CAESAR: Enhancing Federated RL in Heterogeneous MDPs through Convergence-Aware Sampling with Screening

Hei Yi Mak
ETH Zurich
Zurich, Switzerland
heimak@ethz.ch

Flint Xiaofeng Fan
National University of Singapore
Singapore
fxf@u.nus.edu

Luca A. Lanzendörfer
ETH Zurich
Zurich, Switzerland
lanzendoerfer@ethz.ch

Cheston Tan
A*STAR
Singapore
cheston-tan@i2r.a-star.edu.sg

Wei Tsang Ooi
National University of Singapore
Singapore
ooiwt@comp.nus.edu.sg

Roger Wattenhofer
ETH Zurich
Zurich, Switzerland
wattenhofer@ethz.ch

ABSTRACT

In this study, we delve into Federated Reinforcement Learning (FedRL) in the context of value-based agents operating across diverse Markov Decision Processes (MDPs). Existing FedRL methods typically aggregate agents' learning by averaging the value functions across them to improve their performance. However, this aggregation strategy is suboptimal in heterogeneous environments where agents converge to diverse optimal value functions. To address this problem, we introduce the **C**onvergence-**A**war**E** **SA**mpling with sc**R**eening (`CAESAR`) aggregation scheme designed to enhance the learning of individual agents across varied MDPs. `CAESAR` is an aggregation strategy used by the server that combines convergence-aware sampling with a screening mechanism. By exploiting the fact that agents learning in identical MDPs are converging to the same optimal value function, `CAESAR` enables the selective assimilation of knowledge from more proficient counterparts, thereby significantly enhancing the overall learning efficiency. We empirically validate our hypothesis and demonstrate the effectiveness of `CAESAR` in enhancing the learning efficiency of agents, using both a custom-built `GridWorld` environment and the classical `FrozenLake-v1` task, each presenting varying levels of environmental heterogeneity.

KEYWORDS

Reinforcement learning, federated reinforcement learning, heterogeneous environments.

1 INTRODUCTION

Federated Reinforcement Learning (FedRL) [14, 20] is a burgeoning field in Reinforcement Learning. Distinct for its collaborative learning approach, FedRL enables distributed agents to learn collectively while maintaining the privacy of their local data — the raw trajectories sampled from the local environments. FedRL leverages techniques in Federated Learning (FL), notably Federated Averaging [12], to aggregate agent parameters to improve learning efficiency. While existing research on FedRL [2–4, 9, 18, 22, 23] predominantly assumes *homogeneous* environments, where all local environments correspond to the same Markov Decision Process (MDP) [16] with identical dynamics and rewards, real-world applications often defy this assumption. For instance, in the healthcare domain, FedRL holds promise for optimizing predictive models across various hospitals, each characterized by distinct patient demographics and disease patterns [19]. This diversity among patient populations and clinical manifestations leads to inherent *heterogeneity* within the data environments shaped by the MDPs.

This challenge is underscored in the research by Hao et al. [6]. While their work investigates FedRL in the context of heterogeneous environments, it primarily focuses on training a unified model to perform consistently across disparate local environments. This approach, akin to implementing a standard healthcare protocol across hospitals serving diverse patient populations, may prove to be impractical. Such a one-size-fits-all approach fails to accommodate the unique healthcare needs and specific disease prevalence of different communities, potentially resulting in suboptimal or even detrimental outcomes. This underscores the critical need for tailored approaches that respect and respond to the unique characteristics of each environment.

In contrast, our research is centered on scenarios where each agent learns a localized policy for its designated MDP. This is analogous to designing customized healthcare strategies for each hospital, taking into account the unique health demographics and local environmental influences of their patient population. We explore the potential of these agents to collaboratively enhance the learning of localized policies, each specifically tailored to the corresponding environment. A pivotal assumption in our work is the unknown nature of both the number of distinct MDPs and the specific assignments of agents to these MDPs.

In response to these challenges, we propose a convergence-aware adaptive sampling strategy for value-based agents in FedRL settings characterized by heterogeneous environments. This strategy is based on the insight that value functions of agents optimizing for the same MDP are expected to converge towards a singular optimal value over time, thereby naturally reducing the variance in learning trajectories among these agents, or "*peers*." Preliminary experiments suggest that while this strategy is effective in filtering out "*non-peers*"—agents whose environmental contexts or MDPs diverge significantly from one another, leading to disparate optimal policies and value functions—it might inadvertently prioritize the inclusion of suboptimal peers. These are agents within the same MDP whose strategies or learning progress are not as advanced,

Proc. of the Adaptive and Learning Agents Workshop (ALA 2024), Avalos, Müller, Wang, Yates (eds.), May 6-7, 2024, Online, https://ala2024.github.io/. 2024.

potentially anchoring the group to suboptimal points. To address this, we introduce an additional screening process, aimed at incorporating only those agents that exhibit better performance. This dual approach of adaptive sampling and selective screening effectively mitigates the risk of suboptimal peer selection, enhancing the learning efficacy of agents in their respective MDPs.

In this paper, we address the challenges of training individual policies with environmental heterogeneity in FedRL. We begin by formulating the problem setup of FedRL in heterogeneous environments (Sec. 3) and proceed to examine various aggregation schemes (Sec. 4). We then introduce the **C**onvergence-**A**war**E** **SA**mpling with sc**R**eening (`CAESAR`) aggregation scheme that tailors the average value functions for individuals to effectively improve their learning (Sec. 4.5). `CAESAR` stands out for its dual-layered approach: firstly, utilizing a convergence-aware sampling mechanism for efficient peer identification in diverse MDPs (Sec. 4.4); and secondly, incorporating a selective screening process (Sec. 4.6) to refine agent interactions, prioritizing only those agents that demonstrate superior performance. We empirically validate the effectiveness and robustness of `CAESAR` to improve agents' learning using environments of `GridWorld` and `FrozenLake-v1`, engineered for the purpose of illustrating environmental heterogeneity (Sec. 5). We have made our work publicly available and open-sourced,[1] providing new perspectives and viable approaches for tackling the challenges of FedRL in heterogeneous settings.

2 PRELIMINARIES

Markov Decision Processes (MDPs). In the realm of reinforcement learning, sequential decision-making problems are commonly modeled using MDPs [16]. An MDP is characterized by a 6-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \rho)$ where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ represents the action space, $\mathcal{P}(s'|s,a)$ defines the transition probabilities between states, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, γ is the discount factor, and ρ is the initial state distribution.

