# Federated Reinforcement Distillation with Proxy Experience Memory

# Federated Reinforcement Distillation with Proxy Experience Memory

**Han Cha**[1*], **Jihong Park**[2], **Hyesung Kim**[1], **Seong-Lyun Kim**[1] and **Mehdi Bennis**[2]

[1]School of Electrical and Electronic Engineering, Yonsei University, Seoul, Korea

[2]Centre for Wireless Communications, University of Oulu, Finland

[1]{chan, hskim, slkim}@ramo.yonsei.ac.kr, [2]{jihong.park, mehdi.bennis}@oulu.fi

## Abstract

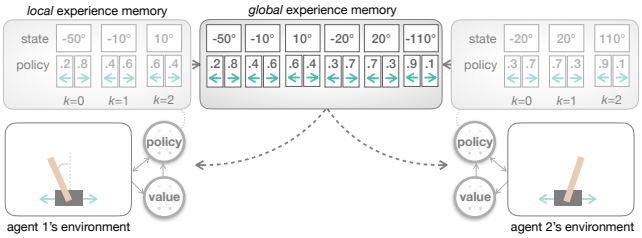
In distributed reinforcement learning, it is common to exchange the experience memory of each agent and thereby collectively train their local models. The experience memory, however, contains all the preceding state observations and their corresponding policies of the host agent, which may violate the privacy of the agent. To avoid this problem, in this work we propose a privacy-preserving distributed reinforcement learning (RL) framework, termed *federated reinforcement distillation (FRD)*. The key idea is to exchange a *proxy experience memory* comprising a pre-arranged set of states and time-averaged policies, thereby preserving the privacy of actual experiences. Based on an advantage actor-critic RL architecture, we numerically evaluate the effectiveness of FRD, and investigate how the performance of FRD is affected by the proxy memory structure and different memory exchanging rules.

## 1 Introduction

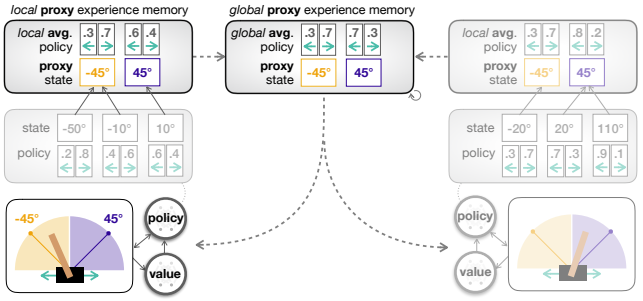
Recent advances in mobile computing power has led to the emergence of intelligent autonomous systems [Park *et al.*, 2018, Shiri *et al.*, 2019], ranging from driverless cars and drones to self-controlled robots in smart factories. Each agent therein interacts with its environment, and carries out decision-making in real time. Distributed deep reinforcement learning (RL) is a compelling framework for such applications, in which multiple agents collectively train their local neural networks (NNs). As illustrated in Figure 1(a), this is often done by: (i) uploading every local *experience memory* to a server, (ii) constructing a global experience memory at the server, and (iii) downloading and replaying the global experience memory at each agent to train its local NN [Rusu *et al.*, 2016]. However, a local experience memory contains all local state observations and the corresponding policies (i.e., action logits), and exchanging this may violate the privacy of its host agent.

To obviate this problem, we propose a distributed RL framework based on a *proxy experience memory*, which is termed *federated reinforcement distillation (FRD)* and depicted in Figure 1(b). In contrast to conventional experience memories containing actual states and policies, a local proxy experience memory at each agent consists of a set of pre-arranged *proxy states* and *locally averaged policies*. In this memory structure, the actual states are mapped into the proxy states, e.g., based on the nearest value rule, and the actual policies are averaged over time. Exchanging the local proxy memories of agents thereby preserves their privacy by hiding each agent's actual experiences from the others.

In this work, we consider actor-critic RL architecture comprising two separate NNs, i.e., policy (actor) and and value (critic) NNs, and study how to construct the local and global proxy memories, how often the proxy memories are exchanged, and finally how to update each agent's local NN using the global proxy memory.




