Federated Model-Based Offline Multi-Agent Reinforcement Learning for Wireless Networks

Dohyeok Kwon
Department of Electronic Engineering
Hanyang University
Seoul, South Korea
kwon1585@hanyang.ac.kr

Yeonseo Jeong
Department of Electronic Engineering
Hanyang University
Seoul, South Korea
wjddustj225@hanyang.ac.kr

Sungweon Hong
Department of Electronic Engineering
Hanyang University
Seoul, South Korea
hongsw94@hanyang.ac.kr

Songnam Hong
Department of Electronic Engineering
Hanyang University
Seoul, South Korea
snhong@hanyang.ac.kr

Abstract

Wireless networks are naturally modeled as multi-agent reinforcement learning (MARL) problems: distributed entities act on partial observations under interference and non-stationary traffic while pursuing common network objectives. Online exploration is risky and sample-inefficient in live systems, yet large operational logs are available, motivating an offline MARL approach. We introduce FedMORL, a federated model-based offline framework that shares an environment model rather than policy parameters. Each client learns dynamics and reward predictors from its logs and periodically aggregates them into a shared world model, which is then used locally to improve policies without environment interaction through (i) a planner-based policy learning scheme that improves training stability, and (ii) a rollout-based data augmentation mechanism that enhances local data coverage. The design preserves decentralized execution and limits privacy exposure by avoiding raw-data and policy sharing. Applied to the canonical Distributed Channel Access (DCA) task, FedMORL significantly reduces collisions and mean delay compared with rule-based baselines. We also empirically demonstrate that model federation is most beneficial, particularly under conditions of high data heterogeneity and limited local coverage. This supports model federation as a practical, privacy-preserving, offline-first path for multi-agent wireless control.

1 Introduction

Core tasks in next-generation wireless networks, such as Distributed Channel Access (DCA), user association, resource allocation, and spectrum/power/slicing allocation, can be naturally formulated as Multi-Agent Reinforcement Learning (MARL) problems. [35, 11, 38], where multiple agents act on distributed, partial observations to optimize a common objective. For instance, user equipments (UEs), access points (APs), and cells act based on limited information while seeking to enhance network-level performance metrics (e.g., throughput, latency, fairness). Such environments are inherently non-stationary due to time-varying traffic and interference, which further motivates the suitability of MARL formulations [37]. Despite this promise, most existing MARL studies focus on *online* learning, where policies are trained through direct interaction with the environment. While exploration is essential in online learning to discover effective policies, it inevitably degrades network performance

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: AI and ML for Next-Generation Wireless Communications and Networking (AI4NextG).

during training. It may even lead to critical risks, such as non-compliance with Ultra-Reliable Low-Latency Communication(URLLC) requirements, which limits its feasibility for deployment in real-world wireless systems [24].

To address these limitations, *offline* RL [13, 36, 16, 7, 17] has emerged as a promising alternative, in which policies are learned entirely from pre-collected large-scale datasets. This approach minimizes risks to real systems and alleviates sample inefficiency. Although the performance of offline RL critically depends on the quantity and quality of available data, wireless networks provide a distinct advantage: vast amounts of high-quality operational logs can be readily collected from deployed systems across diverse UEs and APs [38, 1]. Notably, since incumbent wireless systems are typically governed by well-engineered rule-based policies that already deliver attractive performance, the collected datasets inherently exhibit high quality, thereby offering a reliable foundation for offline training. Consequently, policies trained and validated offline can be deployed with minimal online exploration, attaining performance comparable to incumbent services while reducing deployment risk. In addition, recent offline-to-online (O2O) fine-tuning techniques [41, 42] enable safe, incremental adaptation during rollout and hold the potential to even surpass incumbent performance.

However, applying offline RL to multi-agent systems presents several challenges, as each agent typically has access to only a limited amount of offline data that records its own partial observations [29, 31, 39]. First, offline RL constrains policies to remain within the dataset's support through pessimistic predictions in unexplored regions. When the dataset coverage is narrow, this often results in overly conservative policies, severely reducing agents' ability to explore and potentially leading to training failure. Second, because each agent's dataset is generated under its own policy and contains only local observations, the resulting occupancy (or behavior) distributions can differ markedly. This discrepancy distorts the learning of transition dynamics and induces large errors in value estimation across agents [4, 14].

These issues suggest the adoption of a Centralized Training with Decentralized Execution (CTDE) approach, wherein data is aggregated at a central server for policy learning before distribution to individual agents [6, 8, 14, 39]. However, CTDE entails serious constraints, including communication overhead from transferring data and difficulty securing sufficient server storage capacity. In distributed wireless environments, data collected on each device may include sensitive personal information, and transferring it to a central server increases the risk of leakage and privacy violations [12].