Q-learning. Q-learning [7] stands as a cornerstone in classical reinforcement learning, operating as an off-policy temporal difference algorithm. In Q-learning, an agent learns an action-value function $Q : \mathcal{S} \times \mathcal{A} \to \mathcal{R}$ using a table. The entry $Q(s,a)$, also known as the Q-values, estimates the expected return of taking action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$. The value of $Q(s,a)$ is updated by applying the Bellman equation:

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha(r + \gamma \max_{a'} Q(s',a')) \tag{1}$$

where s is the current state, a is the current action to be executed, r is the immediate reward, s' is the next state, and α is the learning rate. Then a decision policy π_Q can be obtained via exploiting the updated Q-values:

$$\pi_Q(s) \leftarrow a_t = \arg\max_a Q(s_t, a). \tag{2}$$

The *optimal action* at state s_t is defined as $a_t^* = \arg\max_a Q^*(s_t,a)$ where $Q^*(s_t,a)$ is the *optimal Q-function* which gives the expected return for starting in state s_t, taking action a, and following the policy thereafter.

[1] https://github.com/hughiemak/CAESAR

Federated Reinforcement Learning (FedRL). Initially introduced by Zhuo et al. [23], Federated Reinforcement Learning (FedRL) has gained increasing prominence, evidenced by its extensive application in various real-world scenarios [4, 5, 10, 11, 13, 17, 21, 22] and its substantial theoretical development [3, 6, 8, 9, 18]. Notably, Fan et al. [3] conducted pioneering work on the robust convergence of federated policy gradients, demonstrating sublinear speedup. Khodadadian et al. [9], Woo et al. [18] further advanced the field by showcasing linear speedup in federated Q-learning under Markovian Sampling. Shen et al. [15] established a linear speedup for federated Actor-Critic algorithms under i.i.d. sampling. A common assumption in these related works is the homogeneity of MDPs across all agents participating in FedRL. This perspective was expanded by Hao et al. [6], who explored FedRL in the context of environmental heterogeneity. Their research primarily aimed at developing a global shared policy model within an imaginary MDP framework.

3 FEDERATED REINFORCEMENT LEARNING WITH HETEROGENEOUS ENVIRONMENTS

MDP Configuration. In the FedRL setting under consideration, we have N agents, and a collection of MDPs, where the quantity of distinct MDPs (K) is less than or equal to N. Each MDP, denoted as M_k, shares a common state space $\mathcal{S}$ and action space $\mathcal{A}$, but is uniquely defined by its transition dynamics $\mathcal{P}_k(s'|s,a)$ and reward function $\mathcal{R}_k : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. Thus, the MDP M_k is represented by the 6-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}_k, \mathcal{R}_k, \gamma, \rho_k)$, where γ is the discount factor and ρ_k is the initial state distribution specific to MDP M_k. An assignment function $f : [N] \to [K]$ determines the allocation of each agent to these MDPs.

Heterogeneity in MDPs. The core of heterogeneity in this setting stems from the differences in transition dynamics and reward functions among the MDPs. An example of this heterogeneity is depicted in Fig. 1, where two MDPs share the same state and action spaces but have distinct reward functions. This diversity in dynamics and rewards exemplifies the complexity and variability agents encounter in heterogeneous environments.

Operational Assumptions. A pivotal assumption in our approach is the unknown nature of both K (the number of MDPs) and the assignment function f. This uncertainty adds a layer of complexity to the learning process, as agents must navigate and adapt

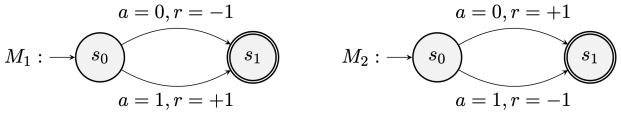


Figure 1: Two heterogeneous MDPs. MDP M_1 rewards -1 for action 0 and $+1$ for action 1, while MDP M_2 rewards $+1$ for action 0 and -1 for action 1. The optimal value functions are $Q_1(s_0,0) = -1, Q_1(s_0,1) = 1$ for M_1, and $Q_2(s_0,0) = 1, Q_2(s_0,1) = -1$ for M_2, respectively. Averaging these value functions results in $\bar{Q}(s_0,0) = \bar{Q}(s_0,1) = 0$, showing a misrepresentation of optimal values for both MDPs.

to their assigned MDPs without prior knowledge of the overall system configuration.

Agent Learning and Objectives. Each agent in our system, denoted as i, is a value-based learner, employing techniques such as Q-learning for policy optimization. Every Agent i interacts solely with a local instantiation of its assigned MDP, $M_{f(i)}$, from which it gathers and analyzes sample trajectories to inform its learning. The primary goal for each agent is to optimize its action-value function Q_i, aiming to achieve optimal expected performance within its unique local environment. This focus on individual optimization within a shared learning framework underscores the challenge of balancing local adaptation with collaborative learning in FedRL.

Federated Updates in FedRL. Following existing FedRL work [3, 4, 9, 23], a central server is available to coordinate the federated learning. We consider a FedRL training process where a federated update takes place every H local updates. At each local update step, each agent performs a standard learning step using the value-based RL algorithm. During the federated update, for each agent i, the server will select a subset of agents $S \subseteq [N]$ and aggregate their value functions into a new value function $\bar{Q}$. For tabular Q-learning, a straightforward aggregation method is averaging the value functions (Q-tables) across selected agents:

$$\bar{Q}(s,a) = \sum_{j \in S} Q_j(s,a), \quad \forall s \in \mathcal{S}, a \in \mathcal{A}. \tag{3}$$

After aggregation, the server sends $\bar{Q}$ to agent i and updates Q_i towards $\bar{Q}$:

$$Q_i(s,a) \leftarrow \beta Q_i(s,a) + (1-\beta)\bar{Q}(s,a), \quad \forall s \in \mathcal{S}, a \in \mathcal{A} \tag{4}$$

where $\beta \in [0,1]$ is a blending parameter controlling the extent of update from the federated value function.

