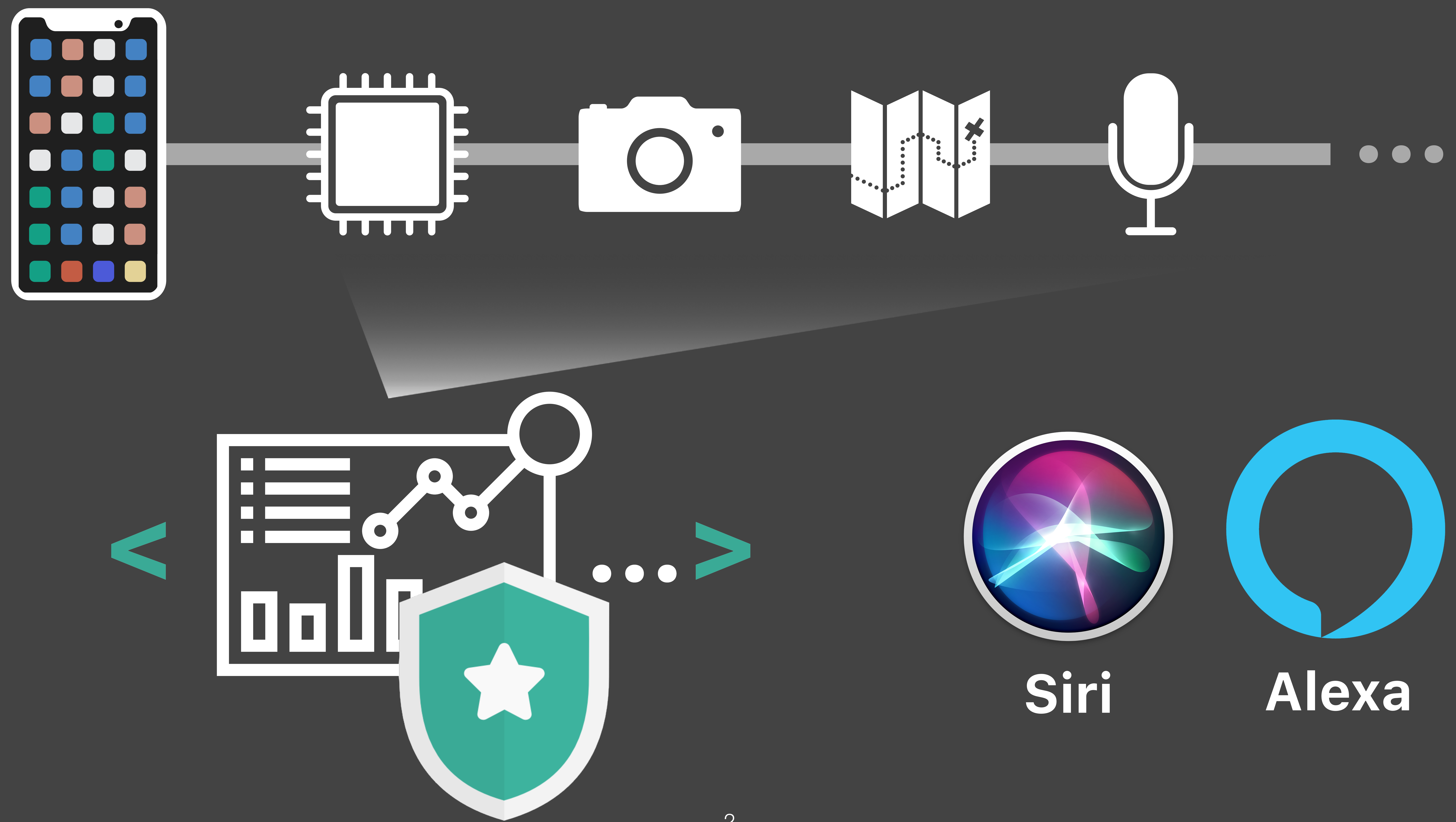
INFOCOM'20

Optimizing Federated Learning on Non-IID Data with Reinforcement Learning

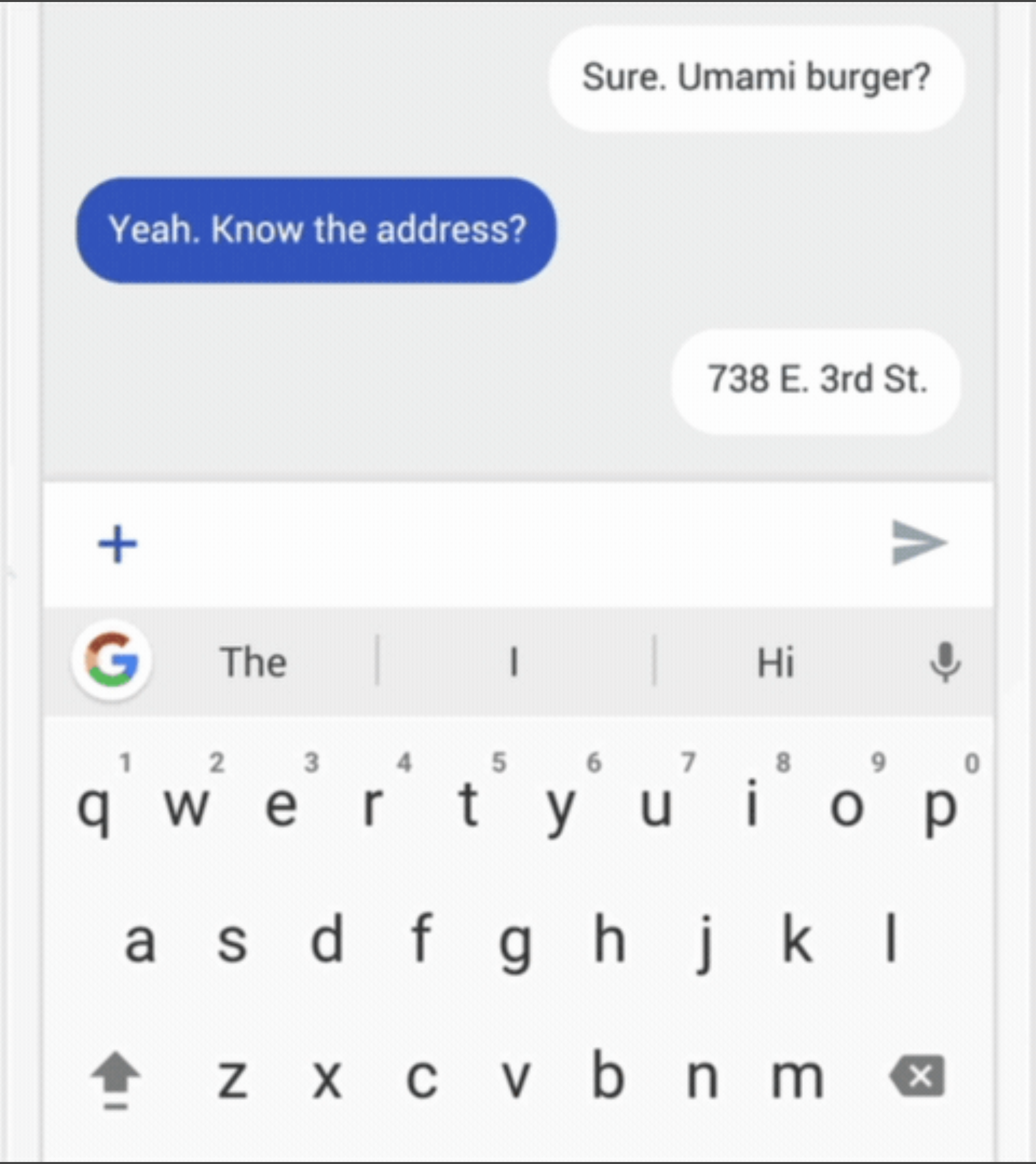
Hao Wang*, Zakhary Kaplan*, Di Niu^, Baochun Li*

*University of Toronto, ^University of Alberta


Siri
Alexa

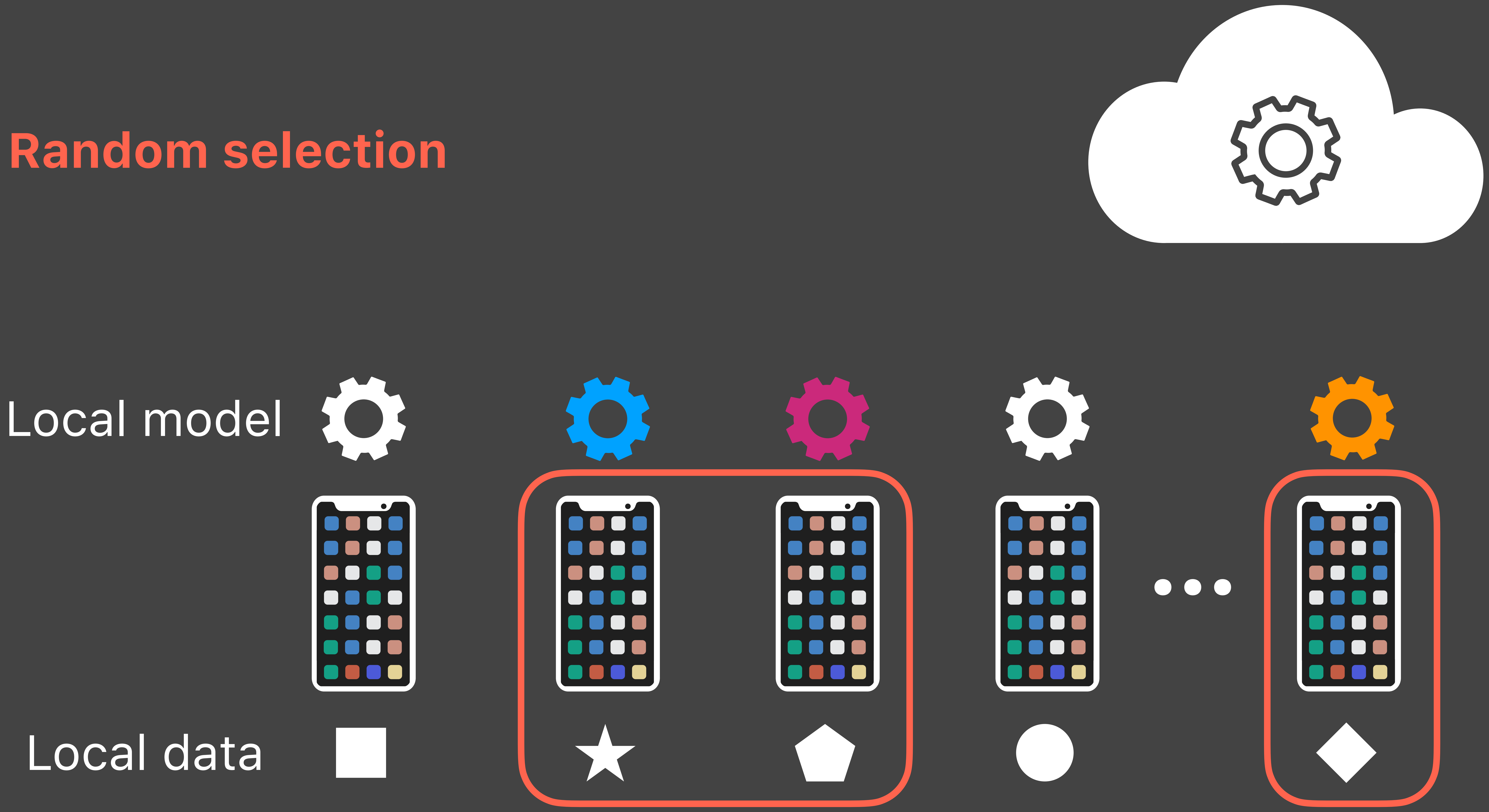
Machine Learning

Federated Learning

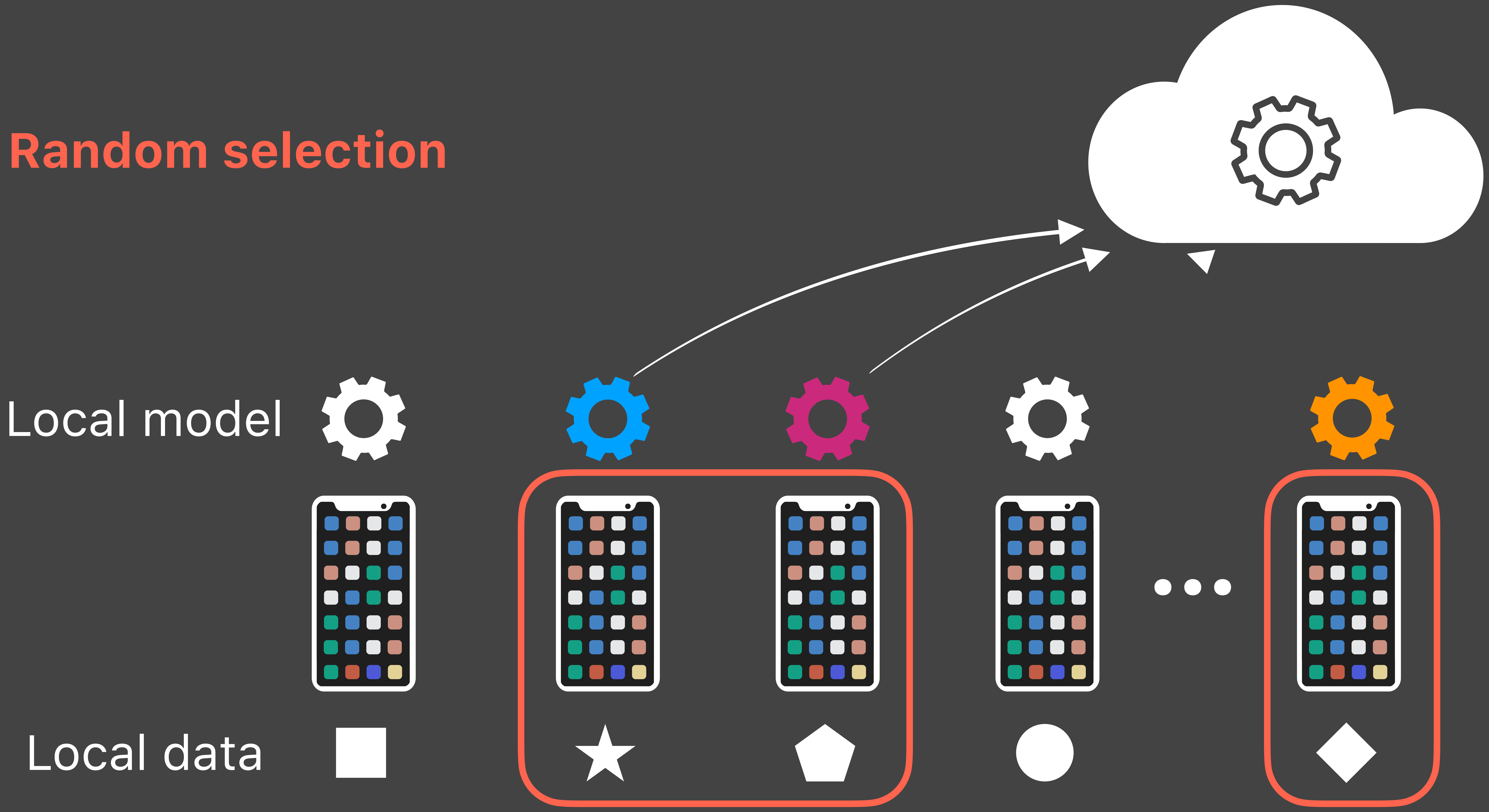

Sure. Umami burger?
Yeah. Know the address?
738 E. 3rd St.
The
I
Hi
1 2 3 4 5 6 7 8 9 0
q w e r t y u i o p
a s d f g h j k l
z x c v b n m