(a) Policy distillation with experience memory.




(b) FRD with proxy experience memory.

Figure 1: Comparison between (a) a baseline distributed reinforcement learning (RL) framework, policy distillation with experience memory [Rusu *et al.*, 2016], and (b) the proposed *federated reinforcement distillation (FRD)* with *proxy experience memory*.

[*]Contact Author

**Related Works**. Distributed deep RL has been investigated as policy distillation [Rusu *et al.*, 2016] and advantage actor-critic (A2C) [Mnih *et al.*, 2016] algorithms, under policy NN and actor-critic based RL architectures, respectively. Both algorithms rely on exchanging actual experience memories. For classification tasks, distributed machine learning via exchanging NN outputs has been proposed as federated distillation (FD) in our preceding work [Jeong *et al.*, 2018]. In FD, the outputs are quantized based on the classification labels, for maximizing communication efficiency. FRD leverage and extend this idea to distributed RL scenarios, in the context of its preserving privacy, rather than improving communication efficiency. It is noted that federated learning [McMahan *et al.*, 2017] is another promising enabler for private distributed RL by exchanging NN model parameters, which has been recently studied as federated reinforcement learning (FRL) in [Zhuo *et al.*, 2019]. In view of this, we conclude this paper by comparing FRL and our proposed FRD in the last section.

## 2 Background: Distributed Reinforcement Learning with Experience Memory

We consider the episodic, distcrete state and action space Markov decision process, with state space $\mathcal{S}$, action space $\mathcal{A}$ and reward at each time slot denoted by $r_t \in \mathbb{R}$. The policy is stochastic and denoted by $\pi_\theta : \mathcal{S} \to \mathcal{P}(\mathcal{A})$, where $\mathcal{P}(\mathcal{A})$ is the set of probability measures on $\mathcal{A}$. The parameters of local model are denoted by $\theta \in \mathbb{R}^n$, and $\pi_\theta(a|s)$ is the conditional probability of $a$ when the state is $s$. The reinforcement learning (RL) interacts with the environment without any prior knowledge about the environment.

In policy distillation presented in [Rusu *et al.*, 2016], the agents $i = 1, \cdots, U$ construct the dataset named *experience memory* for training local model $\theta_i$. The experience memory $\mathcal{M} = \{(s_k, \pi(\mathbf{a}_k|s_k))\}_{k=0}^{N}$ consists of the state $s_k$ and the policy vector $\pi(\mathbf{a}_k|s_k)$ tuple, where $\mathbf{a} = (a^1, \cdots, a^{|\mathcal{A}|})$ is action vector. As illustrated in Figure 1(a), the experience memroy $\mathcal{M}$ is collected with following procedures.

- Each agent records the *local experience memory* $\mathcal{M}_i = \{(s_k, \pi_{\theta_i}(\mathbf{a}_k|s_k))\}_{k=0}^{N_i}$ tuple during $E$ episodes. The size of local experience replay $N_i$ is identical with the learning steps. In this paper, we assume that all the agents wait for the last agent completing the episode.
- After all the agents complete the $E$ episodes, the server collects $\mathcal{M}_i$ of each agent.
- Then, the server constructs a *global experience memory* $\mathcal{M} = \{(s_k, \pi(\mathbf{a}_k|s_k))\}_{k=0}^{N}$, where $N = \sum_{i=1}^{U} N_i$ and $\pi$ denotes the arbitrary policy of agents.

After the global experience memory $\mathcal{M}$ is constructed, the agents update their local model $\theta_i$ with following procedures.