A key challenge in the federated update process is determining the optimal subset S for each agent without prior knowledge of f, the agent's specific environment, or direct access to its local trajectories. The selection of S is pivotal in ensuring that the aggregated value function $\bar{Q}$ is conducive to agent i's learning in its MDP. In Sec. 4, we will explore various aggregation schemes for selecting S.

4 AGGREGATION SCHEMES

In this section, we explore various schemes for selecting the subset of agents, S, for each agent i, culminating in the introduction of our novel CAESAR scheme.

4.1 `Self`

The `Self` scheme serves as a baseline, where agents learn independently, without federated updates. Using this scheme, Eq. (3) can be viewed as:

$$\bar{Q} = Q_i, \ \ \forall i \in N$$

implying no external influence during the federated update phase. Consequently, the selected subset S only includes the agent itself:

$$S = \{i\}.$$

It is a fundamental expectation that, for agents to be incentivized to engage in the federative process, any employed selection scheme must ensure that the aggregated knowledge surpasses the performance achievable by the `Self`. This is essential to justify the collaborative effort in the federative learning context.

4.2 `All`

The scheme `All` is another baseline corresponding to the canonical FedRL averaging scheme where all agents are included for aggregation to compute Eq. (3):

$$\bar{Q}(s,a) = \sum_{j=1}^{N} Q_j(s,a).$$

In this scheme, the selected subset always includes all agents:

$$S = \{1, 2, \ldots, N\}.$$

In a FedRL setting characterized by heterogeneous local environments, the `All` scheme may impede the learning process and potentially obstruct convergence. This issue arises because each agent's value function, denoted as Q_j, is being optimized for different MDPs. In essence, they are converging towards disparate optimal value functions. Consequently, value functions optimized for one MDP might adversely affect the aggregated value function $\bar{Q}$, resulting in misleading guidance for agent i. To illustrate, consider the scenario with two simple MDPs, M_1 and M_2, as shown in Fig. 1. Suppose agents 1 and 2 are learning in these MDPs respectively and have both reached their optimal value functions: $Q_1(s_0, 0) = -1, Q_1(s_0, 1) = 1$ for MDP M_1, and $Q_2(s_0, 0) = 1, Q_2(s_0, 1) = -1$ for MDP M_2. However, the averaged Q-values, $\bar{Q}(s_0, 0)$ and $\bar{Q}(s_0, 1)$, both result in 0. These average values are suboptimal for both MDPs. Updating Q_1 and Q_2 based on $\bar{Q}$ would therefore misguide the agents and steer them away from their currently optimal values, highlighting the challenge of aggregation in heterogeneous environments.

4.3 `Peers`

The `Peers` scheme is an unrealistic scheme in our setting that serves as a hypothetical benchmark. This scheme operates under the assumption of having prior knowledge of MDP assignments, denoted as f, and including only those agents assigned to the same MDP as agent i. These agents are referred to as the 'peers' of agent i. The selected subset of agents is therefore

$$S = \{j \in [N] : f(i) = f(j)\}.$$

Such a presumption renders it impractical in scenarios where this information is not available, i.e., the server lacks insight into the peers of agent i. Despite this, the scheme serves as a valuable benchmark, illustrating the potential advantages of precise, environment-specific aggregation, such that:

$$\bar{Q}(s,a) = \sum_{j \in S} Q_j(s,a), \ \ S = \{j \in [N] : f(i) = f(j)\}.$$

Contrasting with the `All` scheme (Sec. 4.2), this conceptual approach offers greater efficiency by exclusively incorporating value functions that are optimized for the same MDP. This selective aggregation ensures that value functions from disparate MDPs, which could potentially mislead the learning process, are not included. Furthermore, this scheme provides a distinct advantage over the `Self` scheme, where agents learn in isolation. By leveraging the

collective knowledge of agents assigned to the same MDP, it enables a more targeted and effective aggregation of value functions, enhancing the overall learning effectiveness.

4.4 Sampling

Inspired by the advantageous attributes of the hypothetical `Peers` scheme, we explore the feasibility of devising a similar selection scheme. Our goal is to accurately identify the peers of Agent i without relying on the prior assumption of peer knowledge inherent to the `Peers` approach. This task is especially challenging in our scenario due to the lack of prior knowledge about each agent's assigned MDP and the absence of direct access to local trajectories at the server.

To circumvent this, we propose utilizing the convergence of agent value functions as a heuristic for peer detection. The main idea is that if the value functions Q_i and Q_j are both being optimized for the same MDP, they should converge towards a unique optimal value function Q^* over time. As a result, the values $Q_i(s,a)$ and $Q_j(s,a)$ for all state-action pairs (s,a) will progressively become more similar. We empirically validate this convergence behavior in a gridworld setting, as detailed in Fig. 4 (Sec. 5.2).

Given this intuition, the convergence of value functions emerges as a practical heuristic for estimating whether two agents are learning in the same MDP. This insight leads to our convergence-aware sampling scheme, `Sampling`, wherein the subset S is sampled based on probabilities $p_{i1}, \ldots, p_{iN}$. Each probability p_{ij} quantifies the likelihood of including agent j in S and is dynamically adjusted based on the observed convergence between Q_i and Q_j. At the onset of training, the server initializes an $N \times N$ matrix p, where the entry p_{ij} is set as:

$$p_{ij} = \begin{cases} p_0 & i \neq j, \\ 1 & i = j. \end{cases}, \quad \forall i, j \in [N] \tag{5}$$

where p_0 functions as an initial assumption or 'prior' about the task, reflecting the preliminary likelihood of agents being peers before any learning occurs. By default, it can be assigned a value of 0 to encourage self-learning at the start of training when the convergence information is insufficient.