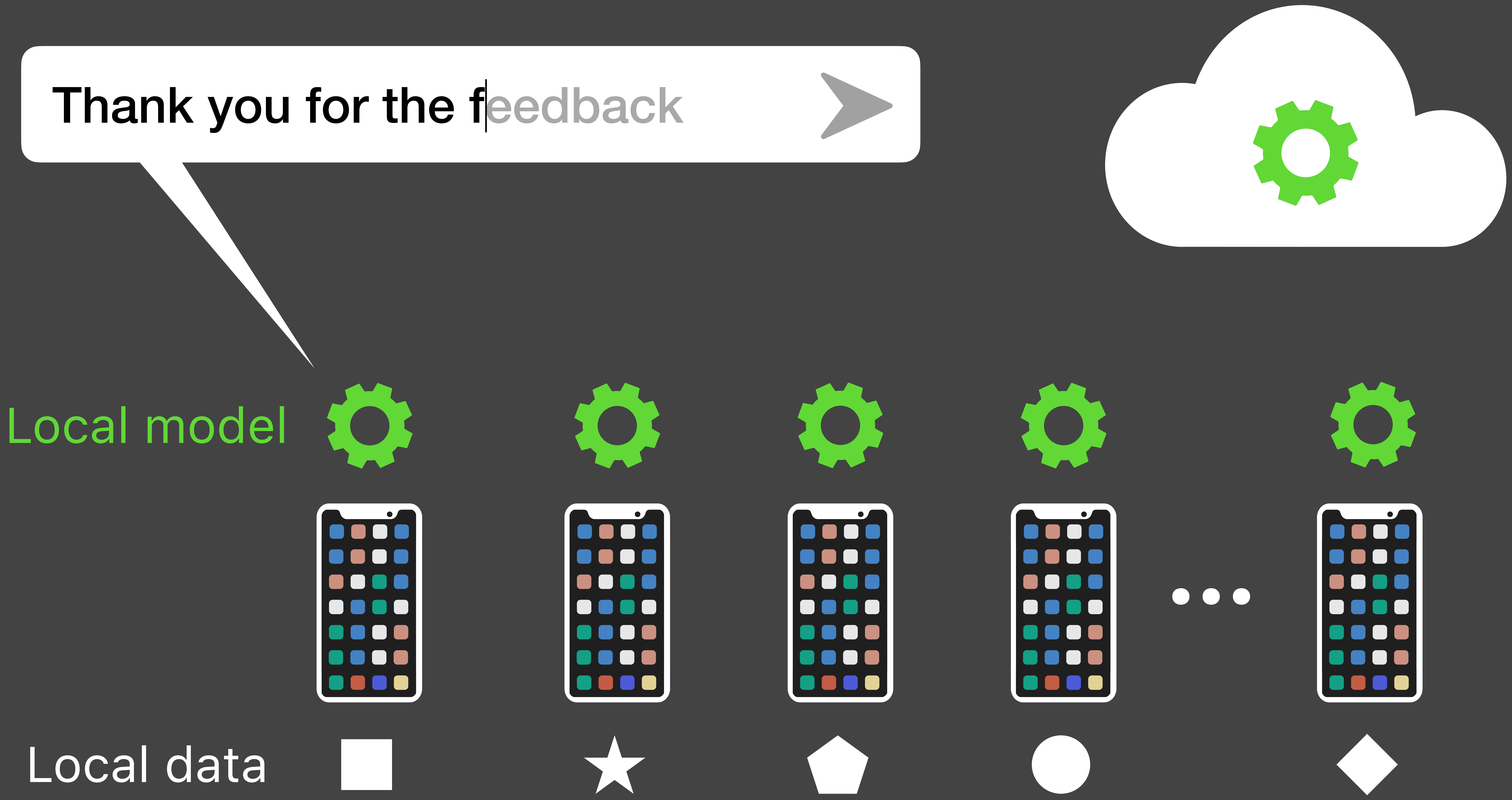
Federated Averaging Algorithm (FedAvg)

Random selection



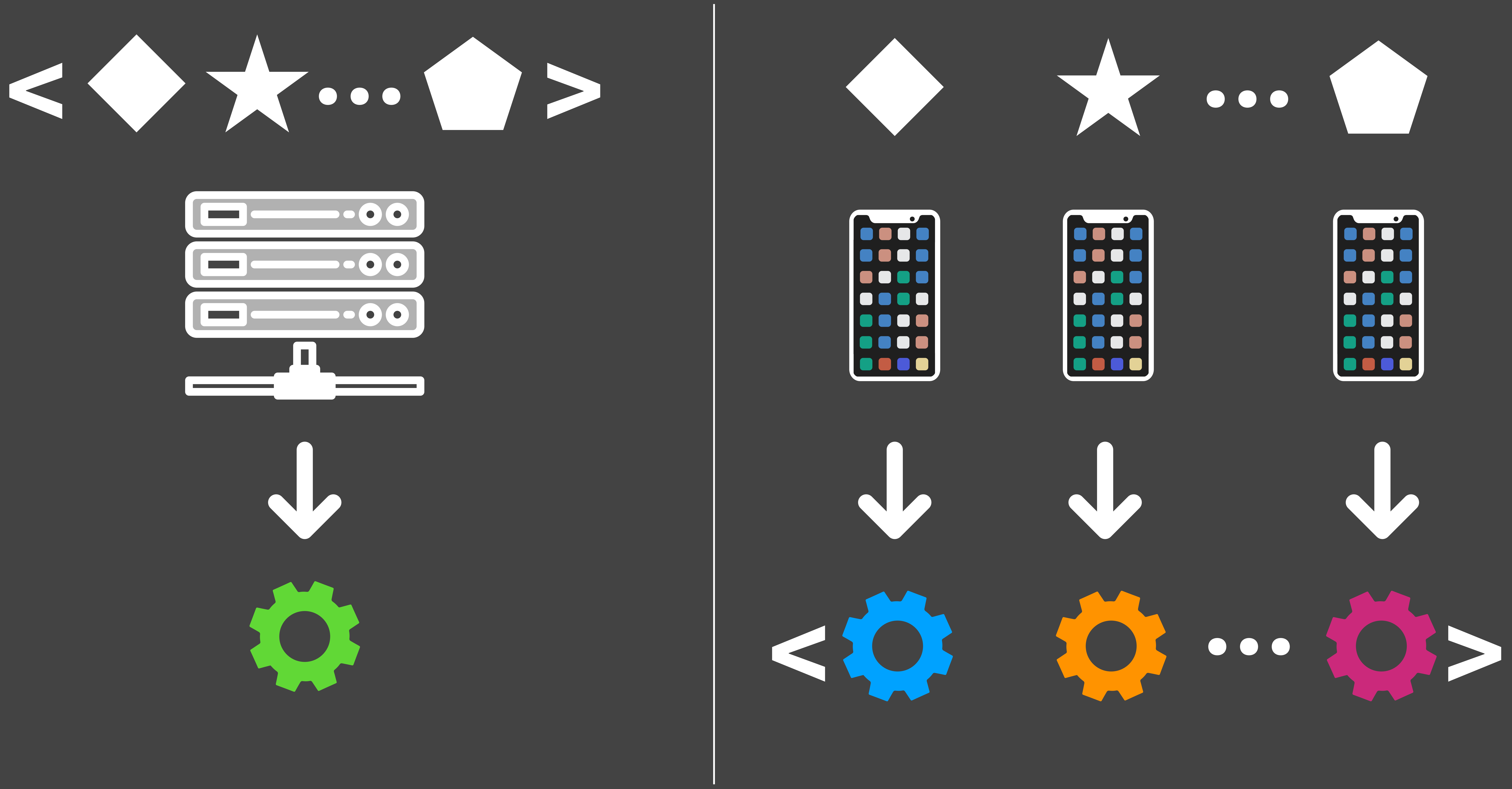
Random selection

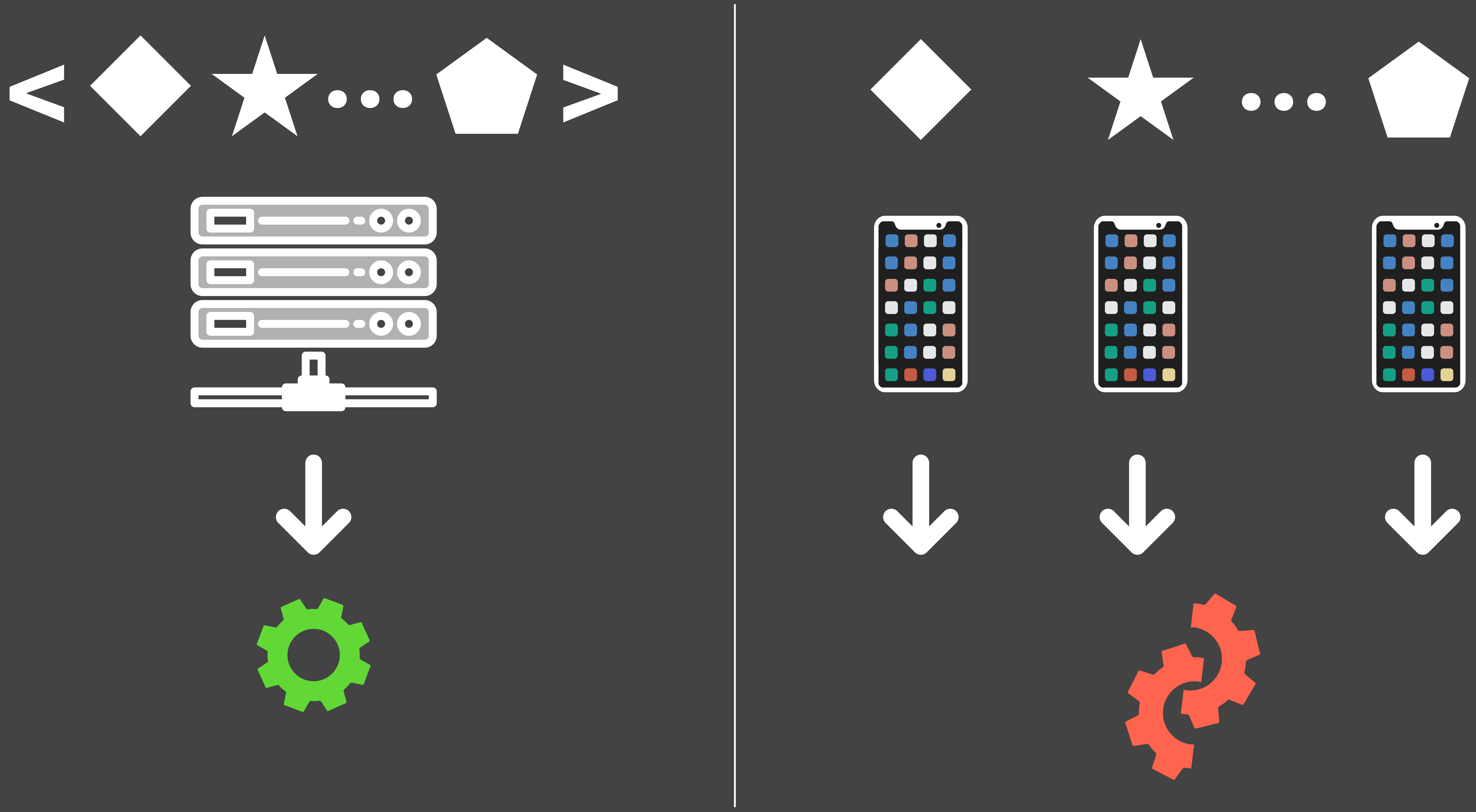



Thank you for the feedback
Local model
Local data

ML algorithms assume the training data is **independent and identically distributed (IID)**