- To reflect the knowledge of other agents, the agents download the global experience memory $\mathcal{M}$ from the server.
- Similar to the conventional classification setting, each agent $i$ fits the local model $\theta_i$ minimizing the cross entropy loss $L_i(\mathcal{M}, \theta_i)$ between the policy of local model $\pi_{\theta_i}(\mathbf{a}_k|s_k)$ and the policy $\pi$ of global experience memories $\mathcal{M}$, where

$$L_i(\mathcal{M}, \theta_i) = -\sum_{k=1}^{N} \pi(\mathbf{a}_k|s_k) \log\left(\pi_{\theta_i}(\mathbf{a}_k|s_k)\right). \quad (1)$$

Unfortunately, direct exchaging the local experience memories of agents has the privacy leakage issues. The server can get the all information about the state visited by the host agent and the corresponding policy of the host agent. To utilize the policy distillation, privacy leakage is inevitable.

## 3 Federated Reinforcement Distillation (FRD) with Proxy Experience Memory

In this section, we introduce the novel federated reinforcement distillation (FRD) method that provides communication-efficient privacy-preserving federated reinforcement distillation. The agents utilizing the FRD construct the novel dataset named *proxy experience memory* $\mathcal{M}^P = \{(s_k^p, \pi^p(\mathbf{a}_k|s_k^p)\}_{k=0}^{N^P}$, where the $s^p$ denotes the *proxy state* and the $\pi^p(\mathbf{a}_k|s_k^p)$ denotes *average policy*. The proxy state is representative state of *state cluster* $C_j \in \mathcal{C}$. Note that the union of proxy state cluster sets is the state space $\mathcal{S}$, i.e., $\mathcal{S} = \bigcup_{j=1}^{|\mathcal{C}|} C_j$ and none of the state cluster has the joint set, i.e., $C_i \cap C_j = \emptyset, i \neq j$.

As illustrated in Figure 1(b), the proxy experience memory $\mathcal{M}^P$ is formed with following procedures.

- Each agent categorizes the policy $\pi_{\theta_i}(\mathbf{a}|s)$ along the states $s$ included in the proxy state cluster, i.e., $s \in C_j$.
- After all the agents complete the $E$ episodes, each agent calculates *local average policy* $\pi_{\theta_i}^p(\mathbf{a}_k|s_k^p)$ by averaging the policy $\pi_{\theta_i}(\mathbf{a}|s)$ in the proxy state cluster $C_j$ and make *local proxy experience memory* $\mathcal{M}_i^P = \{(s_k^p, \pi_{\theta_i}^p(\mathbf{a}_k|s_k^p)\}_{k=0}^{N_i^P}$. The size of local proxy experience memory $N_i^P$ is identical with the number of proxy state cluster that visited by the agent. Note that the $\pi_{\theta_i}^p(\mathbf{a}_k|s_k^p)$ is not generated by the local model of agent.
- When the local proxy experience memories of every agent is ready, the server collects $\mathcal{M}_i^P$ of each agent.
- Then, the server constructs the *global proxy experience memory* $\mathcal{M}^P = \{(s_k^p, \pi^p(\mathbf{a}_k|s_k^p)\}_{k=0}^{N^P}$ by averaging the local average policy of local proxy experience memory along the state cluster. The size of global proxy experience memory $N^P$ is identical with the number of proxy state cluster that visited by entire agents.

As same as the policy distillation case, the agents utilizing the FRD update their local model $\theta_i$ with following procedures.

- As the distritubed RL procedure, the agents download the global proxy experience memory $\mathcal{M}^P$ from the server.
- Each agent $i$ fits the local model $\theta_i$ minimizing the cross entropy loss $L_i^P(\mathcal{M}^P, \theta_i)$ between the policy of local model $\pi_{\theta_i}(\mathbf{a}_k|s_k)$ and the global average policy