Prior to each federated update, the server updates the entries p_{ij} of the probability matrix p. This update is contingent upon evaluating the evolving similarity between the value functions Q_i and Q_j. Specifically, the server assesses how the similarity of these value functions has changed relative to their states observed H steps ago. This dissimilarity between two value functions Q and Q' is defined as the mean absolute difference across all state-action pairs:

$$d(Q, Q') = \frac{1}{|\mathcal{S}| \times |\mathcal{A}|} \sum_{s,a} |Q(s,a) - Q'(s,a)|.$$

Let $Q_k^{(t)}$ be agent k's current value function and $Q_k^{(t-H)}$ be agent k's value function H steps ago. For each pair of agents $\{i, j\}$, we update

$$p_{ij} \leftarrow \begin{cases} \min(p_{ij} + \delta, 1) & \text{if } d(Q_i^{(t-H)}, Q_j^{(t-H)}) - d(Q_i^{(t)}, Q_j^{(t)}) > \xi, \\ \max(p_{ij} - \delta, 0) & \text{otherwise.} \end{cases}$$

where $\delta > 0, \xi \geq 0$. This update rule is designed such that if the value functions Q_i and Q_j demonstrate a sufficient decrease in dissimilarity over a specific time window H, the server will increase the probability value p_{ij}. This increment in p_{ij} effectively raises the likelihood of agent j being selected for agent i's subset for aggregation. The time window H acts as a temporal frame of reference, enabling the server to assess changes in similarity over a defined period. Conversely, if Q_i and Q_j do not exhibit the required degree of convergence over the time window, p_{ij} is reduced. Persistent convergence trends lead to a gradual increment in p_{ij}, favoring the selection of agents with converging value functions. Consequently, the `Sampling` scheme dynamically adapts its selection criteria over time, increasingly favoring the inclusion of agents with value functions that demonstrate a tendency to converge. Parameters δ and ξ control the sensitivity of p_{ij} adjustments and the required degree of convergence, respectively.

Algorithm 1: `CAESAR`

`ServerExecutes`$(H, \sigma, p_0, \delta, \xi, \beta)$**:**
- initialize value functions Q_i for each agent $i \in [N]$
- initialize $Q_i^{old} \leftarrow Q_i$ for each agent $i \in [N]$
- initialize matrix p: $p_{ij} = \begin{cases} p_0 & i \neq j, \\ 1 & i = j. \end{cases}, \forall i, j \in [N]$
- **for** each step $t = 1, \ldots, T$ **do**
  - `LocalUpdate`(i) for each agent $i \in [N]$
  - **if** t mod $H = 0$ **then**
    - $g_k \leftarrow$ `EvalLocalPerformance`$(k), \forall k \in [N]$
    - `UpdatePMatrix`$(p, \delta, \xi, \{Q_k^{old}\}_k, \{Q_k\}_k)$
    - `FederatedUpdate`$(i, \beta, p, \{Q_k\}_k, \{g_k\}_k)$ for each agent $i \in [N]$
    - $Q_i^{old} \leftarrow Q_i$ for each agent $i \in [N]$

`UpdatePMatrix`$(p, \delta, \xi, \{Q_k^{old}\}_k, \{Q_k\}_k)$**:**
- **for** agent $i = 1$ to N **do**
  - **for** agent $j = i + 1$ to N **do**
    - $p_{ij} \leftarrow \begin{cases} \min(p_{ij} + \delta, 1) & \text{if } d(Q_i^{old}, Q_j^{old}) - d(Q_i, Q_j) > \xi, \\ \max(p_{ij} - \delta, 0) & \text{otherwise.} \end{cases}$

`FederatedUpdate`$(i, \beta, p, \{Q_k\}_k, \{g_k\}_k)$**:**
- initialize $S' = \{\}$
- **for** $j \in [N]$ **do**
  - add j to S' with probability p_{ij}
- $S = \{j : j \in S' \text{ and } g_j > g_i\}$
- Construct $\bar{Q}$: $\bar{Q}(s,a) = \sum_{j \in S} Q_j(s,a), \quad \forall s \in \mathcal{S}, a \in \mathcal{A}$
- Update Q_i:
  - $Q_i(s,a) \leftarrow \beta Q_i(s,a) + (1-\beta)\bar{Q}(s,a), \quad \forall s \in \mathcal{S}, a \in \mathcal{A}$

4.5 CAESAR

As will be discussed in Sec. 5, the Sampling scheme excels at filtering out non-peers from the set S, but it also has a potential downside: the server might inadvertently incorporate only peers that are underperforming, confining slow-progressing agents in suboptimal points in the value function space. To mitigate this issue, we introduce an additional screening process to refine agent interactions, prioritizing only those agents that demonstrate superior performance, culminating in the CAESAR aggregation scheme.

In the CAESAR scheme, we initially select a subset of agents based on the probabilities $p_{i1}, \ldots, p_{iN}$, following the same process as in Sampling. The primary objective of this sampling step, akin to that in Sampling, is to identify probable peers by assessing the convergence trends of their value functions. Subsequently, we introduce a screening layer, which focuses on the comparative performance of these selected agents. The rationale behind this additional step is to circumvent the pitfall of updating the value function Q_i towards the average of lower-performing peers, which could hinder the convergence of Q_i to its optimal state.

To implement this, the local performance of each agent k, $g_k \approx \mathbb{E}_{M_k, \pi_{Q_k}}\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}_k(s_t, a_t)\right]$, is measured prior to the federated update. During the update, for a given agent i, the server initially samples a preliminary subset of agents, S', in line with the probabilities $p_{i1}, \ldots, p_{iN}$. It then further refines S' by retaining only those agents whose performance, g_j, exceeds that of agent i ($g_j > g_i$). The final subset for aggregation is thus defined as:

$$S = \{j : j \in S' \text{ and } g_j > g_i\}$$

This resulting subset S is then utilized to assemble the aggregated value function $\bar{Q}$ for updating Q_i of each agent. This completes the outlines for the CAESAR scheme. For a detailed procedural breakdown, refer to the pseudocode presented in Algorithm 1.

4.6 Screen

As a complementary approach, the Screen scheme focuses solely on the screening process based on local performance, without considering convergence trends:

$$S = \{j : g_j > g_i\}$$

Screen selects agents that are performing better than the target agent, but may include those from different MDPs. This scheme tests the efficacy of performance-based selection in isolation.

Each scheme presents a unique approach to aggregating value functions within a FedRL framework. Our goal, as detailed in Sec. 5, is to assess the effectiveness of these schemes in enhancing individual agent performances, particularly in heterogeneous environments. This analytical endeavor aims to uncover the most effective strategies for knowledge aggregation in practical FedRL settings, thereby providing valuable insights into optimizing agent performance in diverse and complex scenarios.

5 EMPIRICAL EVALUATION

5.1 Experimental Settings

In this study, we conduct a comparative analysis of the six aggregation schemes discussed in Sec. 4. For this comparison, we employ Q-learning agents within two distinct environments: a custom-built environment GridWorld and the well-known FrozenLake-v1 task from the OpenAI Gym toolkit [1].