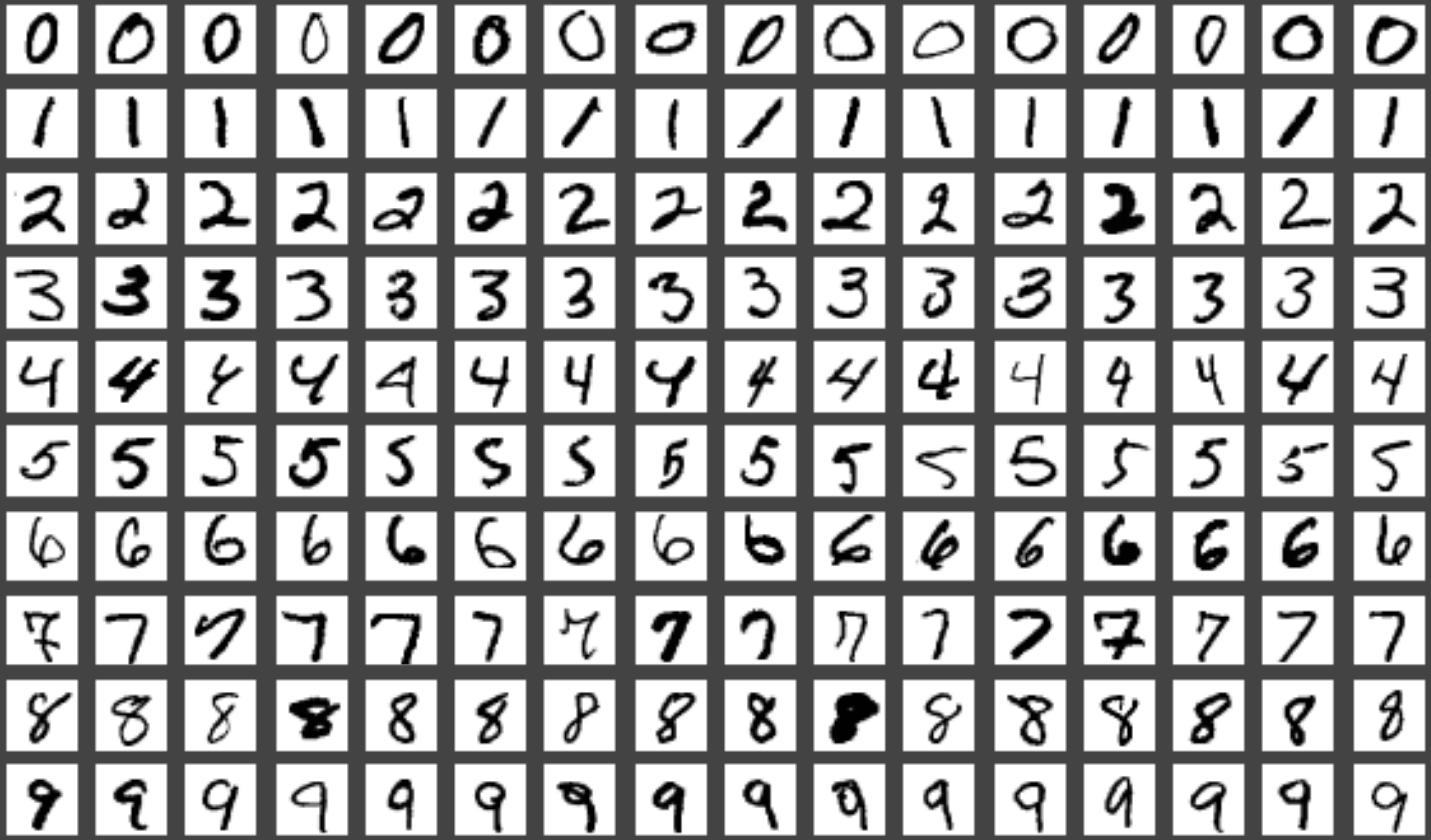
Federated Learning reuses the existing ML algorithms but on **non-IID data**



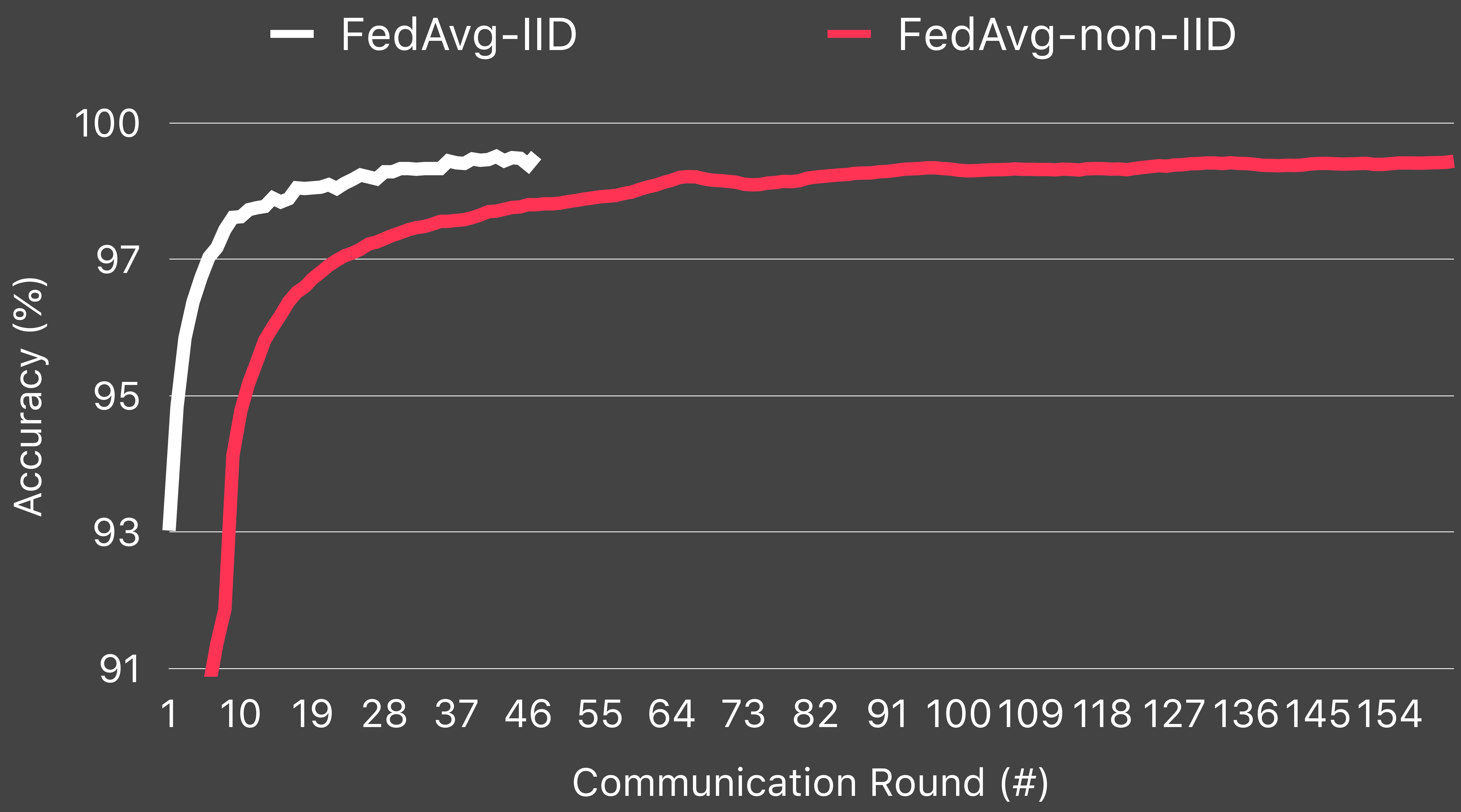


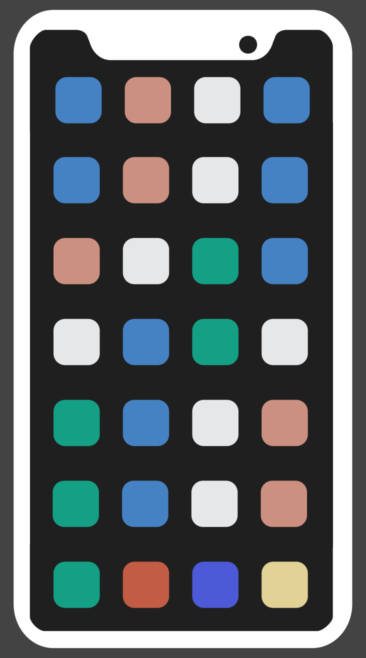
Non-IID data introduces bias into the training and leads to **a slow convergence** and **training failures**

MNIST



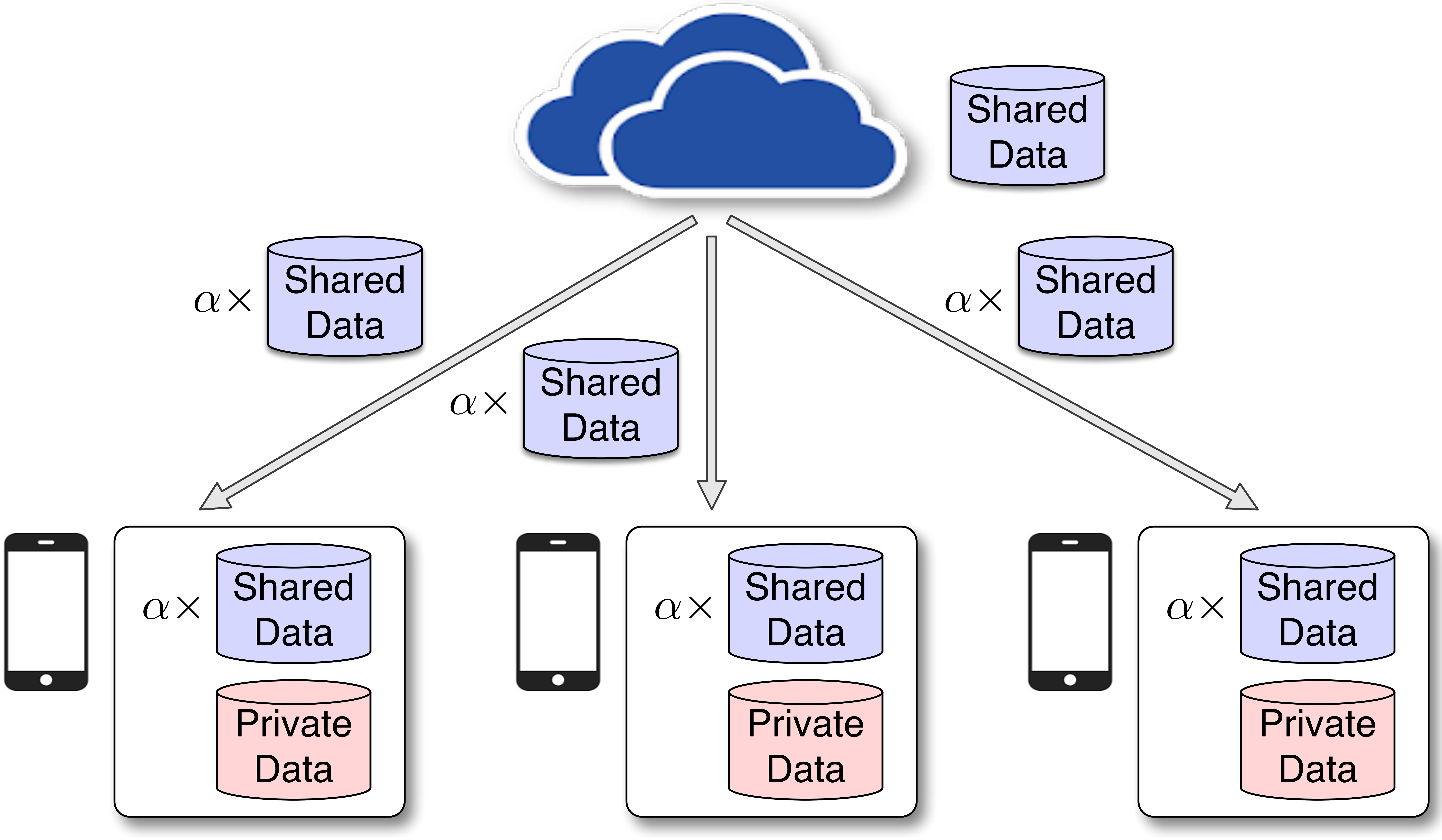
http://yann.lecun.com/exdb/mnist/


FedAvg-IID
FedAvg-non-IID
100
97
95
93
91
Accuracy (%)
1
10
19
28
37
46
55
64
73
82
91
100
109
118
127
136
145
154
Communication Round (#)



Build IID training data?

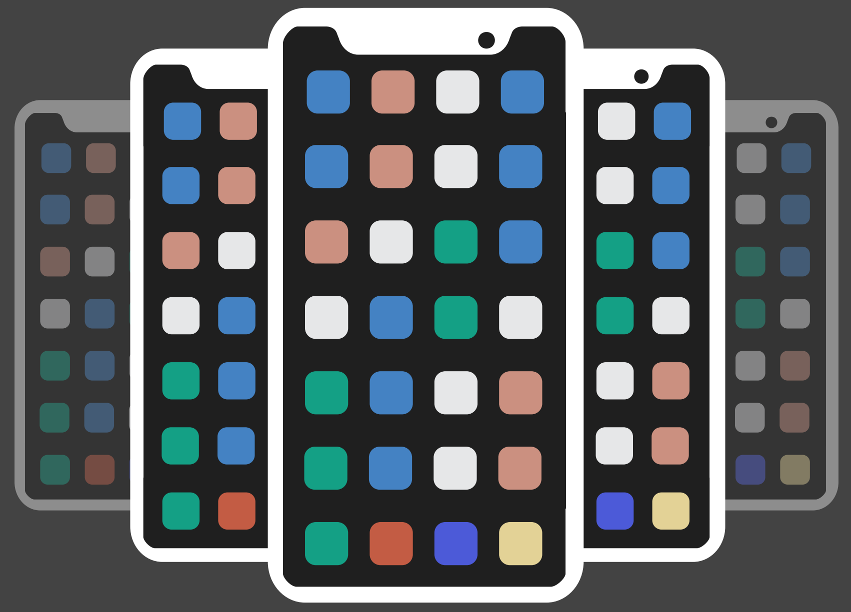
No, we **don’t have any access** to the data on your phone.



