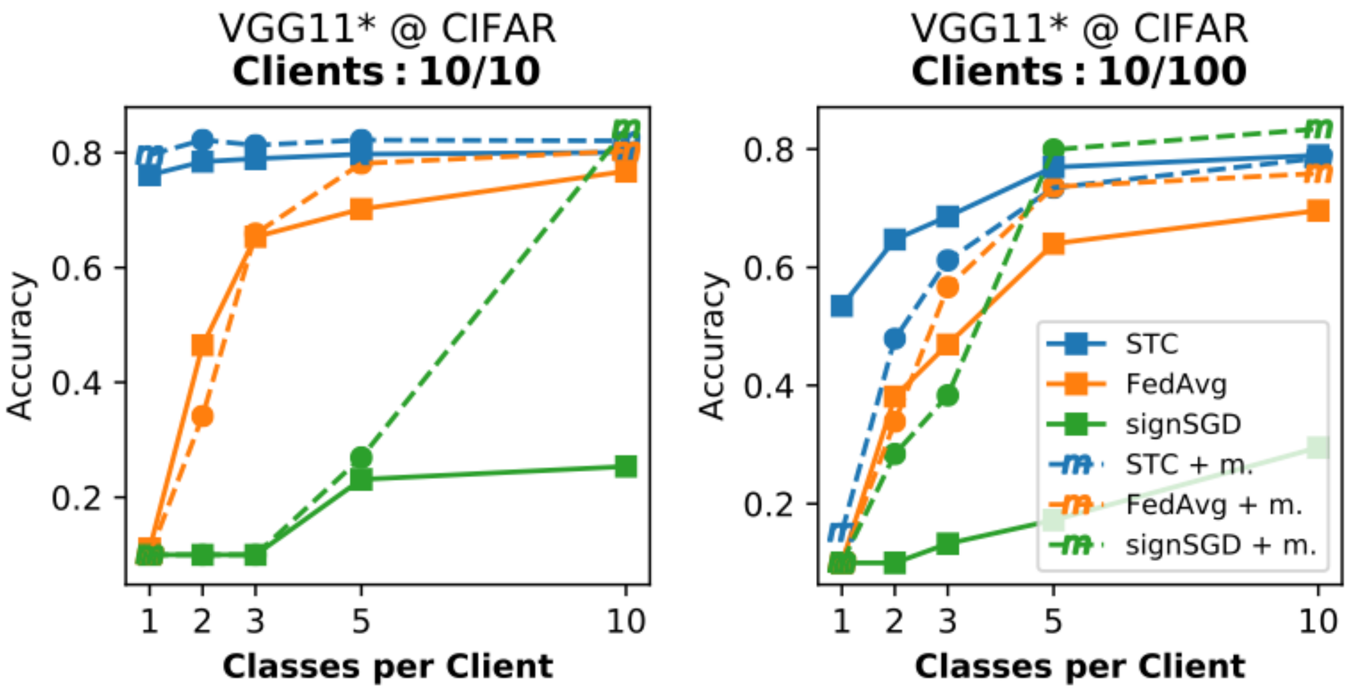
Compression Schemes

Sparsification

Quantization

Compression Schemes

Robust and Communication-Efficient Federated Learning from Non-IID Data



input: initial parameters $\mathcal{W}$

output: improved parameters $\mathcal{W}$

init: all clients C_i, $i = 1, .., [\text{Number of Clients}]$ are initialized with the same parameters $\mathcal{W}_i \leftarrow \mathcal{W}$. Every Client holds a different dataset D_i, with $|\{y : (x, y) \in D_i\}| = [\text{Classes per Client}]$ of size $|D_i| = \varphi_i |\cup_j D_j|$. The residuals are initialized to zero $\Delta\mathcal{W}, \mathcal{R}_i, \mathcal{R} \leftarrow 0$.

for $t = 1, .., T$ **do**

  for $i \in I_t \subseteq \{1, .., [\text{Number of Clients}]\}$ ***in parallel*** **do**