The GridWorld is designed as a 1-dimensional discrete environment, characterized by a state space $\mathcal{S} = \{-5, -4, \ldots, 4, 5\}$ and a binary action space $\mathcal{A} = \{0, 1\}$. The initial state for each episode is set at 0, with terminal states being 5 and -5. The agent's actions impact the state transitions: action 0 moves the state from x to $x - 1$, while action 1 advances the state from x to $x + 1$. Two distinct versions of this environment are considered, corresponding to two different MDPs. Fig. 2 provides a visual representation of these GridWorld environments. In the first MDP (MDP 1), a transition from state 4 to 5 yields a reward of +1, and a transition from -4 to -5 results in a reward of -1. All other state transitions provide a neutral reward of +0. The second MDP (MDP 2) inverts the reward structure of MDP 1, such that $r_2(s, a) = -r_1(s, a)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$). The FrozenLake-v1 environment presents

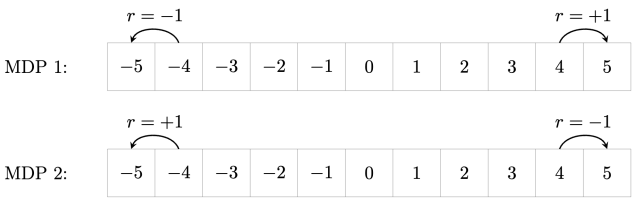


Figure 2: Two GridWorld MDPs. Their initial states are 0. In MDP 1 (top), transiting from state 4 to 5 generates a reward of +1 and transiting from state −4 to −5 yields a reward of −1. In MDP 2 (bottom), the signs of the rewards are flipped.

a 2-dimensional discrete challenge that effectively encapsulates the complexities of environmental heterogeneity. In this environment, agents are tasked with navigating to a designated goal while avoiding hazardous holes. The environment is characterized by a four-directional action space, and episodes end with a reward of +1 upon reaching the goal, or +0 if the agent falls into a hole or exhausts the allowed steps. The heterogeneity of the FrozenLake-v1 environment is induced by the distinct map configurations, as shown in Fig. 3. Each map represents a unique instantiation of a local MDP within the environment, characterized by its own specific arrangement of holes and paths, necessitating different strategic approaches for successful navigation. This diversity in maps provides a practical scenario to assess how FedRL algorithms perform across dynamically varied MDPs.

5.2 Hypothesis Verification Using GridWorld

For GridWorld, we partition $N = 20$ agents into $K = 2$ groups, each comprising 10 agents. These groups are then assigned to two different MDPs, M_1 and M_2, as depicted in Fig. 2. Each agent is trained for $T = 10000$ steps with an exploration rate $\epsilon = 0.1$, and receives a federated update every $H = 100$ steps.

Convergence Among Peers. In the relatively simple GridWorld environment, we capture the agents' Q-tables every H steps. Fig. 4 shows how Q-values $Q_i(s, a)$ for various state-action (s, a) pairs evolve for all agents $i \in [N]$ under Self (independent learning). The optimal values of these state-action pairs are different for M_1

(a) Map 0 (b) Map 1 (c) Map 2

Figure 3: `FrozenLake-v1` environments generated by three different maps. The agent's task is to navigate to the goal (the gift box) without falling into the holes.

and M_2. Notably, Q-values among peers converge towards the optimal values for their respective MDPs over time, supporting the use of Q-value convergence as a heuristic for detecting probable peers, as elaborated in Sec. 4.4.

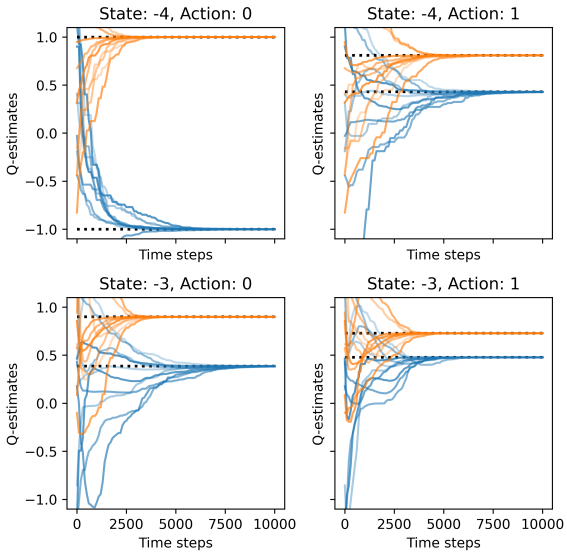


Figure 4: Convergence of Q-values among peers in `GridWorld` under `Self`. Q-values of M_1 agents (blue) and M_2 agents (orange) converge to their respective optimal values (black dotted lines) for state-actions $(s = -4, a = \cdot)$ and $(s = -3, a = \cdot)$ in `GridWorld`. ϵ is set to 0.9 to speed up convergence.

Comparative Performance Analysis. Fig. 5 illustrates the average performance (over 30 random seeds) of all agents under different aggregation schemes in `GridWorld`. We can observe that `All` is outperformed by `Peers`, affirming our hypothesis that including all agents in S to compute $\bar{Q}$ according to Eq. (3) is less effective in heterogeneous environments. The slower learning progress observed under `Screen` is attributed to its selection based solely on local performance, often including high-performing agents from different MDPs. Significantly, `CAESAR` shows comparable results to the hypothetical approach `Peers`, which operates under the assumption of perfect knowledge about agent-MDP assignments. Remarkably, `CAESAR` surpasses both `Sampling` and `Screen`, highlighting the synergistic effect of their combination on learning enhancement.

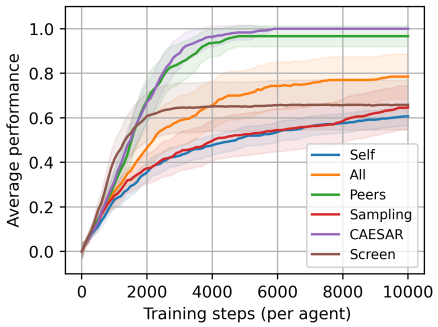


Figure 5: Average performance of the $N = 20$ agents in `GridWorld` under different averaging schemes with exploration rate $\epsilon = 0.1$. The plot averages independent runs over 30 random seeds where the shadows represent the 95% confidence intervals.