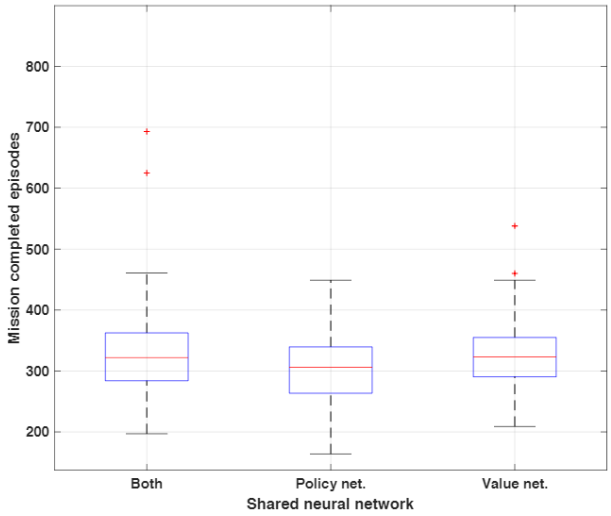



Figure 2: Performance comparison according to exchanging model. The case of exchanging policy network is better than other cases in terms of performance variation.

$\pi^p(s_k^p, \mathbf{a}_k|s_k^p)$ of global proxy experience memory $\mathcal{M}^P$, where

$$L_i^P(\mathcal{M}^P, \theta_i) = -\sum_{k=1}^{N^P} \pi^p(\mathbf{a}_k|s_k^p) \log\left(\pi_{\theta_i}(\mathbf{a}_k|s_k^p)\right). \tag{2}$$

We note that the loss is calculated with the policy produced by the local model as the input of proxy state.

As we refered above, the size of global proxy experience memory is much smaller than that of experience memory due to state clustering. When the memory sharing occurs through wirelee channel, the payload size is key factor of sharing fisibility. In this point of view, FRD provides communication-efficient distributed RL framework.

Furthermore, exchanging the proxy experience memories keeps the privacy of the agents. The server just knows about a group of states that the agent visited and the policy of the host agent is totally concealed due to the policy of agent is shared in the form of averaged policy over the proxy state.

## 4 FRD under Actor-Critic Architectures

The advantage actor-critic (A2C) algorithm [Mnih *et al.*, 2016] consists of two parts, actor and critic NNs. The actor generates the action $a \in \mathcal{A}$ according to the policy $\pi_\theta$ and the critic evaluates selected action how much is beneficial than another actions with respect to gaining more expected future reward. But the actor and the critic have no prior knowledge of environment, the actor-critic pair have to interact with environment and learn optimal policy to getting maximum expected future reward. By adopting neural network structure, the actor and the critic effectively learn optimal policy $\pi^*$.

The advantage function [Wang *et al.*, 2016] is the metric evaluating the action generated by the actor. If the value of the advantage function is positive, it means that the selected action is not the optimal action compare to another actions. In other words, the The advantage function $A$ is defined as follows:

$$\begin{aligned} A^\pi(s_t, a_t) =& Q^\pi(s_t, a_t) - V^\pi(s_t) \\ =& r(s_t, a_t) + \mathbb{E}_{s_{t+1}\sim\mathbb{E}}\left[V^\pi(s_{t+1})\right] - V^\pi(s_t) \\ \approx& r(s_t, a_t) + V^\pi(s_{t+1}) - V^\pi(s_t). \end{aligned} \tag{3}$$

where $Q^\pi(s, a) = \mathbb{E}\left[r_0^\gamma | s_0 = s, a_0 = a; \pi\right]$, the value function $V^\pi(s) = \mathbb{E}[r_0^\gamma | s_0 = s; \pi]$, and $r(s_t, a_t)$ is intant reward at learning step $t$. As we can see in equations (3), we can obtain the advantage function with just only value function. As a result, the neural network of the critic approximates the value function and estimate the advatage function in every updating step of policy network.