Zhao, Yue, et al. "Federated Learning with Non-IID Data." arXiv preprint arXiv:1806.00582 (2018).

Optimizing Federated Learning on Non-IID Data with Reinforcement Learning

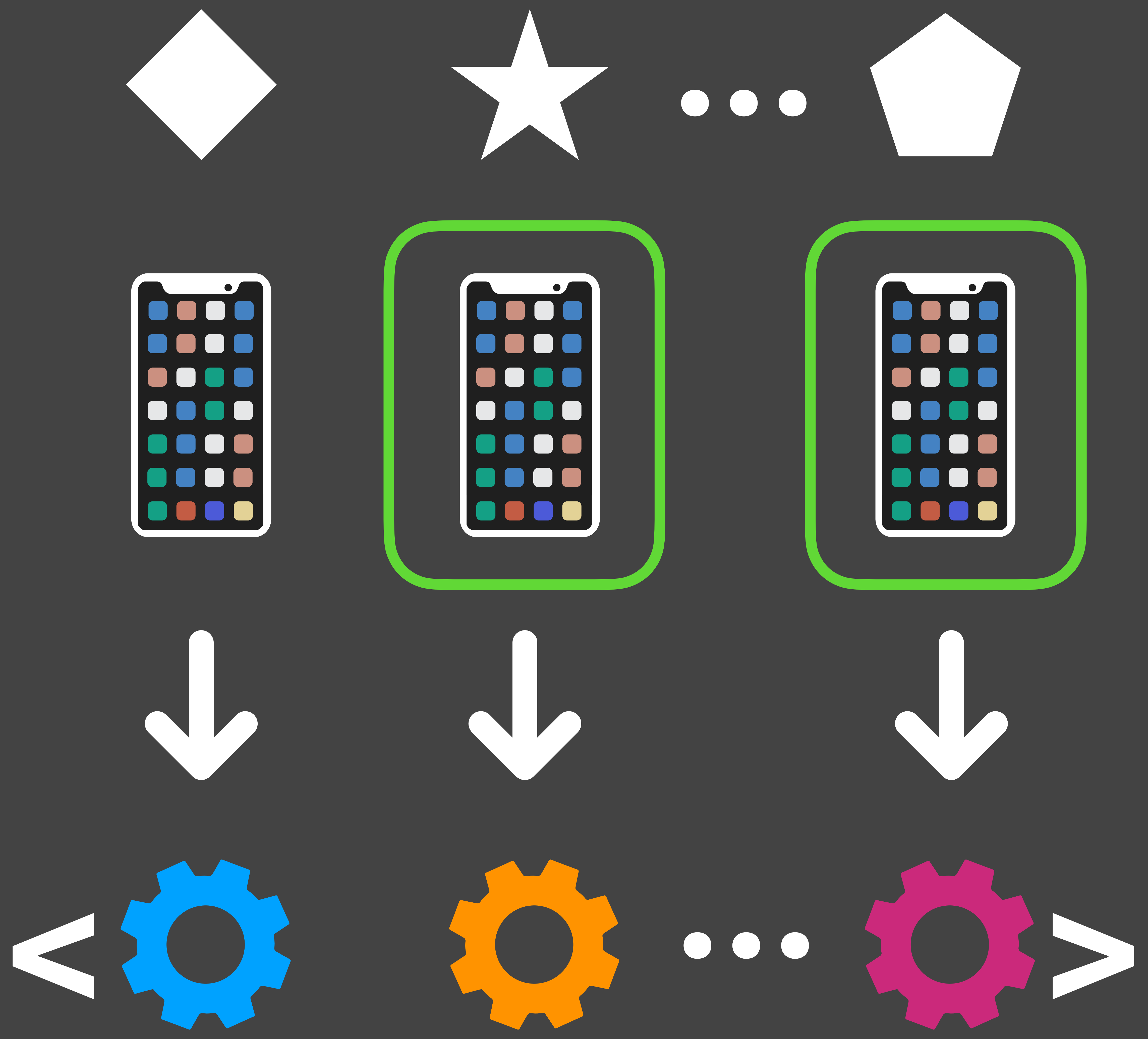
[INFOCOM'20]



Build IID training data? No

Peeking into the data distribution on each device without violating data privacy

Probing the bias of non-IID data



Carefully **select devices** to **balance** the bias introduced by **non-IID data**

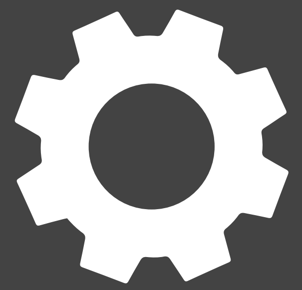
Probing the data distribution

100 devices, each has 600 samples

Non-IID data

80% data has the same label, e.g, “6”

Initial model

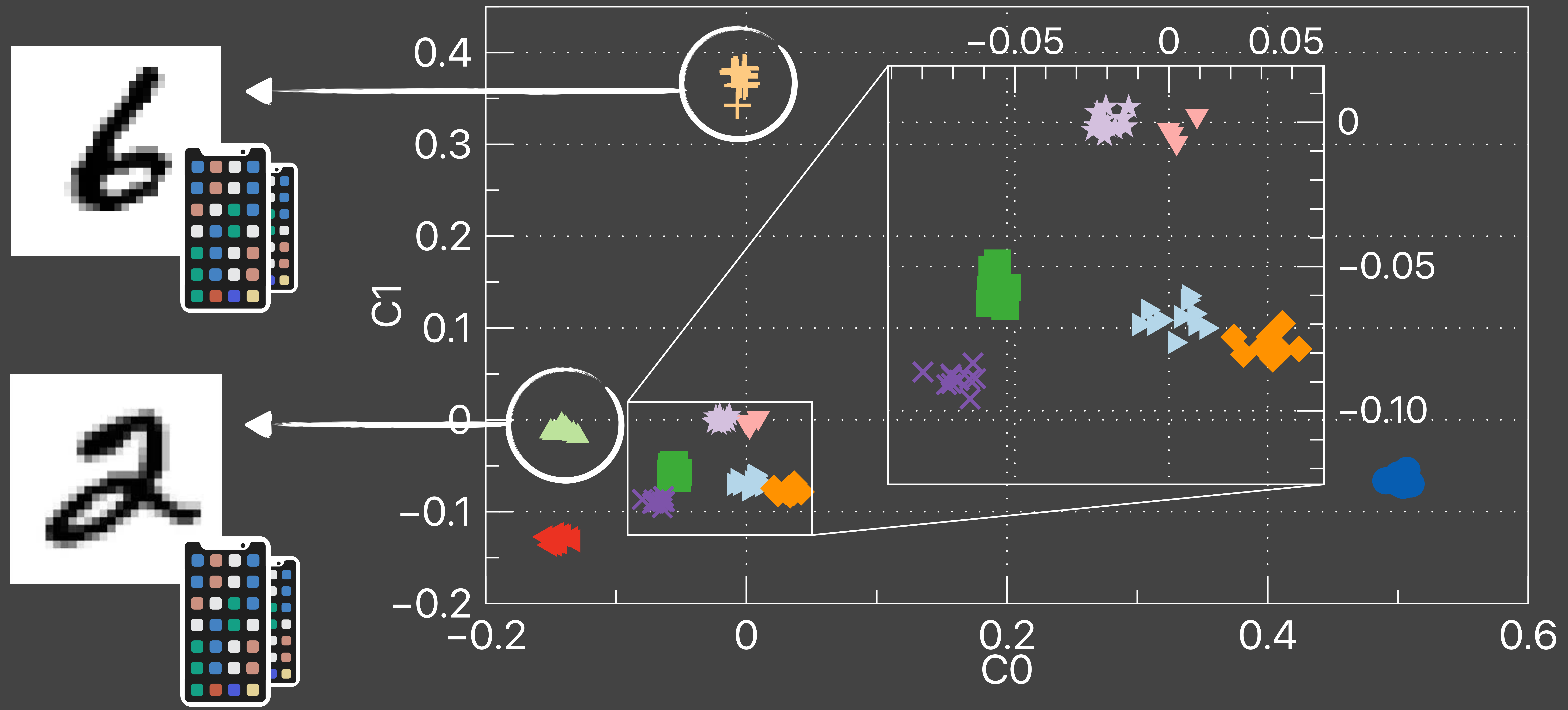


A two-layer CNN model with 431,080 parameters

Local model

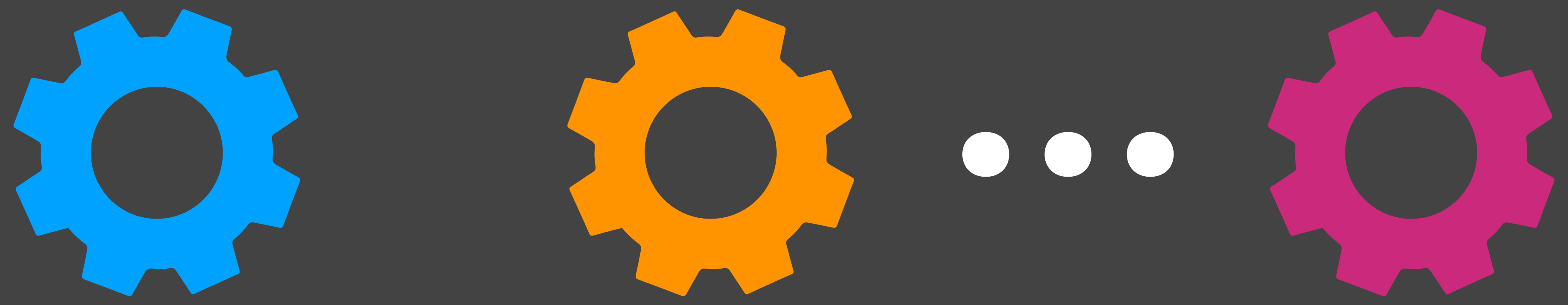
We apply **Principle Component Analysis (PCA)** to reduce dimensionality

431,080-dimension model weight → 2-dimension space



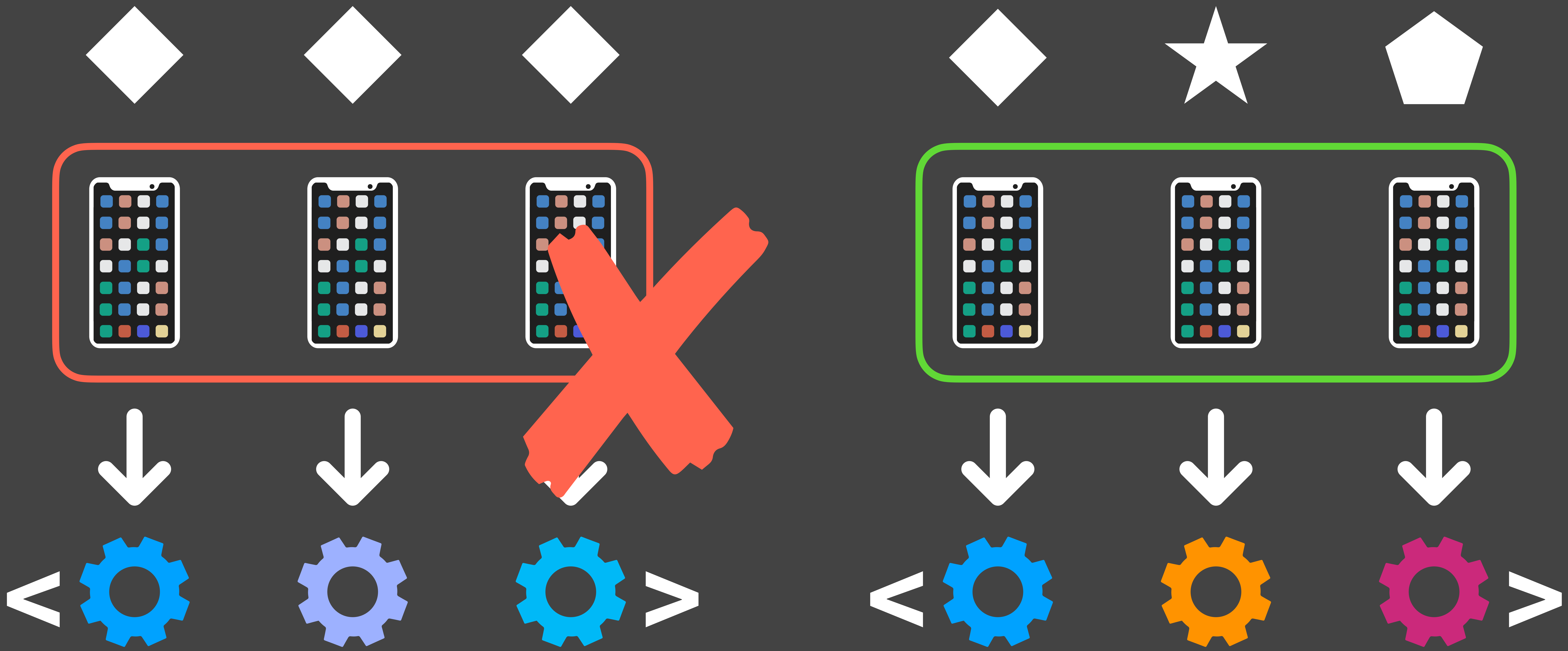
0.4
0.3
0.2
0.1
0
−0.1
−0.2
C1
−0.2
0
0.2
0.4
0.6
C0
−0.05
0
0.05
0
−0.05
−0.10