Analysis of Q-Value Evolution. To understand the critical role of the screening process in the `CAESAR` scheme, we track the progression of Q-values throughout the training period. Fig. 7 and Fig. 8 present the evolution of Q-values for agents assigned to M_1 under the `Sampling` and `CAESAR` schemes, respectively. These plots are generated from training sessions with the same random seed and initial agent configurations. Under `Sampling`, we notice that only two agents are able to approximate the optimal Q-values (indicated by black dotted lines), while the remaining agents stagnate at suboptimal points. In contrast, when employing `CAESAR`, a uniform and rapid convergence to optimal values is observed for all agents.

To gain a deeper understanding of the dynamics at play within the `Sampling` scheme, we analyze the changes in the p-matrix (Sec. 4.4) over various training stages, as illustrated in Fig. 6. This analysis reveals that `Sampling` is highly effective in filtering out non-peers, consistently selecting them with near-zero probability from timestep $t = 4000$ onwards. However, an intriguing behavior is observed: `Sampling` tends to overlook agents who are advancing quickly in their learning curve, opting instead for peers with slower progress rates. Specifically, at $t = 4000$ (as shown in the third plot of Fig. 6), `Sampling` assigns negligible probabilities to aggregate values of the fast learners, Agents 5 and 6, into the learning process of the slower-progressing peers, namely Agents 0, 1, 2, 3, 4, 7, 8, 9, as shown in Fig. 7. Despite the evident progress of the fast learners, `Sampling` scheme leads to a tendency for slower peers to primarily learn from each other, gravitating towards a consensus that strays from the optimal value. Such a strategy, while fostering a form of convergence, risks cementing the learning of slower-progressing agents around suboptimal values.

In contrast, `CAESAR` circumvents this issue through its dual-layered approach: it employs `Sampling` to effectively identify and exclude non-peers, and `Screen`, the screening process that prioritizes the inclusion of fast-progressing agents based on their local performance metrics, as evident in Fig. 8. This strategic selection tends to aggregate knowledge from faster-progressing peers, whose value functions Q_i are more optimal for the same MDP. Hence, `CAESAR` not only avoids the pitfalls of `Sampling` but also facilitates

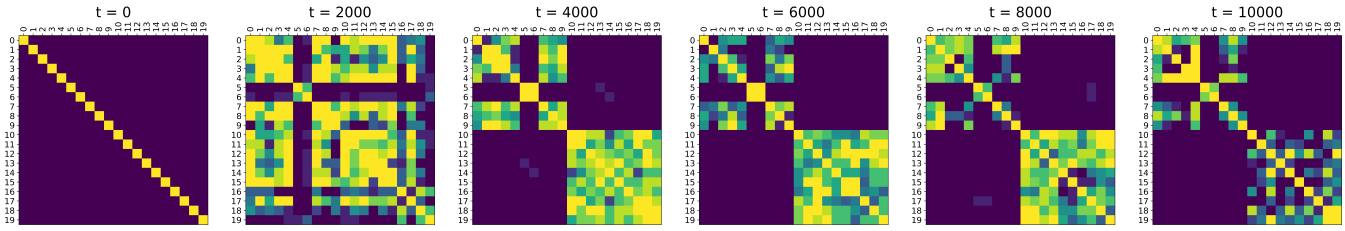


Figure 6: The matrix p as a heatmap (yellow and purple indicate 1 and 0 respectively) at 5 different time points under `Sampling`. The numbers on the axes correspond the agents, where agents 0 to 9 are assigned to M_1 and agents 10 to 19 are assigned to M_2. The color of the cell (i, j) indicates the probability of selecting agent j for agent i.

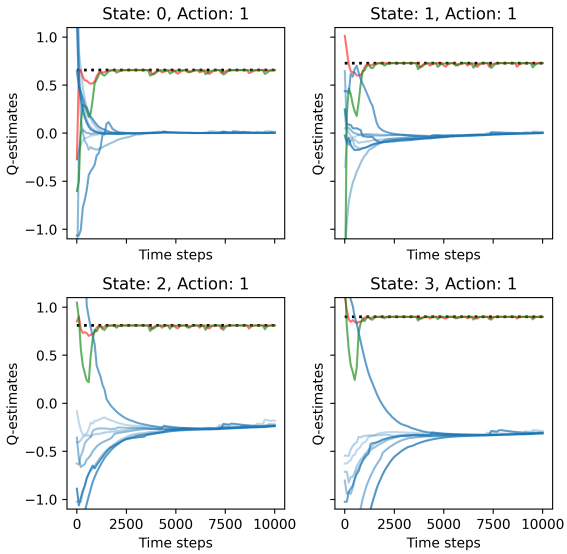


Figure 7: Q-values of the M_1 agents under `Sampling`. Two M_1 agents, Agent 5 (red curve) and Agent 6 (green curve), exhibit fast learning progress and converge to the true optimal values (black dotted lines) but the remaining M_1 agents (blue curves), Agents 0, 1, 2, 3, 4, 7, 8, 9, converge to non-optimal values.

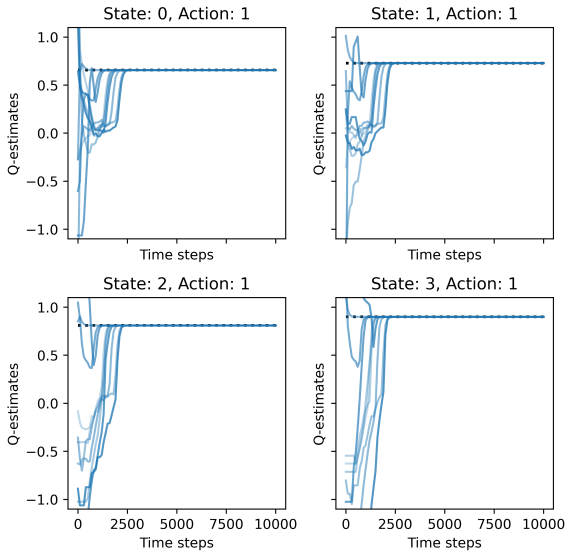


Figure 8: Under `CAESAR`, Q-values of all M_1 agents converge quickly to the true optimal values (black dotted lines).

a more effective knowledge transfer, significantly enhancing the learning efficiency across agents.