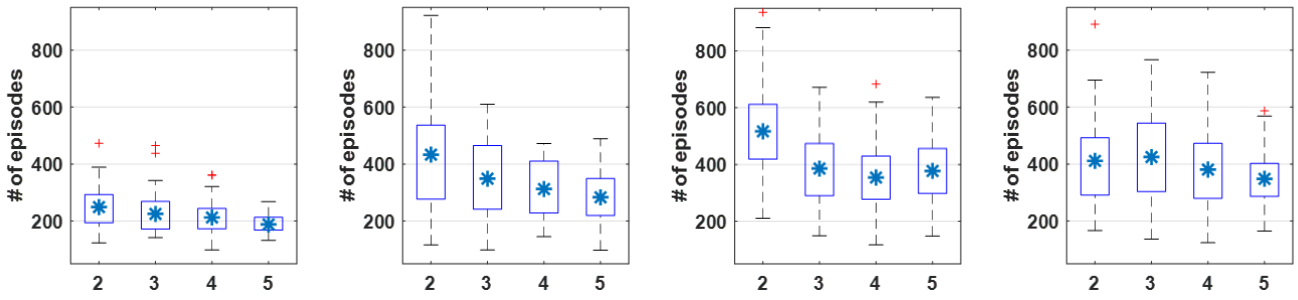
Under A2C algorithm, we have to select which model to learn using FRD framework - only one among two models or both? As we mensioned in section 2 and 3, the policy network forms the experience memory with the policy $\pi$. Similarly, the value network forms the *value memory* that consists of the state and corresponding value pairs. In the FRD case, average policy is replaced by the *average value*.

In Figure 2, we represent the performance comparison in each case: both, policy network, and value network. Three cases have similar performance in terms of the number of episodes until complete the mission. Unlike two other cases, the case exchanging the policy network shows stable learning results, i.e., the variation of mission completion time is smaller than two other cases. For this reason, we select the policy network to applying FRD framework. In the rest of the paper, we utilize FRD framework with the experience memory made by the output of policy network unless we mention.

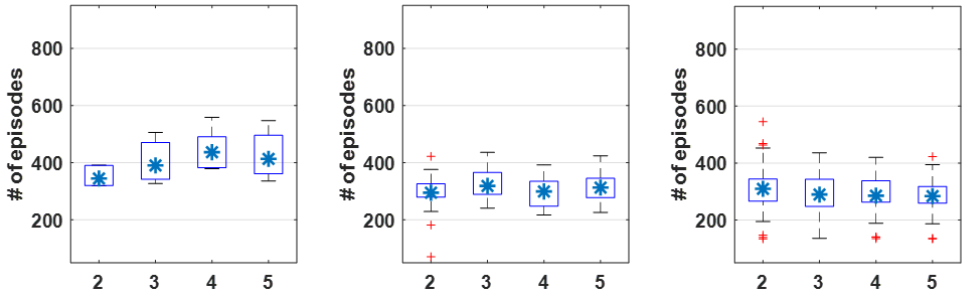
## 5 Experiments

The group of the RL agents share the output of policy network to construct the proxy experience memory $\mathcal{M}^P$ utilizing federated reinforcement distillation under advatage actor-critic algorithm. In this paper, we implement the proposed federated reinforcement distillation framework in the *Cartpole-v1* in *OpenAI gym* environment to evaluating the performance. We evaluate the performance of propose FRD framework in terms of the number of episodes until the group of agent complete the mission. The mission of the group of agents is defined as achieve the average standing duration of the pole over 10 episodes exceed the predetermined time duration. We assume that the group of agents complete the mission if just one of the agents in group completes the mission. Each agent adopts the advantage actor-critic model for local model and the model size of policy network presented in Table 1. Note that the model size of the value network is identical with that of policy network.

Before implement the federate reinforcement distillation, pre-arranging of the state clustering is needed. The state of *Cartpole* environment consists of four component, which is position of cart, velocity of cart, angle of pole, velocity of pole tip. We evenly divide the each component with the number as $S$ subsections. Then, we form




(a) Setting 1. (b) Setting 2. (c) Setting 3. (d) Setting 4.




(e) Setting 5. (f) Setting 6. (g) Setting 7.

Figure 3: Simulation results in *Cartpole* environment. The x-axis label of all graphs is the number of agents in cooperative group and the y-axis label of all graphs is the number of episodes until the agent group completes the mission. The mission of the group of agents is achieve the average standing duration of the pole over 10 episodes exceed the predetermined time duration. We assume that the group of agents complete the mission if one of the agents in group completes the mission. The agents make the proxy experience memory with the output of policy network only.