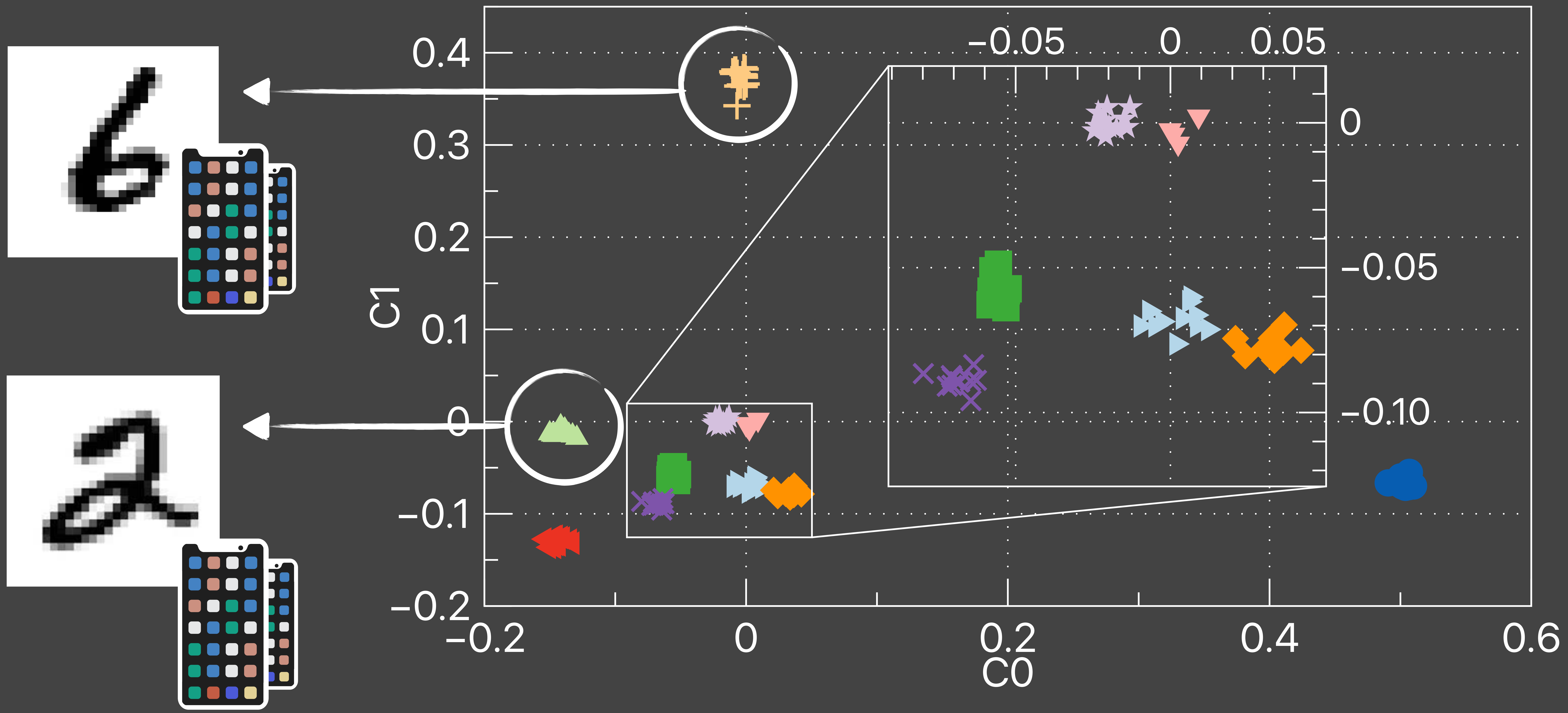
An **implicit connection** between
model weights and **data distribution**



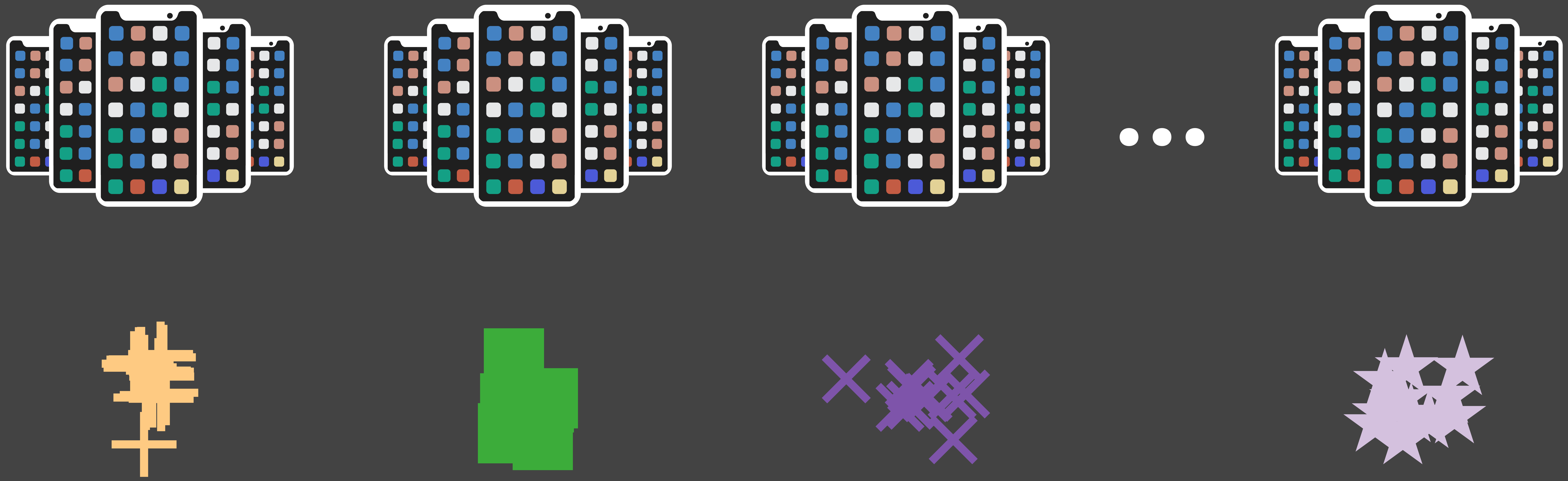
Probing the data distribution

Selecting devices for federated learning

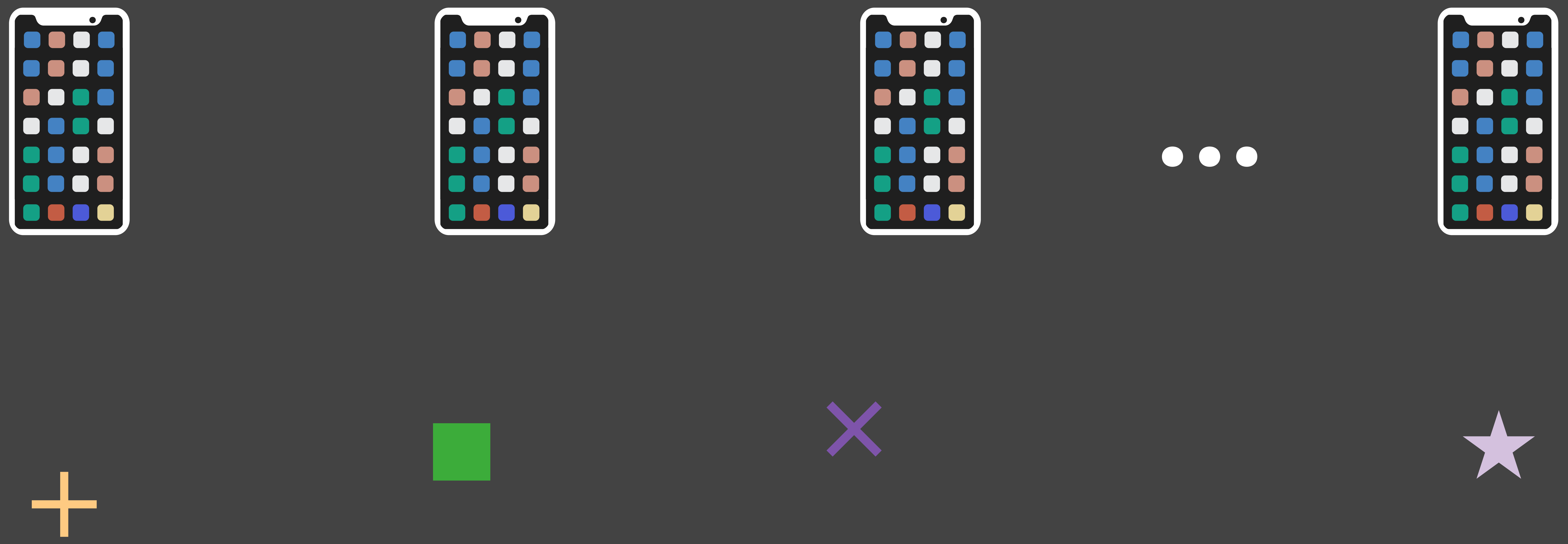


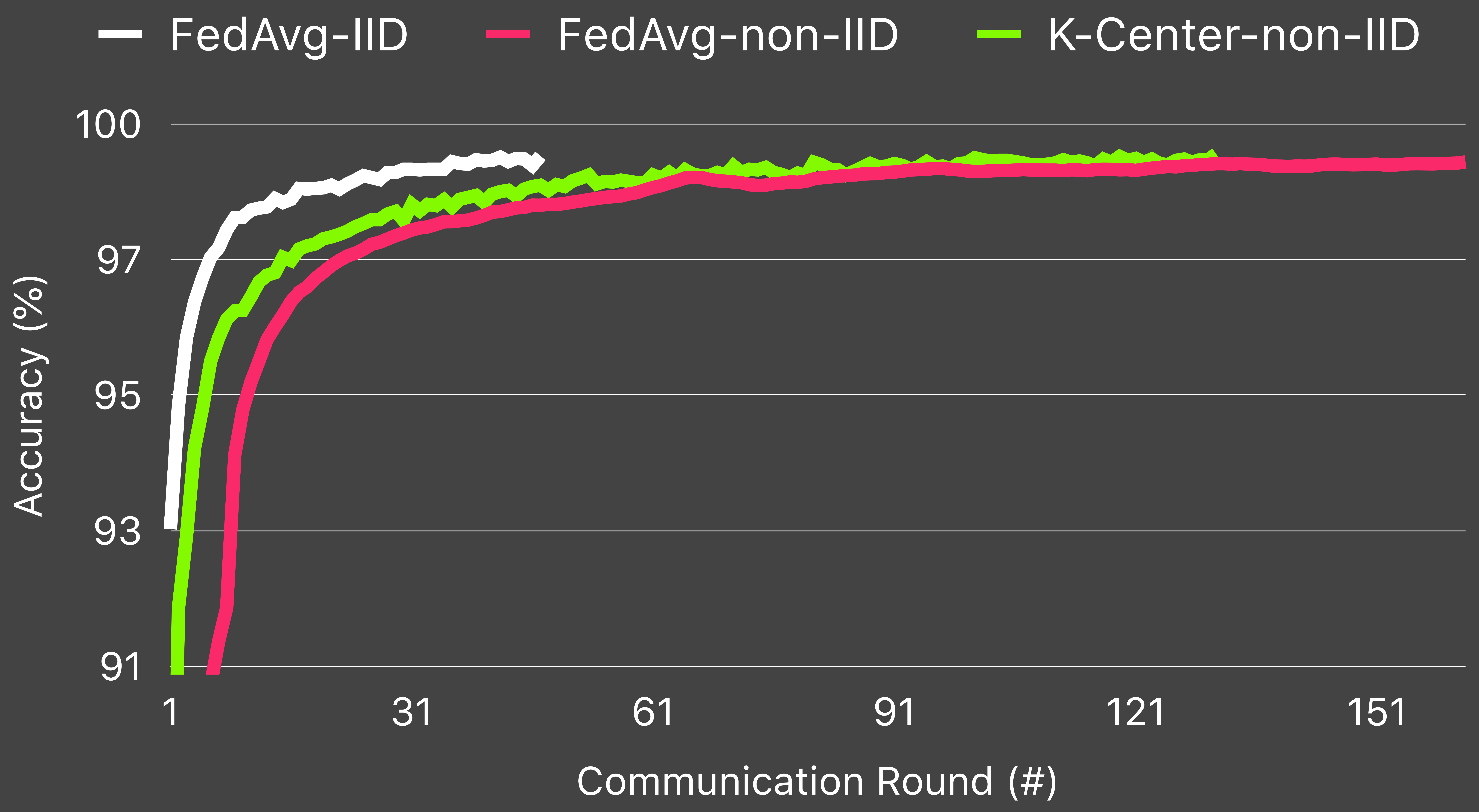

0.4
0.3
0.2
0.1
0
−0.1
−0.2
C1
−0.2
0
0.2
0.4
0.6
C0
−0.05
0
0.05
0
−0.05
−0.10

K-Center Clustering



Random Selection from Groups




FedAvg-IID
FedAvg-non-IID
K-Center-non-IID
Accuracy (%)
100
97
95
93
91
1
31
61
91
121
151
Communication Round (#)

Probing the data distribution

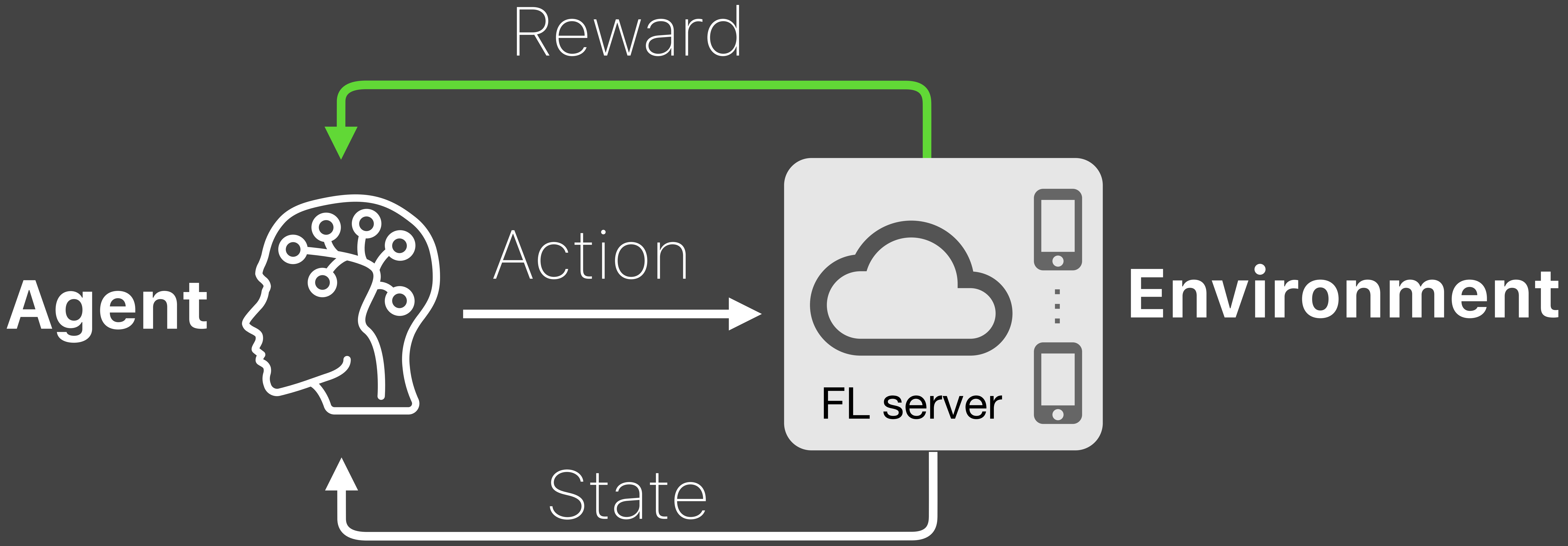
Selecting devices for federated learning

How to select devices to speed up training ?