5.3 Effectiveness evaluation using `FrozenLake-v1`

In our study using `FrozenLake-v1`, we maintain the same experimental settings as in `GridWorld`, with $N = 20$ agents divided into two groups $K = 2$, each group comprising 10 agents assigned to two distinct MDPs M_1 and M_2, respectively. We assess the performance of the aggregation schemes under the following scenarios, each offering a different level of environmental heterogeneity:

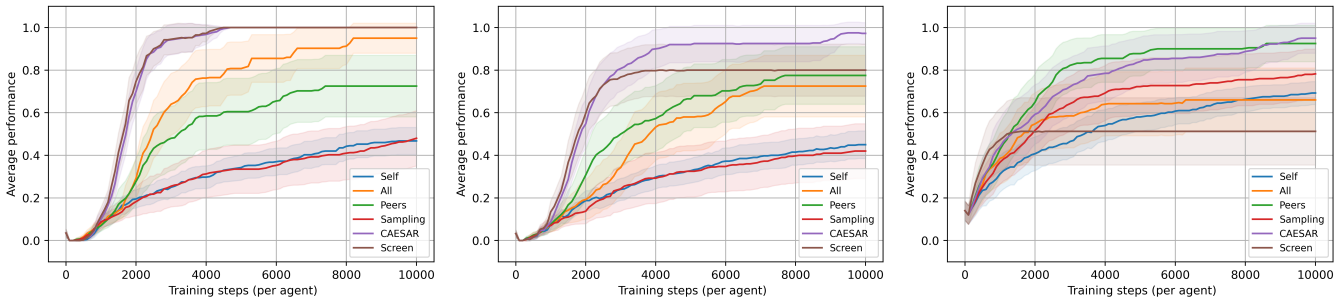
(1) **Homogeneous environments:** M_1 and M_2 are identical, both generated using the same random map with 4 holes. An example of such a map is shown in Fig. 3a (a).

(2) **Randomly heterogeneous environments** M_1 and M_2 are distinct, created using two random maps with differing positions of the 4 holes.

(3) **Strongly heterogeneous environments** Maps 1 and 2, as depicted in Fig. 3a (b) and (c) respectively, exhibit a significant disparity in difficulty levels, with Map 1 being the easier and Map 2 the more challenging. The two maps are designed to have substantial differences in their optimal Q-functions.

Fig. 9 shows the average performance of all agents across the different aggregation schemes in these scenarios. The results reveal that `CAESAR` consistently demonstrates robust performance in all three scenarios, contrasting with other schemes that struggle in at least one scenario.

In Scenario 1 (Fig. 9a), with identical MDPs $M_1 = M_2$, `All` demonstrates superior learning outcomes compared to `Peers`. This aligns with expectations, as all agents are engaged in the same task, making the inclusion of the entire agent pool in S more effective for leveraging collective insights. In this context, `All` benefits from a



(a) Homogeneous environments **(b) Random heterogeneous environments** **(c) Strongly heterogeneous environments**

Figure 9: Average performance of the $N = 20$ agents in `FrozenLake-v1` under different averaging schemes with exploration $\epsilon = 0.1$ in all three settings. The plots average independent runs over 30 random seeds where the shadows represent the 95% confidence intervals.

broader knowledge base than `Peers`, which limits its focus to peers, hence reducing the number of participating agents.

Conversely, in Scenario 2 (Fig. 9b), where M_1 and M_2 differ, `Peers` slightly outperforms `All`. This indicates that peer-based learning is more advantageous when agents are dealing with different MDPs, as it enables more targeted knowledge sharing.

The contrast becomes more pronounced in Scenario 3 (Fig. 9c), where `Peers` significantly surpasses `All`, with the latter even falling behind `Self` (independent learning). This scenario underscores the importance of excluding non-peers from S in heterogeneous environments. The discrepancy arises from the varying difficulty levels of M_1 (Map 1) and M_2(Map 2). Agents in the simpler M_1 quickly master the task, leading to high-value estimates of $Q(s_0, a =\rightarrow)$ which is not optimal for M_2 where the action $a =\rightarrow$ often leads to holes. Therefore, including M_1 agents' value functions in S can detrimentally affect the learning progress of M_2 agents, as their optimal Q-value functions diverge significantly.

`Screen` displays a notably inconsistent performance pattern across different levels of heterogeneity, excelling in Scenarios 1 and 2 but faltering in Scenario 3, where its results are even inferior to those of the independent learning approach, `Self`. This phenomenon stems from that the `Screen` scheme is effectively the `All` scheme with an additional screening process. In homogeneous environments (Scenario 1), where $M_1 = M_2$, this screening process effectively boosts performance by prioritizing agents with superior performance. However, in the more complex Scenario 3, `Screen` tends to erroneously include high-performing agents from the simpler M_1, whose optimal values are counterproductive in M_2. As a result, `Screen` inadvertently hinders the learning process for M_2 agents by propagating suboptimal Q-values. This issue is clearly demonstrated in Fig. 9c, where `Screen` achieves an average performance of only 0.5, suggesting that half of the agents are unable to effectively address their assigned tasks.

`CAESAR` demonstrates remarkable robustness across all three scenarios. Notably, in Scenario 1 (Fig. 9a), where M_1 and M_2 are identical and thus all agents are peers, the inclusion of the sampling process within `CAESAR` does not impede learning gains. This is evidenced by its performance being on par with `Screen`, suggesting that the additional process does not detract from learning efficiency in homogeneous environments. In scenarios where M_1 and M_2 differ, particularly in the more complex Scenario 3, `CAESAR` continues to show strong performance, in stark contrast to the diminishing results of `Screen` and `All`. This resilience is primarily attributed to the sampling process integral to `CAESAR`, which effectively filters out non-peers, thereby ensuring that agents are exposed to relevant and beneficial strategies for their specific environments. It is important to note that while `Peers` excels in scenario 3, its implementation is not practical in real applications where the agent-MDP assignments are not known, as discussed in Sec. 4.3. `CAESAR` stands out in practical settings where the degree of heterogeneity among environments might be unknown or unpredictable. Its consistent performance across diverse scenarios underscores its suitability as a versatile and reliable aggregation strategy for FedRL in a practical setting.