These challenges with CTDE naturally suggest Federated Learning (FL). FL is a distributed learning paradigm where clients collaboratively train a global model by sharing model updates rather than raw data. In wireless networks, FL is often applied to address practical challenges like non-IID data or communication efficiency [27, 43, 34, 18]. However, the standard *policy* FL approach [15, 19, 33], which federates the policy networks, faces fundamental issues. First, it poses severe privacy risks, as policy gradients can be used to reconstruct agent behaviors [21]. Second, it struggles with partial observability. Averaging policy networks that are conditioned on different, local observations is technically ill-defined and often fails.

This motivates our central thesis: federating the environment models (e.g., dynamics, reward models) instead. A shared environment model is (1) inherently more private as it does not encode decision logic (i.e., policy network), and (2) it logically resolves the observability mismatch, as all agents are modeling the same shared world transition dynamics.

1.1 Contributions

We propose a general training pipeline, Federated Model-Based Offline MARL (FedMORL) that realizes this model-federation thesis. The FedMORL framework is flexible and can be applied across diverse wireless network tasks by simply replacing the environment model structure and algorithmic specifics, without modifying the overall pipeline. As such, it provides a concrete mechanism for agents to resolve the partial observability mismatch and enhance privacy by avoiding the sharing of policy networks, which mitigates the severe risks of behavior reconstruction.

As the core of this pipeline, we introduce two distinct methods for agents to leverage the aggregated global model for local policy learning. The first is planner-based policy learning (Algorithm 2), which uses the federated reward model to provide a stable and dense target for stabilizing local Q-learning. The second is Rollout-based Data Augmentation (Algorithm 3), which uses the federated dynamics model as a simulator to generate new data and enrich sparse local datasets. Finally, we

demonstrate the effectiveness of FedMORL on the challenging Distributed Channel Access (DCA) task. Our empirical results show that this approach yields significant reductions in key network metrics, including collisions and mean delay, compared to standard rule-based baselines.

2 Preliminaries

We consider a MARL setting formulated as a decentralized partially observable Markov decision process (Dec-POMDP) [25]. A Dec-POMDP is defined by the tuple

$$\mathcal{G} = \langle N, S, \{A_i\}_{i=1}^N, P, \{O_i\}_{i=1}^N, R, \gamma \rangle,$$

where N is the set of agents, S the global state space, A_i the action space of agent i, O_i the local observation space of agent i, $P(s'|s,a)$ the transition dynamics, $R(s,a)$ the reward function, and $\gamma \in (0,1)$ the discount factor.

At each time step t, the environment is in state $s_t \in S$. Each agent $i \in N$ selects an action $a_i^t \in A_i$ based on its local observation $o_i^t \in O_i$. The joint action $\mathbf{a}_t = (a_1^t, \ldots, a_N^t)$ induces the next-state distribution according to $P(s_{t+1}|s_t, \mathbf{a}_t)$, and all agents receive a reward $R(s_t, \mathbf{a}_t)$. The MARL objective is to maximize the expected discounted return $J(\pi) = \mathbb{E}_{\pi,P}[\sum_{t=0}^{\infty} \gamma^t R(s_t, \mathbf{a}_t)]$, where $\pi = \{\pi_i(a_i|o_i)\}_{i=1}^N$ denotes the set of decentralized policies.

In wireless networks, the Dec-POMDP abstraction naturally captures the decentralized nature of UEs and BSs under partial observability and stochastic interference [26, 38, 8]. Traditional MARL often adopts CTDE, where agents exploit global state information during training but act on local observations at execution.

However, to better reflect practical wireless constraints, we instead adopt *decentralized training and execution* (DTE) [2], where each agent optimizes its local policy $\pi_i(a^i|o^i)$ using only local observations and rewards, i.e., $J_i(\pi_i) = \mathbb{E}_{\pi_i,P}[\sum_{t=0}^{\infty} \gamma^t r_i(o_t^i, a_t^i)]$. DTDE is also naturally compatible with federated learning, since only model updates—not raw data—need to be exchanged. Recent studies have shown that Dec-POMDP [38, 8] and DTDE [26] formulations achieve strong performance in tasks such as user association and channel access, motivating our federated, model-based offline MARL framework.

3 Federated Model-based Offline MARL

3.1 Algorithm

We proposed a federated model-based offline MARL pipeline (Algorithm 1). Each client k trains a local environment model $M_k^{(r)}$ from its offline log dataset $\mathcal{D}_k$ and uploads only model parameters (and optional lightweight statistics) to a server. The server applies an aggregation function (e.g., weighted averaging) to obtain an updated global model $M_{\text{global}}^{(r)}$. This repeated until convergence. After the federation ends, every client independently learns its policy π_k offline, using $M_{\text{global}}^{\star}$ as a common model. Raw data and policy network parameters are never shared.