It is difficult to select the appropriate subset of devices

- Model weights —> device selection choice
- A dynamic and undeterministic problem

Reinforcement Learning (RL)



(…,state, action, reward, state′, action′, …,end)

Episode

(…,state, action, **reward**, state′, action′, …,end)

(…,state, action, **reward**, state′, action′, …,end)

(…,state, action, **reward**, state′, action′, …,end)

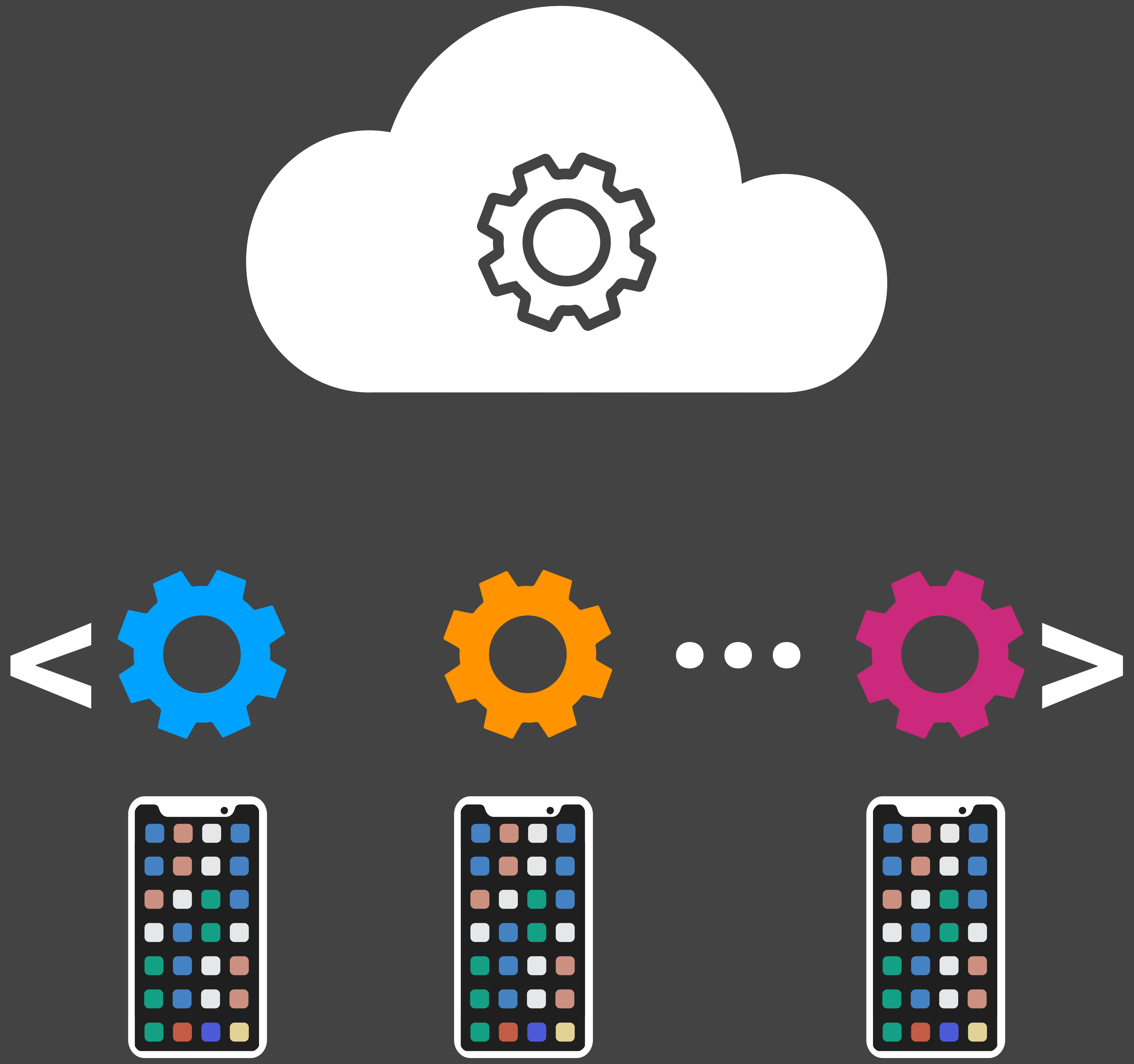
Learn to maximize `sum(reward)`

⋮

(…,state, action, **reward**, state′, action′, …,end)

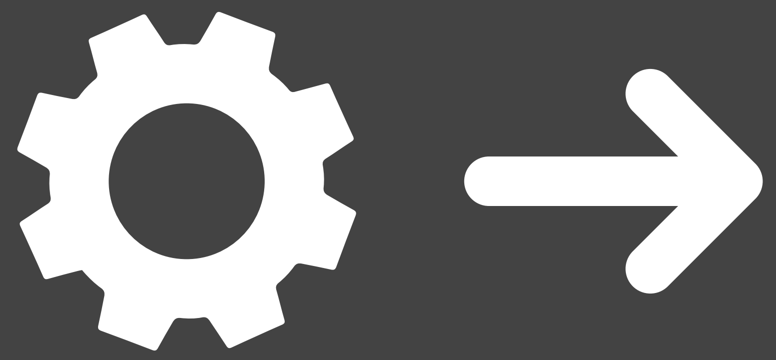
(…,state, action, **reward**, state′, action′, …,end)

States



Global weights

Local model weights



100-dimension vector

Actions

Select K devices from a pool of N devices — a huge action space

Selecting 10 devices from a pool of 100 devices leads to

1.7310309e+13 possible actions

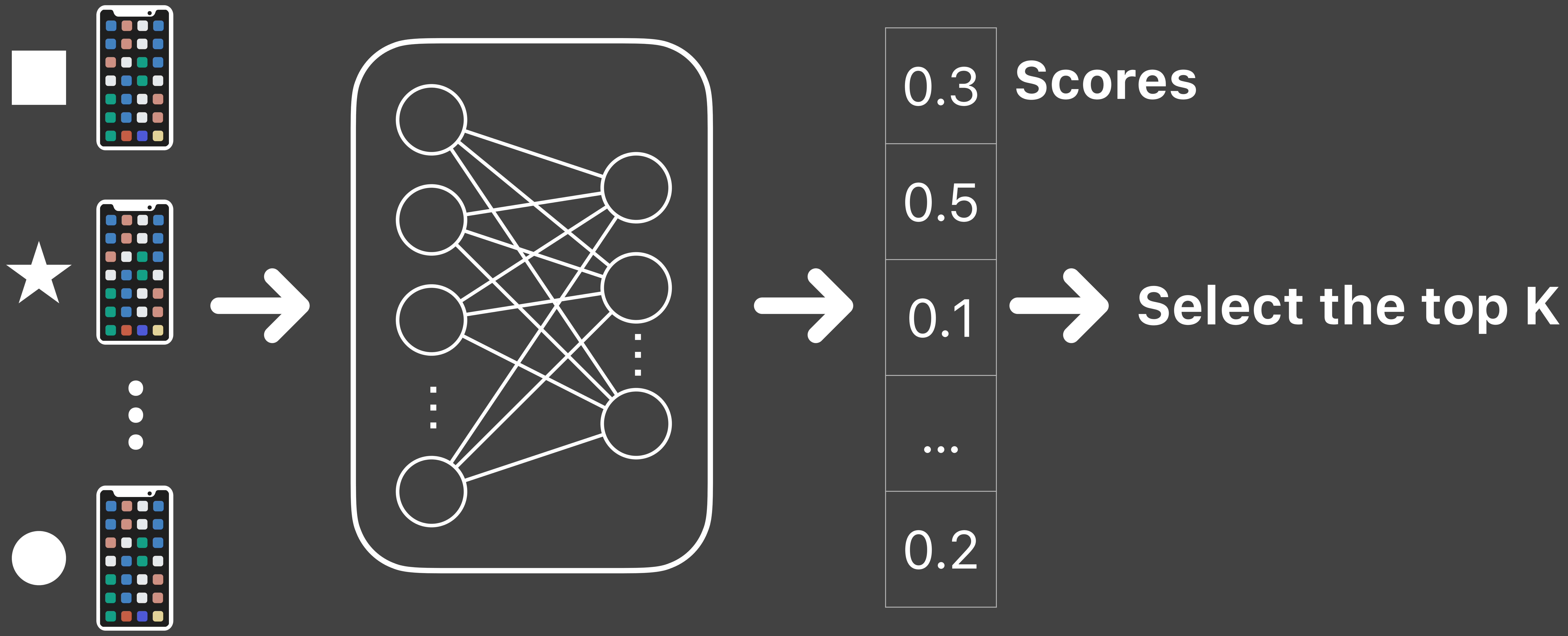
Modify the RL training algorithm

Selecting the Top K Devices

Only one device is selected during the RL training

Now the action space is **{1, 2, ..., N}**, instead of selecting K devices from N devices

Evaluating Each Device



Rewards

$$r_t = \Xi^{(\omega_t - \Omega)} - 1$$

$$0 \leqslant \omega_t \leqslant \Omega \leqslant 1$$

$$r_t \in (-1, 0]$$

Ξ	Positive constant
ω_t	Training Accuracy
Ω	Target accuracy
t	Communication round #

👍 **Accuracy increase:** ω_t⬆ —> r_t⬆

👎 **More communication rounds:** t⬆ —> **sum**(r_t)⬇

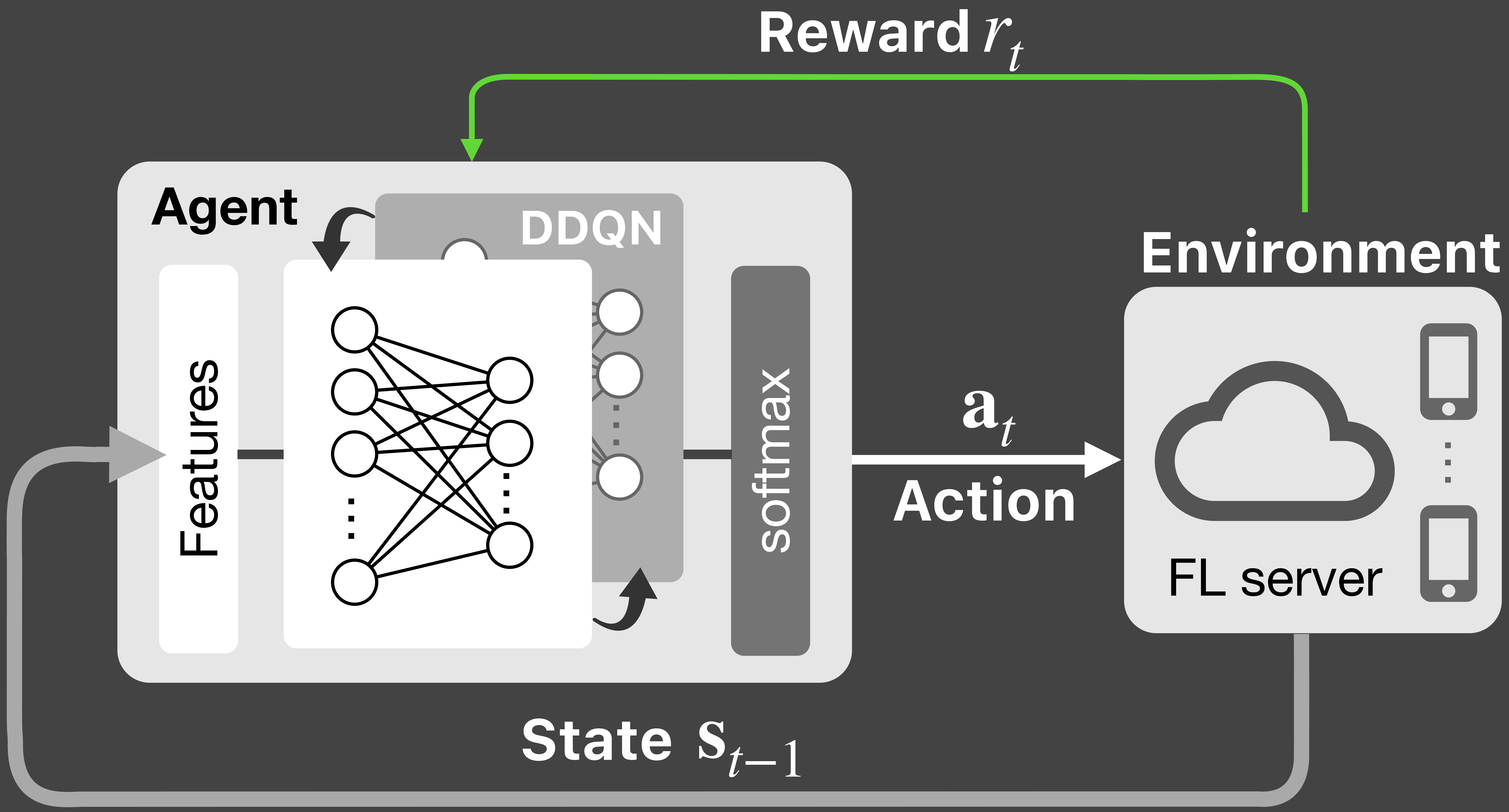
Training the DRL Agent

Look for a **function** that points out the **actions** leading to the maximum cumulative **return** under a particular **state**