Table 1: Hyper parameters of federated reinforcement distillation.

| Setting | # of proxy states ($S^4$) | Memory exchange period ($E$) | Initial learning time ($I$) | # of weights per hidden layer ($n$) | # of hidden layers |
|---|---|---|---|---|---|
| *1* | $100^4$ | 25 | 50 | **24** | 2 |
| *2* | $\mathbf{100^4}$ | 25 | **50** | **100** | 2 |
| *3* | $100^4$ | 25 | **100** | 100 | 2 |
| *4* | $\mathbf{50^4}$ | 25 | 50 | 100 | 2 |
| *5* | $100^4$ | **10** | **0** | 24 | 1 |
| *6* | $100^4$ | **50** | **0** | 24 | 1 |
| *7* | $100^4$ | 25 | **125** | 24 | 1 |

the state cluster as the combination of divided components. As a result, the number of state cluster $|\mathcal{C}|$ is identical with $S^4$. The proxy state of each state cluster is defined as the middle value of each subsection of components. For example, the proxy state of the corresponding state cluster $C_j$ is $s_p = [0.5, -0.75, 0.75, 0.05]$ when $C_j = \{[0, 1), [-1, -0.5), [0.5, 1), [0, 0.1)\}$.

We perform the simulations with various hyper parameter settings presented in Table 1 and corresponding results are presented in Figure 3. We investigate the impact of each hyper parameter in terms of the number of episodes until the agent group complete the mission. The box in Figure 3 represents the data from 25% to 75%. The blue star represents the average of data. The red line represents the median of data.

**Impact of the Proxy State Size**. In the *Setting 2* and *Setting 4*, we can observe the impact of the proxy state size on the performance of FRD. When the multiple agents cooperate, the performance of *Setting 4* is better than that of *Setting 2* in terms of average number and the variance of episodes. As the number of agents is increase, the relation is reversed. Because the policy resolution of proxy state with smaller size is low, the knowledge of agents is blurred compare to that of proxy state with bigger size. Nevertheless, multiple agents case of *Setting 4* has better performance though the proxy state size is 16 times smaller than that of *Setting 2*. It means that the group of agent choose the proxy state size to reducing the payload size of exchanging information. If the agents cooperate through wireless channel, they can select the proper state cluster size sacrificing a bit of learning performance.

**Impact of Memory Exchange Preiod**. In the *Case 5*, the performance of group agent is getting worse as the number of agents is increase. Too frequent memory exchange and local model update has no merit on increasing the number of agents. As shwon in the *Setting 6*, moderate frequency of memory exchange brings stable performance enhancement.

**Impact of Initial Learning Time**. If there is no initial learning time before exchanging the experience memory, the performance of FRD is degraded as well as unstable. In the *Setting 5*, absence of initial learning time results in performance degradation as the number of agents is increase.

The local model of agent is not trained enough to exchange there proxy experience memory. Furthermore, too long initial learning time is also negative to the performance of FRD. Because too long initial learning time may give a chance of learning bad policy of indivisual local model of agent, the cooperation of agents is getting worse the training of local model of each agent. Comparing the *Setting 2* and *Setting 3*, the performance of the *Setting 2* is better than that of the *Setting 3*. As a result, the initial learning time should be selected properly to achieve higher performance.

**Impact of Neural Network Model Size**. As we can see in *Setting 1* and *Setting 2*, smaller NN has better performance in terms of the number of episodes until the group agent complete the mission. Because we measure how fast the group agent complete the mission, bigger NN has disadvantage in terms of convergence duration. In future work, the advantage of big NN compare to small NN can be evaluated in the more complicate and score-pursuing environment like *Atari games* in *OpenAI gym*. On the other hand, too small NN has marginal gain about FRD. In *Setting 6* and *Setting 7*, the performance enhancement along increment of the number of agents is limited in certain average value boundary.