Algorithm 1 FEDERATED MODEL-BASED OFFLINE MARL (FEDMORL)

Require: Offline Dataset $\{\mathcal{D}_k\}_{k=1}^K$
Initialize Environment Model $\{M_k\}_{k=1}^K$
for $r = 1, 2, \ldots$ **until** convergence **do** ▷ **Model Training Step**
  for each client $k \in [K]$ **in parallel do**
    $M_k^{(r)} \leftarrow \text{MODELTRAINING}(\mathcal{D}_k)$
    send parameters of $M_k^{(r)}$ and (optional) statistics $\sigma_k^{(r)}$ to server.
  end for
  $M_{\text{global}}^{(r)} \leftarrow \text{AGGREGATE}(\{M_k^{(r)}\}_{k=1}^K, \{\sigma_k^{(r)}\}_{k=1}^K)$
end for
for each client $k \in [K]$ **in parallel do** ▷ **Policy Learning Step**
  $\pi_k \leftarrow \text{POLICYLEARNING}(\mathcal{D}_k, M_{\text{global}}^{\star})$
end for

3.2 Model Training Step

A standard environment model comprises a *transition dynamics model* $p_\psi(s' \mid s, a)$ and/or a *reward model* $r_\phi(s, a)$. These components can be implemented as neural networks and trained from offline datasets $\{\mathcal{D}_k\}$ to predict the environment's transition and reward structure. In model-based reinforcement learning, they are typically used (i) to generate additional data via short-horizon rollouts from the offline buffer, or (ii) as core components of planning algorithms that provide value estimates the learned policy alone cannot reliably capture [23].

However, in multi-agent systems, when each agent learns its environment model independently, limited local observations make it difficult to accurately capture complex system-wide interactions and global state evolution [6, 2]. Our core hypothesis is that an individual agent's dataset $\mathcal{D}_k$ only covers a specific fraction of the system-wide dynamics (e.g., in DCA, one agent's logs might primarily capture low-contention scenarios, while another's captures high-contention scenarios). A model M_k trained only on $\mathcal{D}_k$ will thus be biased. By federating models trained on this heterogeneous data [22, 20], the resulting global model $M^\star_{\text{global}}$ can form a more accurate and comprehensive representation of the entire state-action space than any single local model. Aggregation of these local models through federated learning provides a comprehensive understanding of the global environment dynamics, addressing the challenges of partial observability and complex interactions inherent in decentralized multi-agent systems [5]. Consequently, in MARL, a *global* transition dynmaics model can capture complex, non-stationary interference patterns. Similarly, a *global* reward model can learn to predict system-wide outcomes like collisions, a joint-reward signal, even if an agent's local observations o_k are partial or its dataset $\mathcal{D}_k$ rarely contained collision events.

3.3 Policy Learning Step

Planner-based Policy Learning. We train each local policy π_{θ_k} using a Q-weighted imitation objective, guided by the federated global model $M^\star_{\text{global}}(p_{\phi^\star}, r_{\psi^\star})$, which serves as a planner or critic. This approach combines model-based value estimation, as in PlaNet [10] and Dreamer [9], with this imitation learning principle, which is similar to Critic Regularized Regression (CRR) [32]. Conceptually, it is similar in spirit to MuZero [28], where a learned world model enables consistent policy improvement through planning.

For each transition $(o, a, r, o') \sim \mathcal{D}_k$, the planner evaluates the quality of the *real* logged action a via:

$$\hat{Q}_{\text{plan}}(o, a) = r_{\psi^\star}(o, a) + \gamma\, \mathbb{E}_{o' \sim p_{\phi^\star}(\cdot|o,a)}[\max_{a'} \hat{Q}_{\text{plan}}(o', a')]. \tag{1}$$

In practice, this value is approximated by executing a short-horizon planning algorithm (e.g., Monte Carlo Tree Search) using the learned world model $M^\star_{\text{global}}$. The local policy is updated by minimizing the Q-weighted loss:

$$\mathcal{L}_{\text{plan}} = \mathbb{E}_{(o,a)\sim\mathcal{D}_k}\big[-\log \pi_{\theta_k}(a|o)\, \exp(\hat{Q}_{\text{plan}}(o, a)/\beta)\big], \tag{2}$$

where β controls the weighting sharpness. This encourages imitation of high-value actions identified by the federated planner while suppressing inferior behaviors.