    Client C_i does:

    - $\text{msg} \leftarrow \text{download}_{S \to C_i}(\text{msg})$
    - $\Delta\mathcal{W} \leftarrow \text{decode}(\text{msg})$
    - $\mathcal{W}_i \leftarrow \mathcal{W}_i + \Delta\mathcal{W}$
    - $\Delta\mathcal{W}_i \leftarrow \mathcal{R}_i + \text{SGD}(\mathcal{W}_i, D_i, b) - \mathcal{W}_i$
    - $\Delta\tilde{\mathcal{W}}_i \leftarrow \text{STC}_{p_{up}}(\Delta\mathcal{W}_i)$
    - $\mathcal{R}_i \leftarrow \Delta\mathcal{W}_i - \Delta\tilde{\mathcal{W}}_i$
    - $\text{msg}_i \leftarrow \text{encode}(\Delta\tilde{\mathcal{W}}_i)$
    - $\text{upload}_{C_i \to S}(\text{msg}_i)$

  end

  Server S does:

  - $\text{gather}_{C_i \to S}(\Delta\tilde{\mathcal{W}}_i),\ i \in I_t$
  - $\Delta\mathcal{W} \leftarrow \mathcal{R} + \frac{1}{|I_t|} \sum_{i \in I_t} \Delta\tilde{\mathcal{W}}_i$
  - $\Delta\tilde{\mathcal{W}} \leftarrow \text{STC}_{p_{down}}(\Delta\mathcal{W})$
  - $\mathcal{R} \leftarrow \Delta\mathcal{W} - \Delta\tilde{\mathcal{W}}$
  - $\mathcal{W} \leftarrow \mathcal{W} + \Delta\tilde{\mathcal{W}}$
  - $\text{msg} \leftarrow \text{encode}(\Delta\tilde{\mathcal{W}})$
  - $\text{broadcast}_{S \to C_i}(\text{msg}),\ i = 1, .., M$

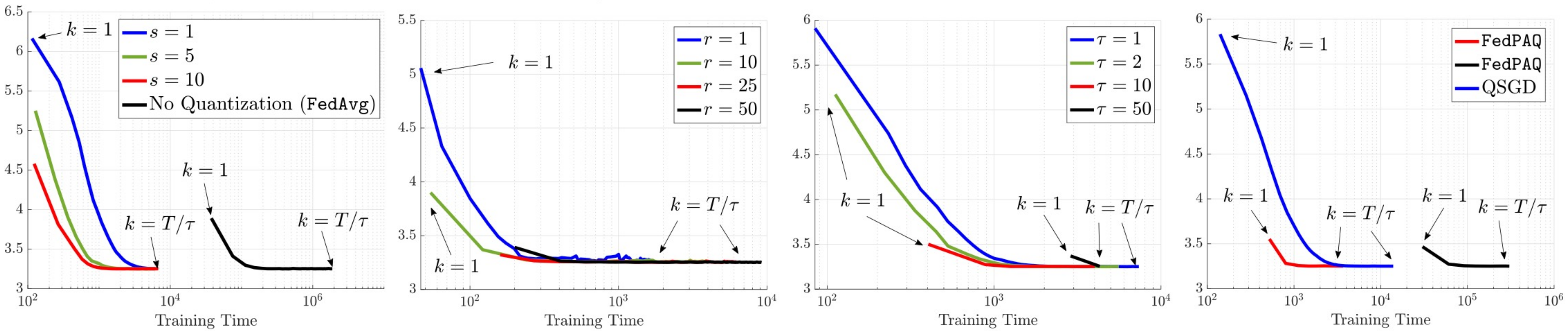
end

return $\mathcal{W}$

input: flattened tensor $T \in \mathbb{R}^n$, sparsity p
output: sparse ternary tensor $T^* \in \{-\mu, 0, \mu\}^n$
- $k \leftarrow \max(np, 1)$
- $v \leftarrow \text{top}_k(|T|)$
- $\text{mask} \leftarrow (|T| \geq v) \in \{0, 1\}^n$
- $T^{masked} \leftarrow \text{mask} \odot T$
- $\mu \leftarrow \frac{1}{k} \sum_{i=1}^{n} |T_i^{masked}|$