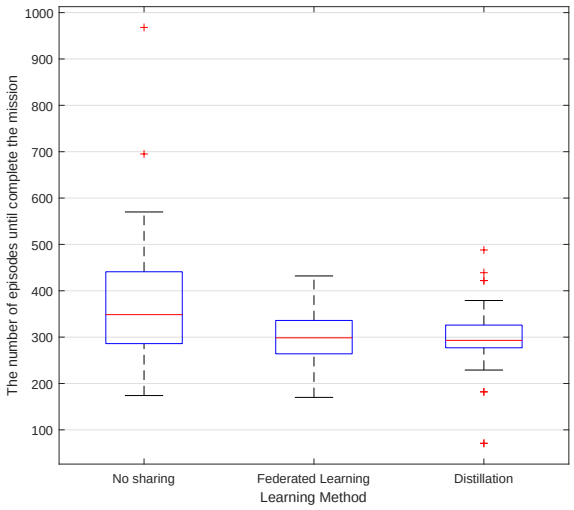



Figure 4: Performance comparison between the federated learning and federated reinforcement distillation when the multiple agents collaboate.

## 6 Discussion and Concluding Remarks

In this paper we introduce privacy-preserving distributed reinforcement learning framework, termed *federated reinforcement distillation (FRD)*. The key idea is to exchange a *proxy experience memory* comprising a pre-arranged set of states and time-averaged policies. It makes possible to conceal the actual experience and additionally has benefit of reduced memory size. When the distributed learning is conducted in communication-constrained situation, e.g., through wireless channel, proposed FRD framework has advantage to existing policy distillation.

Based on advantage actor-critic (A2C) algorithm, we evaluate the performance of FRD in various proxy memory structure and different memory exchanging rules. First, we investigate the impact of proxy memory structure which network is used for FRD in A2C algorithm - policy network, value network or both. Second, based on the first investigation, we implement policy network based FRD and evaluate the performance in various setting of memory exchanging rules - when, how often, how large.

As the future work, performance comparison between the federated learning and FRD is promising. We evaluate the performance in simple setting when the multiple agents collaborate. The performance in terms of average number of episodes until the group of agent complete the mission is fairly equivalent. But in terms of variation, FRD has better performance than federated learning. The performance difference is due to the amount of noise when knowlede transfer occur. It means that the noise of FRD is less than that of federated learning.

## References


[Jeong *et al.*, 2018] Eunjeong Jeong, Seungeun Oh, Hyesung Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-IID private data. In *Proc. NeurIPS Workshop on Machine Learning on the Phone and other Consumer Devices (MLPCD)*. Montréal, Canada, December 2018.

[McMahan *et al.*, 2017] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proc. of AISTATS*, Fort Lauderdale, FL, USA, April 2017.

[Mnih *et al.*, 2016] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1928–1937, New York, USA, 20–22 Jun 2016. PMLR.

[Park *et al.*, 2018] Jihong Park, Sumudu Samarakoon, Mehdi Bennis, and Mérouane Debbah. Wireless network intelligence at the edge. *submitted to Proc. IEEE* [Online]. ArXiv preprint: https://arxiv.org/abs/1812.02858, 2018.

[Rusu *et al.*, 2016] A. Rusu, S. Colmenarejo, C. Gulcehre, G. Desjardins, J. Kirkpatrick, and R. Pascanu. Policy distillation. *ICRL*, 2016.

[Shiri *et al.*, 2019] Hamid Shiri, Jihong Park, and Mehdi Bennis. Massive autonomous UAV path planning: A neural network based mean-field game theoretic approach. *submitted to GLOBECOM 2019* [Online]. *ArXiv preprint: https://arxiv.org/abs/1905.04152*, 2019.

[Wang *et al.*, 2016] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Freitas. Dueling network architectures for deep reinforcement

learning. *Proceedings of the 33rd International Conference on International Conference on Machine Learning*, 48:1995–2003, 2016.

[Zhuo *et al.*, 2019] Hankz Hankui Zhuo, Wenfeng Feng, Qian Xu, Qiang Yang, and Yufeng Lin. Federated reinforcement learning. [Online]. *Arxiv preprint: https://arxiv.org/abs/1901.08277*, 2019.