Algorithm 2 FEDMORL(PLAN): Planner-based Policy Learning.

Require: Global model $M^\star_{\text{global}}(p_{\phi^\star}, r_{\psi^\star})$, local dataset $\mathcal{D}_k$, temperature β
Initialize local policy π_{θ_k}
for each training iteration **do**
  Sample mini-batch $\{(o_i, a_i, r_i, o'_i)\} \sim \mathcal{D}_k$
  for each (o, a, r, o') in batch **do**
    Compute $\hat{Q}_{\text{plan}}(o, a)$ via Eq. (1)
    Compute $\mathcal{L}_{\text{plan}}$ via Eq. (2)
    Update $\theta_k \leftarrow \theta_k - \eta \nabla_{\theta_k} \mathcal{L}_{\text{plan}}$
  end for
end for
return Local policy π_{θ_k}

Rollout-based Data Augmentation. While the planner-based policy learning exploits the global model for value estimation, the same model can also be used to *expand* the local experience distribution. Specifically, the federated global model $M^{\star}_{\text{global}} = (p_{\phi^\star}, r_{\psi^\star})$ acts as a simulator that performs short-horizon rollouts to generate additional transitions. This rollout strategy has been widely used in model-based reinforcement learning, from early work such as Dyna-Q [30] to more recent methods like MOPO [40]. From a MARL perspective, $M^{\star}_{\text{global}}$ can produce plausible transitions (e.g., collision or cooperation scenarios) that are underrepresented in $\mathcal{D}_k$, improving data coverage and diversity.

For each real transition $(o, a, r, o') \sim \mathcal{D}_k$, we initiate a model rollout from its terminal state o'. Over H simulated steps, actions are drawn from a local policy $\pi^{(i)}_{\theta_k}$, and next states and rewards are generated by the global model:

$$o_{t+1} \sim p_{\phi^\star}(\cdot|o_t, a_t), \qquad r_t = r_{\psi^\star}(o_t, a_t).$$

Finally, the merged dataset $\mathcal{D}^{\text{tot},(i)}_k = \mathcal{D}_k \cup \mathcal{D}^{\text{aug},(i)}_k$ is used to train a model-free algorithm to update the policy $\pi^{(i+1)}_{\theta_k}$, and this rollout–training process is repeated until convergence.

Algorithm 3 FEDMORL(ROLLOUT): Rollout-based Data Augmentation.

Require: Global model $M^{\star}_{\text{global}}(p_{\phi^\star}, r_{\psi^\star})$, local dataset $\mathcal{D}_k$, rollout horizon H

Initialize $\mathcal{D}^{\text{aug},(0)}_k \leftarrow \mathcal{D}_k$
for iteration $i = 1, 2, \ldots$ **until** convergence **do**
  for each transition $(o, a, r, o') \in \mathcal{D}_k$ **do**
    $o_0 \leftarrow o'$ ▷ Start rollout from the real next state
    for $t = 0$ **to** $H - 1$ **do**
      $a_t \sim \pi^{(i)}_{\theta_k}(o_t)$
      $o_{t+1} \sim p_{\phi^\star}(\cdot|o_t, a_t)$; $r_t \leftarrow r_{\psi^\star}(o_t, a_t)$
      Add (o_t, a_t, r_t, o_{t+1}) to $\mathcal{D}^{\text{aug,(i)}}_k$
    end for
  end for
  $\mathcal{D}^{\text{tot},(i)}_k \leftarrow \mathcal{D}^{\text{aug},(i-1)}_k \cup \mathcal{D}^{\text{aug},(i)}_k$
  Train policy $\pi^{(i)}_{\theta_k}$ on $\mathcal{D}^{\text{tot}}_k$
end for
return Final policy $\pi^{\star}_{\theta_k}$

4 Offline MARL for Distributed Channel Access

4.1 System Model

We evaluate our approach on the Distributed Channel Access (DCA) task, where N user equipments (UEs) contend for uplink access to a shared wireless channel. The simulator follows an IEEE 802.11-like slotted structure with a slot duration of 9 μs. Each UE maintains a finite FIFO buffer of size 10, and packet arrivals follow a Poisson process. A fixed-size packet transmission occupies 120 slots, including the payload and acknowledgement sequence.

Channel access is governed by the standard Distributed Coordination Function (DCF). Each UE maintains a contention window (CW) and a backoff counter that freezes when the medium is busy and decrements when idle. After a successful transmission, the CW resets to $\text{CW}_{\min}$, while collisions trigger the binary exponential backoff rule, doubling the CW up to $\text{CW}_{\max}$. These dynamics produce realistic idle/success/collision patterns consistent with IEEE 802.11.

