Multi Agent RL

e.g. in task offloading: each end user has to make a decision to offload data……each end User Device is considered an RL agent

In this case,

CS will be agent….	Select Edge Nodes (EN)
each Edge Node will be an agent….	select a subset of end User Devices

Action Space:

Each Agent …….. select k devices to participate in FL ……epsilon-greedy

epsilon-greedy method (instead of top-k in Infocom Paper)

1-epsilon ……select top-k

Epsilon……select a random node to participate

top-k……..if k =3, after selecting top-1, recalculate Q, and select top-2, then recompute Q and select top-3….

Dimension reduction: As in original paper

Latency: For each device: Latency (phi) MEASURED at CS/EN when send/receive model weights

Reward Function to incorporate latency …… (w – omega – phi)

Phi for latency penalty…… [0 <= phi <= 1]

Phi (latency value) should be between 0 and 1, and normalized so that it does not change the reward function too much

With Central Server ……. N devices directly connecting to CS

With K Edge ……

K Edge Connect to CS

N/K approximately connect to each Edge

Utility Function: compare local update of each EN k with global update to measure their distance as

$$u_{E_k}^{F_i} = \left| w^{t+1} - w_k^{t+1} \right|$$

Higher distance between the ENs's local update and global update, shows less/low contribution to the global update by this EN, which ultimately means that this EN makes low contribution to model learning and thus has lower utility.

For Selection of K:
#Central Server considers Edge Nodes (EN) , among Edge Nodes assign weight to each Edge Node,
compare local update of each EN J with global update to find the utility of each EN....
(calculate distance between local update of each EN J with global update, for example "Euclidean distance")

$$u_{E_k}^{F_i} = \left| w^{t+1} - w_k^{t+1} \right|$$

utility between 0 and 1.....all utilities add up to 1
100% .... 10 ENs ... utility x 100%

if total weight is 1 and total 10 ENs,
EN1 get weight 20%
EN2 15%, ,.., ...
EN1 select 4 devices, EN2 select 3 devices...
cs helps based on global knowledge,

The original work uses a two tier structure for the federated mobile edge network, with two components being client devices and central server, and the client devices directly connect with the central server. This scheme can suffer from high communication latency for reasons like, higher distance between client devices and the central server, client devices having slow and/or unreliable network connection etc. Further, scalability is another important shortcoming of this approach, especially when enormous number of client devices want to connect directly to the central server, or the client devices are far away from the central server.

The original work does not take into account the communication latency for selection of the client devices in each round to participate in the Federated learning training. Communication latency can be a crucial factor given the slow and/or unreliable network connection of client devices. Some of the devices may be very slow as compared to the other devices, and if such a device is selected, this can increase the time it takes to converge.

Whereas in this work we consider a three tier mobile edge network, with the introduction of edge nodes as the middle layer between the central server and the client devices. The client devices can connect to the closest edge node. This can help in both reducing the communication latency and increasing scalability.

For communication latency, the edge nodes are closer to the client devices as compared to the central server, which results in faster communication and reduction in resource consumption of the client devices.

As compared to only one central server, multiple edge nodes allow for a larger number of client devices to be connected to the network, and result in more scalable network. The workload is also distributed to the multiple edge nodes and the central server, which can result in better overall performance.

In this work, we also introduce a penalty for the communication latency, so as to penalize the slow devices or the devices with unreliable network connection. The reward received by slower devices will be reduced by the penalty, resulting in less chances of being selected in the next round.

In this work, we introduce the utility of edge nodes or the contribution they make to the global model update. In the original work, action-value or Q-function value is used to select the top-k client devices in each round to participate in the next round of training, whereas in our work, the central server allocates k for each of the edge nodes based on the contribution of the edge node to the global model.