6 CONCLUSION

In this study, we have tackled the intricate challenge of training distinct policies for agents across diverse environments within the realm of Federated Reinforcement Learning. Our investigation entailed a thorough analysis of six different aggregation strategies within the FedRL paradigm.

The experiments conducted in both customized `GridWorld` and `FrozenLake-v1` demonstrated the efficacy of Q-value convergence as a heuristic for peer detection in FedRL. Notably, the proposed `CAESAR` scheme stood out for its adaptability and resilience across a spectrum of environmental heterogeneity, consistently surpassing other evaluated baselines. This adaptability makes `CAESAR` particularly advantageous for real-world FedRL applications, where the unique characteristics of each environment are accommodated.

While our exploration focused on a tabular setting, future research directions include extending our methodologies to more complex and dynamic environments, especially those featuring a continuous control space. Furthermore, this work is primarily centered around agents that employ Q-value-based strategies. Acknowledging this as a limitation, another valuable direction for future research would be the incorporation of policy-based methods.

REFERENCES

[1] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. Openai gym. In *arXiv preprint arXiv:1606.01540*.

[2] Zhongxiang Dai, Flint Xiaofeng Fan, Cheston Tan, Trong Nghia Hoang, Bryan Kian Hsiang Low, and Patrick Jaillet. 2024. Chapter 14 - Federated sequential decision making: Bayesian optimization, reinforcement learning, and beyond. In *Federated Learning*. Academic Press, 257–279.

[3] Flint Xiaofeng Fan, Yining Ma, Zhongxiang Dai, Wei Jing, Cheston Tan, and Bryan Kian Hsiang Low. 2021. Fault-Tolerant Federated Reinforcement Learning with Theoretical Guarantee. In *35th Conference on Neural Information Processing Systems (NeurIPS)*.

[4] Flint Xiaofeng Fan, Yining Ma, Zhongxiang Dai, Cheston Tan, Bryan Kian Hsiang Low, and Roger Wattenhofer. 2023. *FedHQL: Federated Heterogeneous Q-Learning*. arXiv:2301.11135.

[5] Koki Fujita, Shugo Fujimura, Yuwei Sun, Hiroshi Esaki, and Hideya Ochiai. 2022. Federated Reinforcement Learning for the Building Facilities. In *2022 IEEE International Conference on Omni-layer Intelligent Systems (COINS)*. 1–6. https://doi.org/10.1109/COINS54846.2022.9854959

[6] Jin Hao, Yang Peng, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2022. Federated reinforcement learning with environment heterogeneity. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

[7] Watkins Christopher JCH and Peter Dayan. 1992. Q-learning. In *Machine learning 8*.

[8] Philip Jordan, Florian Grötschla, Flint Xiaofeng Fan, and Roger Wattenhofer. 2024. Decentralized Federated Policy Gradient with Byzantine Fault-Tolerance and Provably Fast Convergence. *arXiv preprint arXiv:2401.03489* (2024).

[9] Sajad Khodadadian, Pranay Sharma, Gauri Joshi, and Siva Theja Maguluri. 2022. Federated Reinforcement Learning: Linear Speedup Under Markovian Sampling. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*.

[10] Xinle Liang, Yang Liu, Tianjian Chen, Ming Liu, and Qiang Yang. 2023. Federated transfer reinforcement learning for autonomous driving. In *Federated and Transfer Learning*. Springer, 357–371.

[11] Boyi Liu, Lujia Wang, and Ming Liu. 2019. Lifelong Federated Reinforcement Learning: A Learning Architecture for Navigation in Cloud Robotic Systems. *IEEE Robotics and Automation Letters* 4, 4 (2019), 4555–4562. https://doi.org/10.1109/LRA.2019.2931179

[12] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

[13] Chetan Nadiger, Anil Kumar, and Sherine Abdelhak. 2019. Federated Reinforcement Learning for Fast Personalization. In *2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*. 123–127. https://doi.org/10.1109/AIKE.2019.00031

[14] Jiaju Qi, Qihao Zhou, Lei Lei, and Kan Zheng. 2021. Federated reinforcement learning: Techniques, applications, and open challenges. (2021).

[15] Han Shen, Kaiqing Zhang, Mingyi Hong, and Tianyi Chen. 2023. Towards Understanding Asynchronous Advantage Actor-critic: Convergence and Linear Speedup. *IEEE Transactions on Signal Processing* (2023).

[16] Richard Sutton and Andrew Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

[17] Xiaofei Wang, Chenyang Wang, Xiuhua Li, Victor CM Leung, and Tarik Taleb. 2020. Federated deep reinforcement learning for Internet of Things with decentralized cooperative edge caching. *IEEE Internet of Things Journal* 7, 10 (2020), 9441–9455.

[18] Jiin Woo, Gauri Joshi, and Yuejie Chi. 2023. The Blessing of Heterogeneity in Federated Q-Learning: Linear Speedup and Beyond. In *Proceedings of the 40th International Conference on Machine Learning*.

[19] Zeyue Xue, Pan Zhou, Zichuan Xu, Xiumin Wang, Yulai Xie, Xiaofeng Ding, and Shiping Wen. 2021. A resource-constrained and privacy-preserving edge-computing-enabled clinical decision system: A federated reinforcement learning approach. *IEEE Internet of Things Journal* 8, 11 (2021), 9122–9138.

[20] Qiang Yang, Yang Liu, Yong Cheng, Yan Kang, Tianjian Chen, and Han Yu. 2020. Federated Learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* (2020), 121–131.

[21] Shuai Yu, Xu Chen, Zhi Zhou, Xiaowen Gong, and Di Wu. 2020. When Deep Reinforcement Learning Meets Federated Learning: Intelligent Multitimescale Resource Management for Multiaccess Edge Computing in 5G Ultradense Network. *IEEE Internet of Things Journal* 8, 4 (2020), 2238–2251.

[22] Mingyue Zhang, Zhi Jin, Jian Hou, and Renwei Luo. 2022. Resilient Mechanism Against Byzantine Failure for Distributed Deep Reinforcement Learning. In *2022 IEEE 33rd International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 378–389.

[23] Hankz Hankui Zhuo, Wenfeng Feng, Yufeng Lin, Qian Xu, and Qiang Yang. 2019. *Federated deep reinforcement learning*. arXiv:1901.08277.