4.2 Decentralized Observation, Action, and Reward

Each UE receives a local observation that reflects only its own state and the current channel condition. Following our implementation, the observation for agent i at slot t is

$$o_i(t) = \big[c(t),\ a_i(t-1),\ \tilde{d}_i(t),\ \tilde{b}_i(t)\big],$$

where $c(t)$ is the discrete channel state, $a_i(t-1)$ is the previous action, and $\tilde{d}_i(t)$ is the normalized delay-to-last-transmission (D2LT). Finally, $\tilde{b}_i(t) = b_i(t)/B$ is the normalized buffer ratio, where $b_i(t)$ is the current queue length and B is the maximum buffer capacity, indicating local queue urgency. Each agent selects a binary action $a_i(t) \in \{0, 1\}$, corresponding to attempting or deferring transmission. The simulator masks physically infeasible actions (e.g., transmitting during BUSY periods), but agents conceptually output an action at every slot.

We adopt a shaped per-UE reward that decomposes into a global throughput component and a local delay penalty:

$$r_i(t) = r^{\text{glob}}(t) + \alpha\, r_i^{\text{local}}(t), \tag{3}$$

where $\alpha > 0$ controls the strength of delay regularization. The global term $r^{\text{glob}}(t)$ provides a shared reward based on the slot's outcome: $+1$ for a successful transmission, -1 for a collision, and 0 otherwise (idle or ACK slots). This component encourages the network to allocate the channel to successful transmissions while penalizing collisions. The delay component for agent i is given by $r_i^{\text{local}}(t) = -\tilde{d}_i(t)\tilde{b}_i(t)$. This local penalty discourages policies that keep packets waiting in the buffer for a long time, thereby improving delay and fairness across UEs. This approach of balancing network efficiency with per-agent fairness via a local reward component is also utilized in MARL-based wireless access [8]. The formal Dec-POMDP specification is provided in Appendix A.2.

5 Experiments

5.1 Experimental Setup

We evaluate the performance of FedMORL on the DCA task described in Section 4.1. Our objective is to assess whether the policy learned from offline, sub-optimal data can discover a superior strategy compared to standard stochastic baselines.

Offline Dataset Generation. Offline datasets are generated using the CSMA/CA implementation. Each UE performs carrier sensing, freezes/decrements its backoff counter based on channel state, and attempts transmission when the counter reaches zero. The CW is reset upon success and doubled after collisions, following IEEE 802.11 DCF with $\mathrm{CW}_{\min} = 7$ and $\mathrm{CW}_{\max} = 1023$. This rule-based behavior, which is highly aggressive, produces diverse offline trajectories covering idle, collision, and heavy-load regimes.

Baselines. We compare FedMORL against four key baselines: (i) **EDCA**(AC_VO, AC_VI, AC_BE): Three standardized stochastic random-access categories from IEEE 802.11. (ii) **CSMA/ECA:** A deterministic collision-avoidance protocol designed to converge to collision-free access. Details of these algorithms are in Appendix A.1.

For our method, we evaluate two variants: **FedMORL(plan)**, which utilizes the planner-based policy learning (Algorithm 2), and **FedMORL(roll)**, which uses rollout-based data augmentation (Algorithm 3). Both are trained using the global model M^*_{global}.

Scenario and Metrics. We conduct the evaluation under a saturated Poisson traffic condition, where all UEs always have packets to send. This setup creates a high-contention scenario to stress-test the coordination capabilities of each algorithm. We vary the number of UEs (N) from 2 to 14 and measure three key network metrics: (i) Throughput Ratio, (ii) Mean Packet Delay, and (iii) Collision Rate.

5.2 Performance Evaluation

Figure 1 presents the performance comparison. The results demonstrate that FedMORL, despite being trained on aggressive and sub-optimal data, successfully learns a more conservative and effective policy, matching the best-performing stochastic baseline (AC_BE) and significantly outperforming its data-source policy.

FedMORL vs. Stochastic Baselines (EDCA). As shown in Figure 1(a) and (c), the stochastic baselines reveal a clear trade-off in high-contention. Aggressive policies (AC_VO, AC_VI) with