return $T^* \leftarrow \mu \times \text{sign}(T^{masked})$

Compression Schemes



FedPAQ: A Communication-Efficient Federated Learning Method with Periodic Averaging and Quantization

for $k = 0, 1, \cdots, K-1$ **do**
    server picks r nodes $\mathcal{S}_k$ uniformly at random
    server sends $\mathbf{x}_k$ to nodes in $\mathcal{S}_k$
    for node $i \in \mathcal{S}_k$ **do**
        $\mathbf{x}_{k,0}^{(i)} \leftarrow \mathbf{x}_k$
        for $t = 0, 1, \cdots, \tau - 1$ **do**
            compute stochastic gradient
            $\widetilde{\nabla} f_i(\mathbf{x}) = \nabla \ell(\mathbf{x}, \xi)$ for a $\xi \in \mathcal{P}^i$
            set $\mathbf{x}_{k,t+1}^{(i)} \leftarrow \mathbf{x}_{k,t}^{(i)} - \eta_{k,t} \widetilde{\nabla} f_i(\mathbf{x}_{k,t}^{(i)})$
        end for
        send $Q(\mathbf{x}_{k,\tau}^{(i)} - \mathbf{x}_k)$ to the server
    end for
    server finds $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \frac{1}{r} \sum_{i \in \mathcal{S}_k} Q(\mathbf{x}_{k,\tau}^{(i)} - \mathbf{x}_k)$
end for

Bibliography

- Huang, X., Ding, Y., Jiang, Z.L. *et al.* DP-FL: a novel differentially private federated learning framework for the unbalanced data. *World Wide Web* **23**, 2529–2545 (2020). https://doi.org/10.1007/s11280-020-00780-4
- Strubell, Emma & Ganesh, Ananya & Mccallum, Andrew. (2019). Energy and Policy Considerations for Deep Learning in NLP. 3645-3650. 10.18653/v1/P19-1355.
- Shahid, Osama, et al. "Communication efficiency in federated learning: Achievements and challenges." *arXiv preprint arXiv:2107.10996* (2021).
- Briggs, Christopher, Zhong Fan, and Peter Andras. "Federated learning with hierarchical clustering of local updates to improve training on non-IID data." *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020.
- Anh, Tran The, et al. "Efficient training management for mobile crowd-machine learning: A deep reinforcement learning approach." *IEEE Wireless Communications Letters* 8.5 (2019): 1345-1348.

Bibliography

- S. Abdulrahman, H. Tout, A. Mourad and C. Talhi, "FedMCCS: Multicriteria Client Selection Model for Optimal IoT Federated Learning," in IEEE Internet of Things Journal, vol. 8, no. 6, pp. 4723-4735, 15 March15, 2021, doi: 10.1109/JIOT.2020.3028742.
- Guha, Neel, Ameet Talwalkar, and Virginia Smith. "One-shot federated learning." *arXiv preprint arXiv:1902.11175* (2019).
- Zhang, Jie, et al. "A Practical Data-Free Approach to One-shot Federated Learning with Heterogeneity." *arXiv preprint arXiv:2112.12371* (2021).
- Roy, Abhijit Guha et al. “BrainTorrent: A Peer-to-Peer Environment for Decentralized Federated Learning.” *ArXiv* abs/1905.06731 (2019): n. pag.
- F. Sattler, S. Wiedemann, K. -R. Müller and W. Samek, "Robust and Communication-Efficient Federated Learning From Non-i.i.d. Data," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 9, pp. 3400-3413, Sept. 2020, doi: 10.1109/TNNLS.2019.2944481.
- Reisizadeh, Amirhossein, et al. "Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization." *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020.