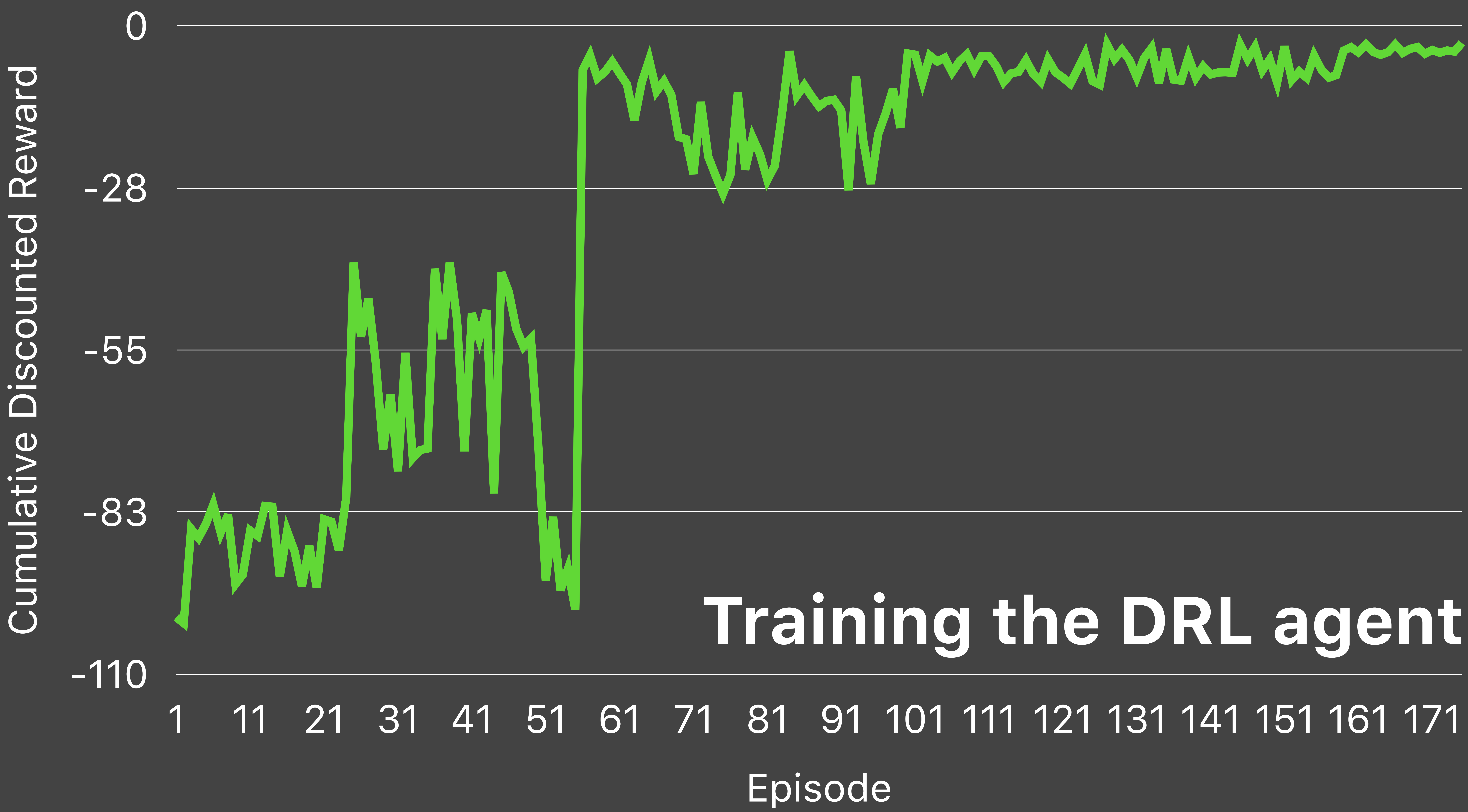
Max $R = \sum_{t=1}^{T} \gamma^{t-1} r_t = \sum_{t=1}^{T} \gamma^{t-1} (\Xi^{(\omega_t - \Omega)} - 1)$

discount factor

$\gamma \in (0,1)$



Reward r_t
Agent
DDQN
Features
softmax
$\mathbf{a}_t$
Action
Environment
FL server
State s_{t-1}


Training the DRL agent
Cumulative Discounted Reward
0
-28
-55
-83
-110
1
11
21
31
41
51
61
71
81
91
101
111
121
131
141
151
161
171
Episode