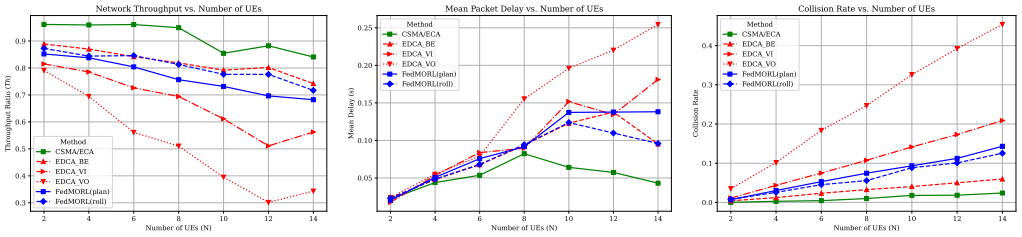

Figure 1: Performance comparison under saturated traffic. (a) Throughput Ratio, (b) Mean Delay (log scale), and (c) Collision Rate vs. Number of UEs. FedMORL(plan, roll) learns to match the best-performing stochastic baseline (AC_BE) and outperforms the more aggressive AC_VI and AC_VO.

small CW_{min} (7, 15) suffer from severe performance degradation as N increases, due to a sharp rise in collisions. Conversely, the most conservative policy, AC_BE (CW_{min}=31), avoids collisions more effectively and thus achieves the highest throughput and lowest delay among the EDCA baselines.

This highlights our key finding: our FedMORL methods, which were trained only on data from the aggressive CSMA/CA policy ($CW_{min} = 7$), converge to the performance of the superior AC_BE baseline. This demonstrates that FedMORL did not simply imitate its sub-optimal, aggressive data source. Instead, it learned from the federated model that an aggressive policy leads to high collisions (negative rewards) and successfully discovered a more conservative, superior policy that matches the best-performing conventional heuristic.

Data Composition Analysis. The results in Figure 1 validate FedMORL's ability to extract a superior policy from sub-optimal, stochastic data. Building on this observation, we are further examining two key aspects of dataset composition: data heterogeneity and data quality.

First, we are evaluating the framework's robustness under heterogeneous client datasets, where each device may log trajectories generated under different traffic intensities or contention levels. This setting represents a realistic and challenging federated environment in which local data distributions are inherently non-IID. Second, to study the influence of data quality, we are analyzing mixed offline datasets that combine the original CSMA/CA trajectories with high-quality, near-optimal samples produced by the deterministic CSMA/ECA protocol. These studies aim to characterize how FedMORL's learning performance scales when policies are informed by varying proportions of low-quality and high-quality experience.

6 Conclusion

In this paper, we introduced FedMORL, a federated model-based offline MARL framework tailored for decentralized wireless network control. By federating environment models rather than policy networks, our approach enables agents to share global context while avoiding the privacy and stability issues commonly associated with policy federation. Empirical results on the DCA task show that FedMORL can learn coordinated channel-access strategies from sub-optimal offline data, outperforming the original rule-based baselines.

Looking ahead, we aim to incorporate advanced offline RL techniques to better mitigate distributional shift and further enhance robustness under highly heterogeneous (non-IID) client datasets. In addition, because FedMORL relies on the fidelity of the shared world model, addressing model uncertainty during planning and policy learning remains an important direction. These extensions will help shape a fully adaptive, offline-first control framework for next-generation wireless networks.

Acknowledgements

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. RS-2024-00409492, RS-2024-00334904).

References

[1] Pegah Alizadeh, Anastasios Giovanidis, Pradeepa Ramachandra, Vasileios Koutsoukis, and Osama Arouk. Offline reinforcement learning for mobility robustness optimization. *arXiv preprint arXiv:2506.22793*, 2025.

[2] Christopher Amato. An initial introduction to cooperative multi-agent reinforcement learning. *arXiv preprint arXiv:2405.06161*, 2024.

[3] Jaume Barcelo, Boris Bellalta, Anna Sfairopoulou, Cristina Cano, and Miquel Oliver. Csma with enhanced collision avoidance: A performance assessment. In *VTC Spring 2009-IEEE 69th Vehicular Technology Conference*, pages 1–5. IEEE, 2009.

[4] Paul Barde, Jakob Foerster, Derek Nowrouzezahrai, and Amy Zhang. A model-based solution to the offline multi-agent reinforcement learning coordination problem. *arXiv preprint arXiv:2305.17198*, 2023.

[5] Ignasi Clavera, Jonas Rothfuss, John Schulman, Yasuhiro Fujita, Tamim Asfour, and Pieter Abbeel. Model-based reinforcement learning via meta-policy optimization. In *Conference on robot learning*, pages 617–629. PMLR, 2018.

[6] Christian Schroeder De Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip HS Torr, Mingfei Sun, and Shimon Whiteson. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533*, 2020.

[7] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pages 2052–2062. PMLR, 2019.

[8] Ziyang Guo, Zhenyu Chen, Peng Liu, Jianjun Luo, Xun Yang, and Xinghua Sun. Multi-agent reinforcement learning-based distributed channel access for next generation wireless networks. *IEEE Journal on Selected Areas in Communications*, 40(5):1587–1599, 2022.

[9] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

[10] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR, 2019.

[11] Fenghe Hu, Yansha Deng, and A Hamid Aghvami. Scalable multi-agent reinforcement learning algorithm for wireless networks. *arXiv preprint arXiv:2108.00506*, 2021.

[12] Shuyan Hu, Xiaojing Chen, Wei Ni, Ekram Hossain, and Xin Wang. Distributed machine learning for wireless communication networks: Techniques, architectures, and applications. *IEEE Communications Surveys & Tutorials*, 23(3):1458–1493, 2021.

[13] Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. *arXiv preprint arXiv:1907.00456*, 2019.

[14] Jiechuan Jiang and Zongqing Lu. Offline decentralized multi-agent reinforcement learning. *arXiv preprint arXiv:2108.01832*, 2021.

[15] Hao Jin, Yang Peng, Wenhao Yang, Shusen Wang, and Zhihua Zhang. Federated reinforcement learning with environment heterogeneity. In *International Conference on Artificial Intelligence and Statistics*, pages 18–37. PMLR, 2022.

[16] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in neural information processing systems*, 32, 2019.

[17] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33: 1179–1191, 2020.

[18] Dohyeok Kwon, Jonghwan Park, and Songnam Hong. Tighter regret analysis and optimization of online federated learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(12):15772–15789, 2023.

[19] Guangchen Lan, Dong-Jun Han, Abolfazl Hashemi, Vaneet Aggarwal, and Christopher G Brinton. Asynchronous federated reinforcement learning with policy gradient updates: Algorithm design and convergence analysis. *arXiv preprint arXiv:2404.08003*, 2024.

[20] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

[21] Guanlin Liu and Lifeng Lai. Efficient adversarial attacks on online multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 36:24401–24433, 2023.

[22] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[23] Thomas M Moerland, Joost Broekens, Aske Plaat, Catholijn M Jonker, et al. Model-based reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 16(1):1–118, 2023.

[24] Ravi Teja Mullapudi, Steven Chen, Keyi Zhang, Deva Ramanan, and Kayvon Fatahalian. Online model distillation for efficient video inference. In *Proceedings of the IEEE/CVF International conference on computer vision*, pages 3573–3582, 2019.

[25] Frans A Oliehoek, Christopher Amato, et al. *A concise introduction to decentralized POMDPs*, volume 1. Springer, 2016.

[26] Mohamed Sana, Antonio De Domenico, Wei Yu, Yves Lostanlen, and Emilio Calvanese Strinati. Multi-agent reinforcement learning for adaptive user association in dynamic mmwave networks. *IEEE Transactions on Wireless Communications*, 19(10):6520–6534, 2020.

[27] Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. Robust and communication-efficient federated learning from non-iid data. *IEEE transactions on neural networks and learning systems*, 31(9):3400–3413, 2019.

[28] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[29] Jianzhun Shao, Yun Qu, Chen Chen, Hongchang Zhang, and Xiangyang Ji. Counterfactual conservative q learning for offline multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 36:77290–77312, 2023.

[30] Richard S Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine learning proceedings 1990*, pages 216–224. Elsevier, 1990.

[31] Xiangsen Wang, Haoran Xu, Yinan Zheng, and Xianyuan Zhan. Offline multi-agent reinforcement learning with implicit global-to-local value regularization. *Advances in Neural Information Processing Systems*, 36:52413–52429, 2023.

[32] Ziyu Wang, Alexander Novikov, Konrad Zolna, Josh S Merel, Jost Tobias Springenberg, Scott E Reed, Bobak Shahriari, Noah Siegel, Caglar Gulcehre, Nicolas Heess, et al. Critic regularized regression. *Advances in Neural Information Processing Systems*, 33:7768–7778, 2020.

[33] Jiin Woo, Gauri Joshi, and Yuejie Chi. The blessing of heterogeneity in federated q-learning: Linear speedup and beyond. *Journal of Machine Learning Research*, 26(26):1–85, 2025.

[34] Chuhan Wu, Fangzhao Wu, Lingjuan Lyu, Yongfeng Huang, and Xing Xie. Communication-efficient federated learning via knowledge distillation. *Nature communications*, 13(1):2032, 2022.

[35] Junjie Wu and Xuming Fang. Collaborative optimization of wireless communication and computing resource allocation based on multi-agent federated weighting deep reinforcement learning. *arXiv preprint arXiv:2404.01638*, 2024.

[36] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[37] Shahaf Yamin and Haim H Permuter. Multi-agent reinforcement learning for network routing in integrated access backhaul networks. *Ad Hoc Networks*, 153:103347, 2024.