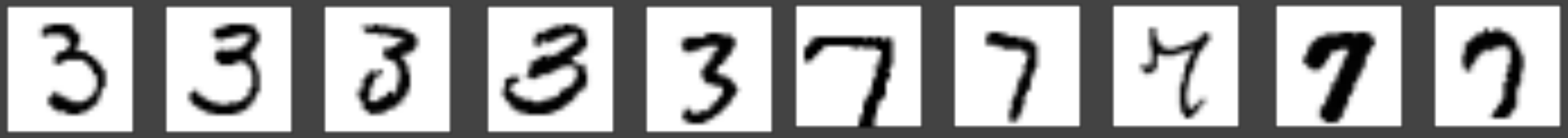

Check-in
Selection
Update
Probing
Update weight
DRL agent
DRL agent

Evaluating Our Solution

Benchmark: MNIST, FashionMNIST, CIFAR-10

Non-IID level: 1, half-and-half, 80%, 50%

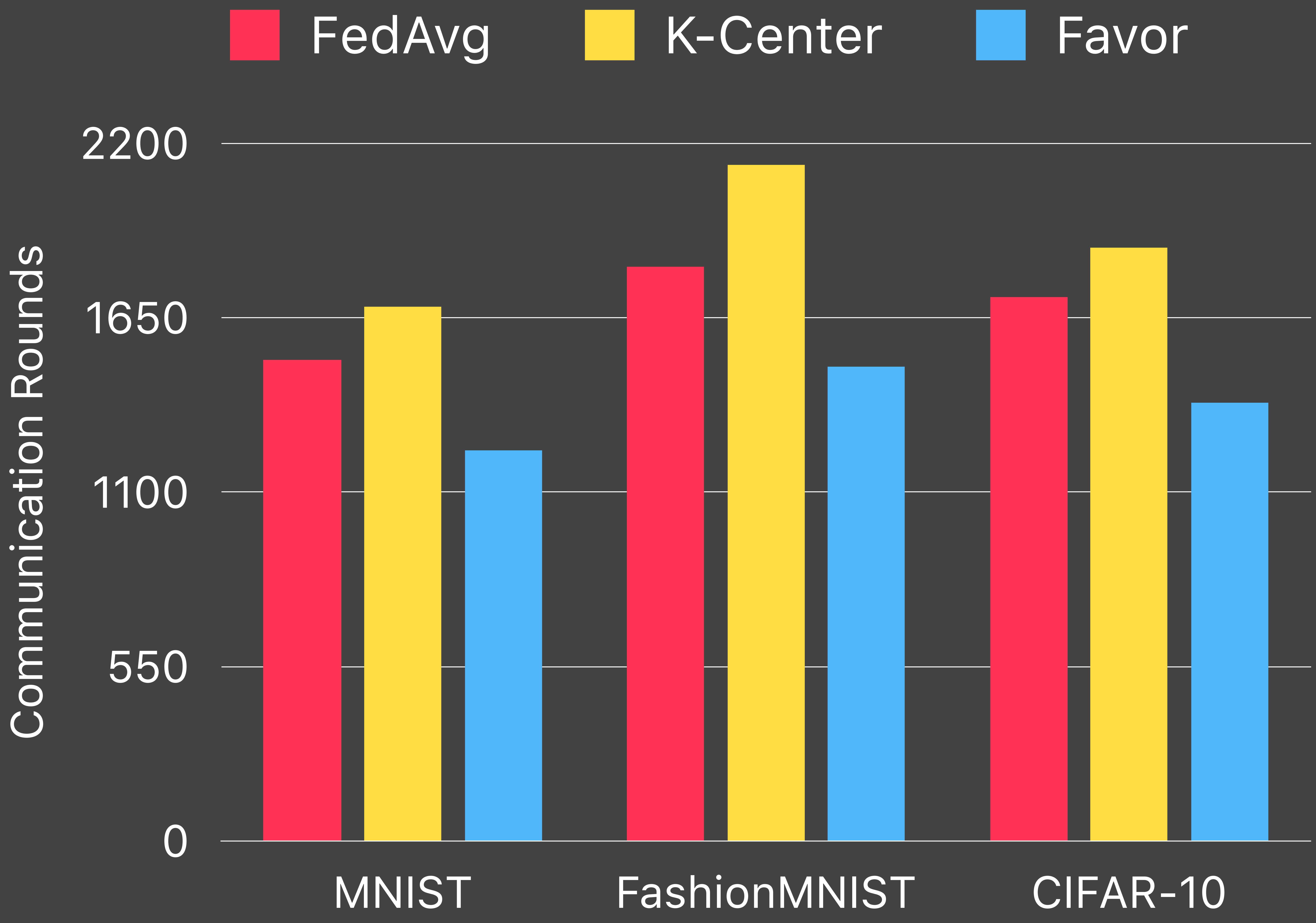
Half-and-half



80%

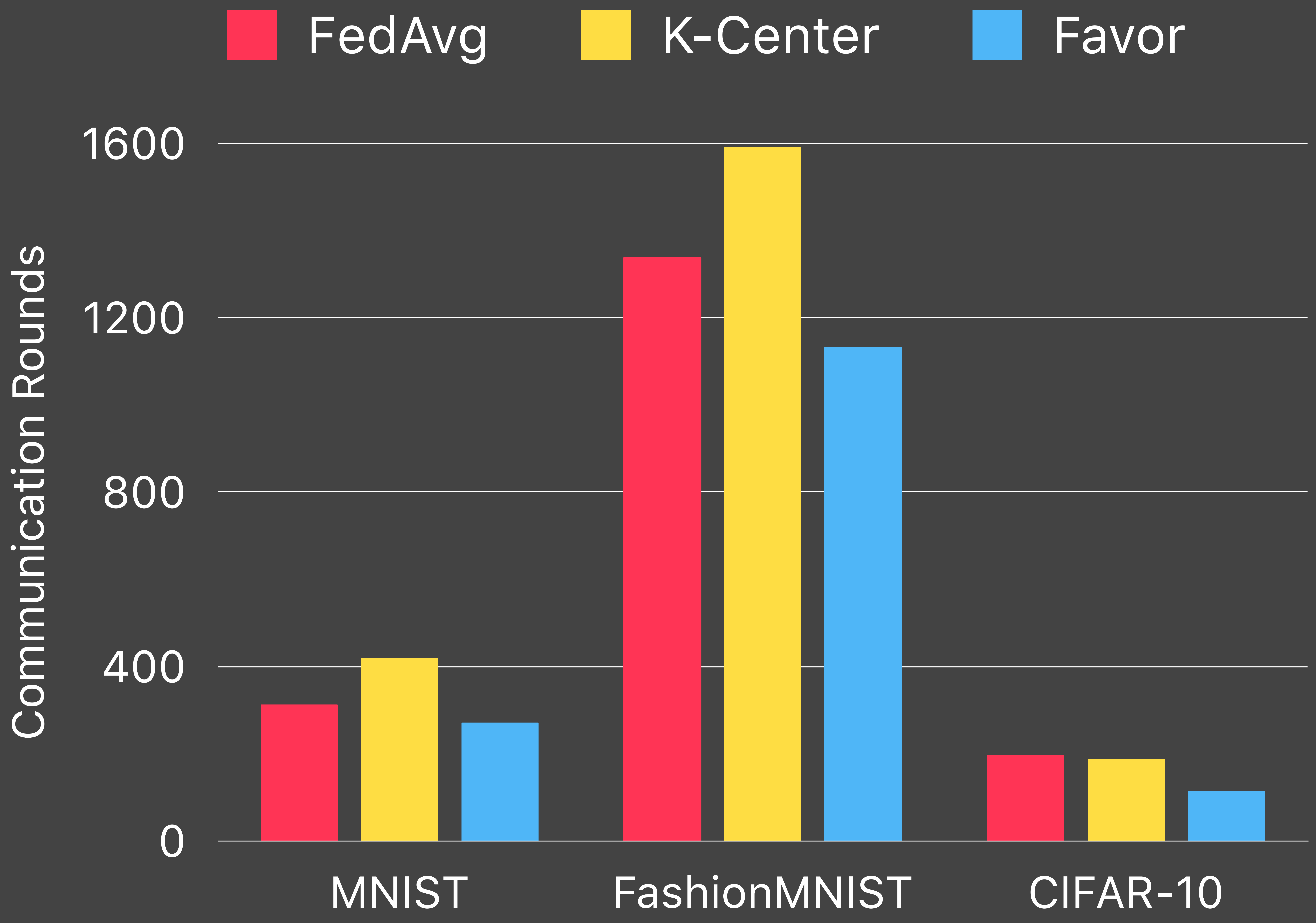
Non-IID level

1



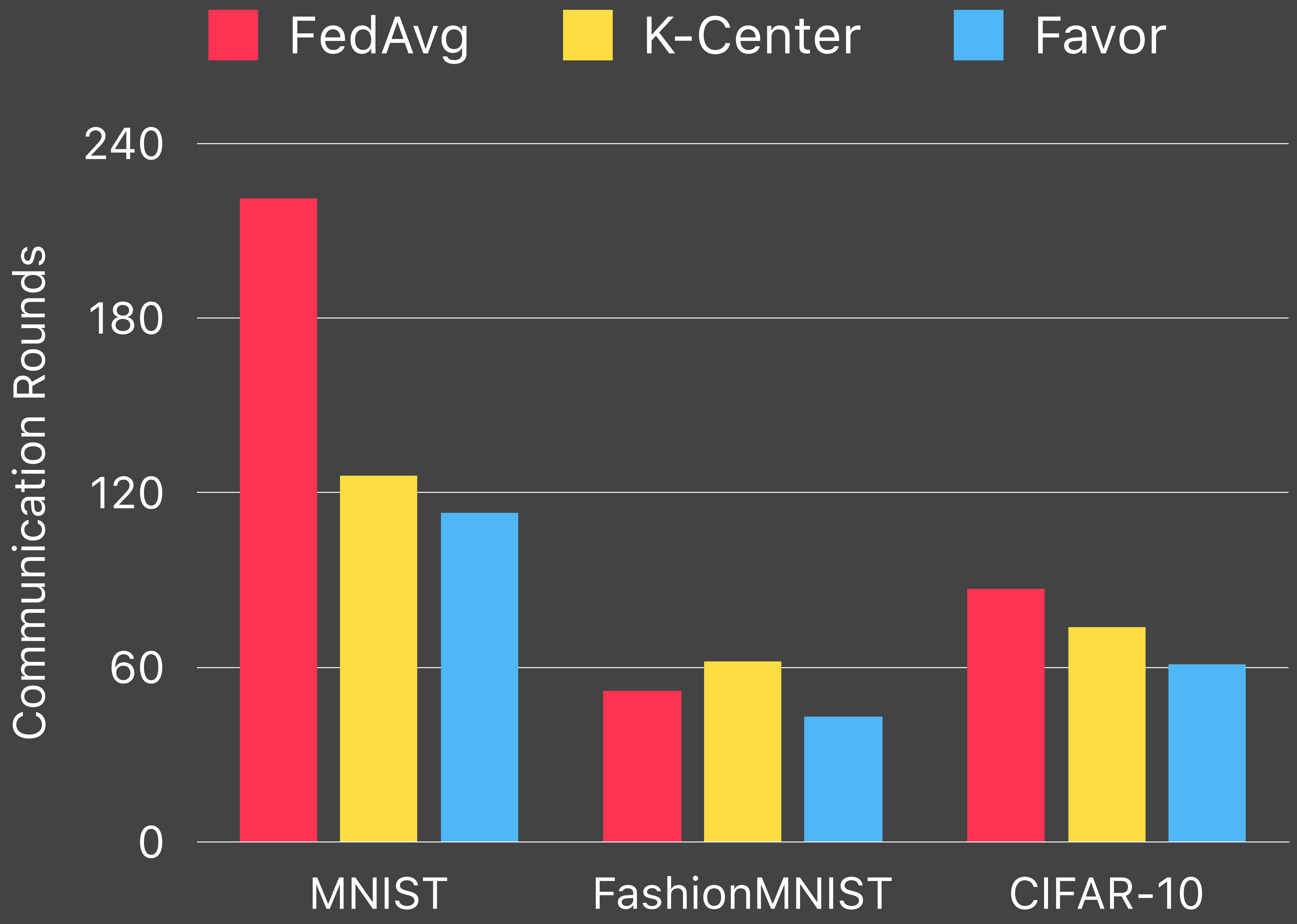
Non-IID level

half & half



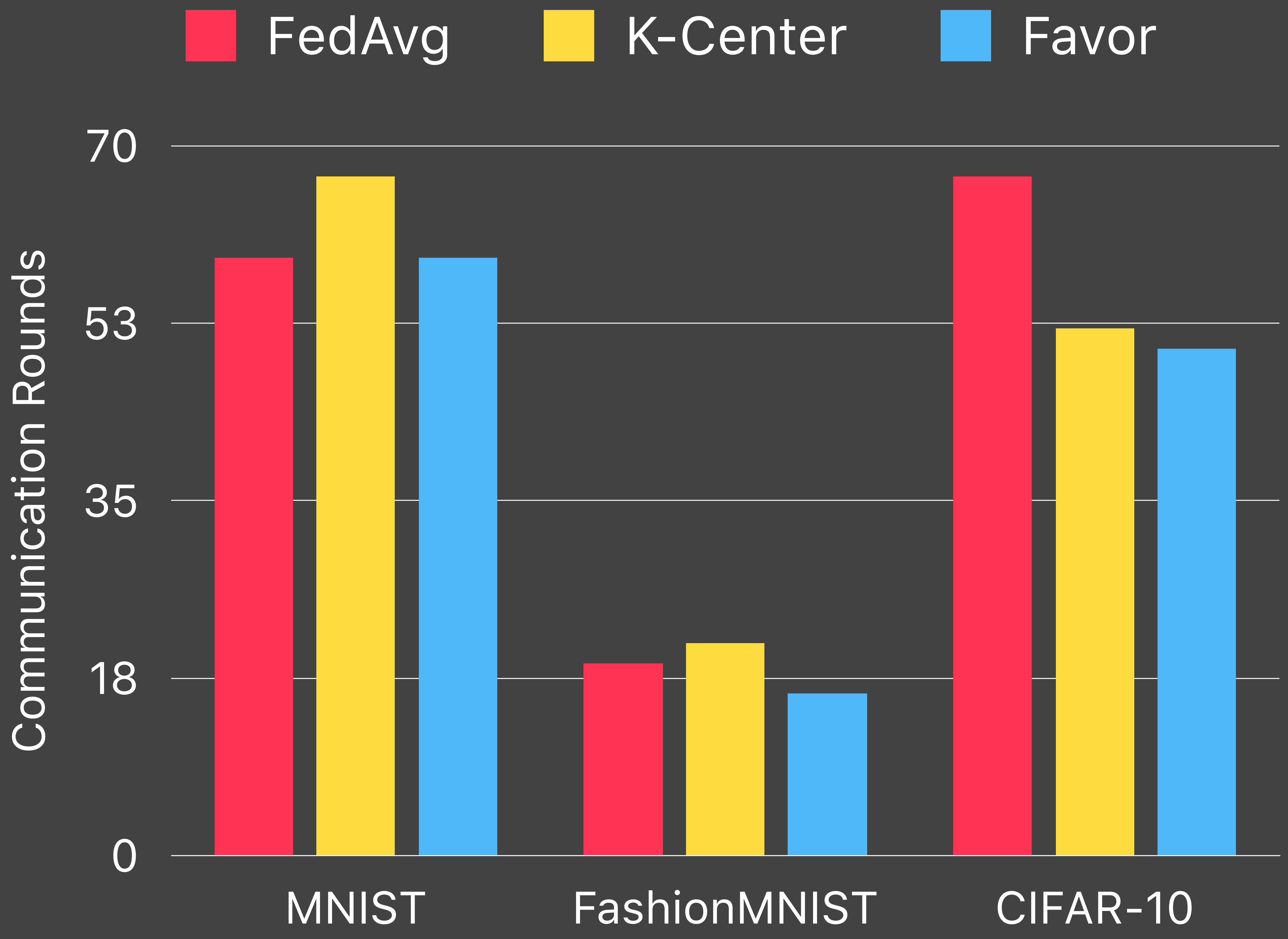
Non-IID level

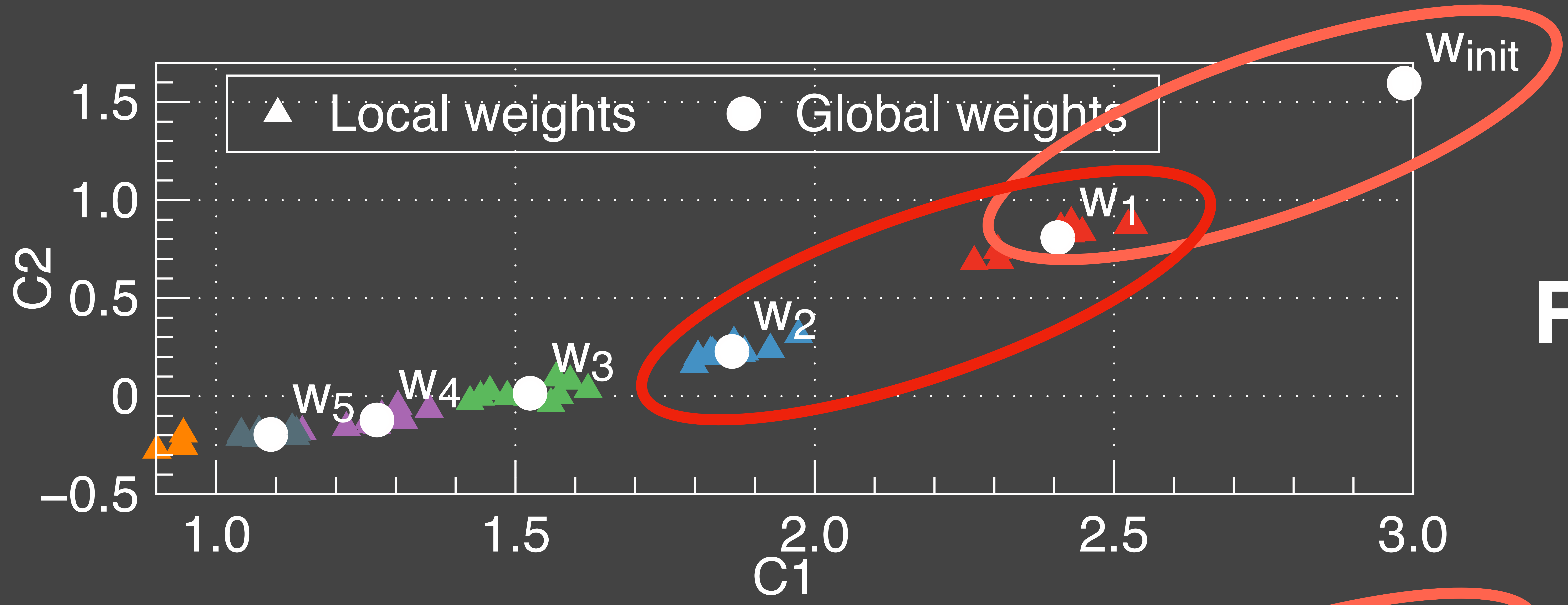
80%



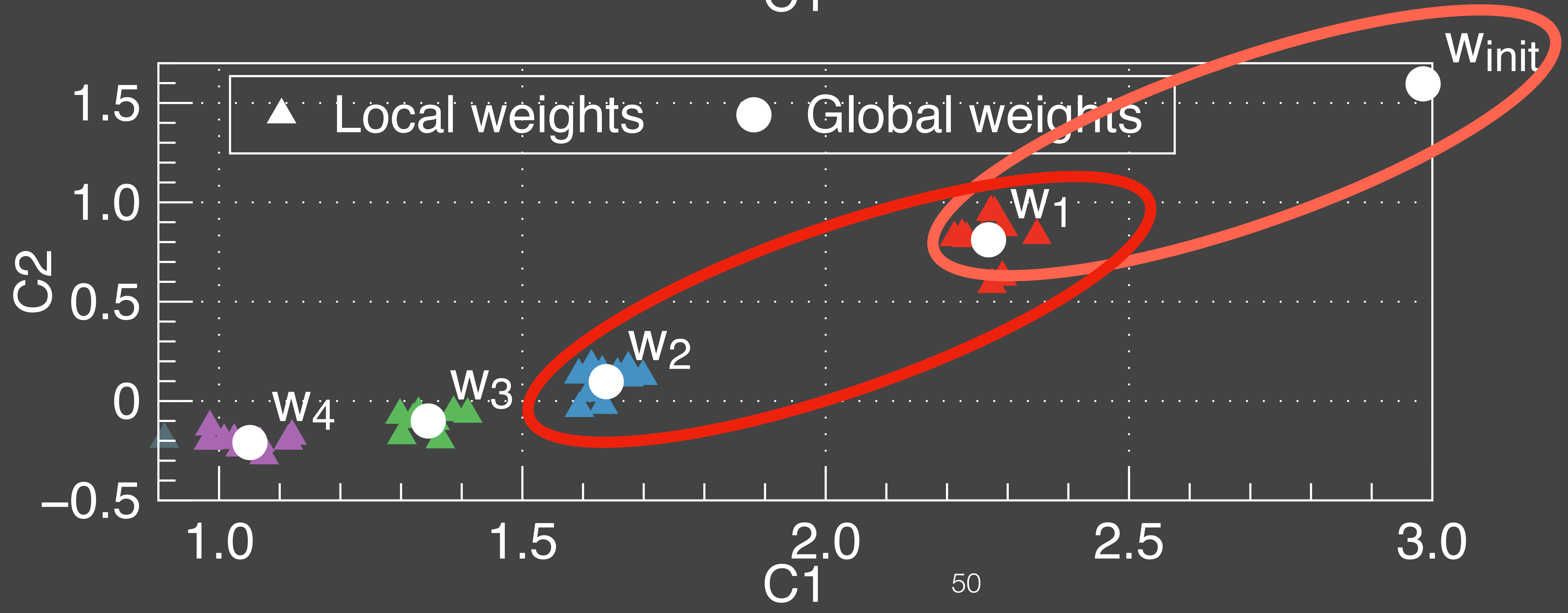
Non-IID level

50%





FedAvg



Favor

Indirect data distribution probing

DRL-based device selection

Communication rounds can be reduced by up to

- **49% on the MNIST**
- **23% on FashionMNIST**
- **42% on CIFAR-10**