[38] Kun Yang, Chengshuai Shi, Cong Shen, Jing Yang, Shu-Ping Yeh, and Jaroslaw J Sydir. Offline reinforcement learning for wireless network optimization with mixture datasets. *IEEE Transactions on Wireless Communications*, 23(10):12703–12716, 2024.

[39] Yiqin Yang, Xiaoteng Ma, Chenghao Li, Zewu Zheng, Qiyuan Zhang, Gao Huang, Jun Yang, and Qianchuan Zhao. Believe what you see: Implicit constraint approach for offline multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34:10299–10312, 2021.

[40] Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.

[41] Haichao Zhang, We Xu, and Haonan Yu. Policy expansion for bridging offline-to-online reinforcement learning. *arXiv preprint arXiv:2302.00935*, 2023.

[42] Han Zheng, Xufang Luo, Pengfei Wei, Xuan Song, Dongsheng Li, and Jing Jiang. Adaptive policy learning for offline-to-online reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 11372–11380, 2023.

[43] Hangyu Zhu, Jinjin Xu, Shiqing Liu, and Yaochu Jin. Federated learning on non-iid data: A survey. *Neurocomputing*, 465:371–390, 2021.

A DCA Environment and Implementation Details

A.1 DCA System details

For benchmarking against standardized Wi-Fi QoS mechanisms, we use the Enhanced Distributed Channel Access (EDCA) parameters defined in IEEE 802.11. We evaluate three access categories (ACs), namely AC_VO, AC_VI, and AC_BE, each with distinct contention window settings as summarized in Table 1, providing practical QoS-oriented baselines for comparison.

Table 1: EDCA Access Category Parameters

Parameter	**AC_VO**	**AC_VI**	**AC_BE**
SIFS (μs)	18		
DIFS (μs)	36		
$\mathrm{CW}_{\min}$	7	15	31
$\mathrm{CW}_{\max}$	15	31	1023

We additionally implement the Enhanced Collision Avoidance (CSMA/ECA) mechanism [3] to compare against a deterministic collision-avoidance scheme. Stations initially behave according to standard random backoff, but after the first successful transmission, they switch to deterministic operation by fixing the backoff counter to

$$V = \frac{CW_{\min}}{2}.$$

enabling convergence toward collision-free periodic access. Upon a collision, a temporary random backoff is applied, after which the station returns to the deterministic value V following the next successful transmission.

A.2 Dec-POMDP Specification

For completeness, we summarize the underlying Dec-POMDP formulation. Let $\mathcal{N} = \{1, \ldots, N\}$ denote the set of UEs.

State and dynamics. The global state at slot t is $s_t \in \mathcal{S}$, which comprises:

- the channel state $c_t \in \{\texttt{IDLE}, \texttt{SUCCESS}, \texttt{COLLISION}, \texttt{ACK}\}$,
- per-UE queue lengths q_t^i and backoff counters b_t^i,
- per-UE delay statistics d_t^i (D2LT),
- CW values and other bookkeeping variables required by the DCF mechanism.

The transition kernel $P(s_{t+1} \mid s_t, a_t)$ is implicitly defined by the discrete-event simulator: given (s_t, a_t), the simulator advances the MAC state, updates backoff counters and buffers, resolves collisions, and generates the next state s_{t+1}.

Actions. Each agent (UE) i chooses a binary action

$$a_t^i \in \mathcal{A}_i = \{0, 1\},$$

where $a_t^i = 1$ denotes a transmit attempt and $a_t^i = 0$ denotes deferring. Let $a_t = (a_t^1, \ldots, a_t^N)$ denote the joint action. In the implementation, physically infeasible attempts (e.g., during `BUSY` periods) are masked by the underlying MAC logic, but from the Dec-POMDP perspective agents conceptually select a_t^i at every slot.

Observations. The local observation o_t^i, as defined in the main text, includes normalized D2LT statistics derived from the global state. Let d_t^i denote the (unnormalized) D2LT and define the normalized components as

$$\tilde{d}_t^i = \frac{d_t^i}{\sum_{j \in \mathcal{N}} d_t^j + \varepsilon},$$

for a small $\varepsilon > 0$. This observation design is consistent with the decentralized setting: agents do not observe each other's buffers or CW values, but only local MAC-level feedback.

Rewards. The per-agent reward $r_i(t)$, formally $r_i(s_t, a_t)$, follows the structure from Eq. (3):

$$r_i(s_t, a_t) = r^{\text{glob}}(s_t, a_t) + \alpha\, r_i^{\text{delay}}(s_t).$$

The global throughput component is formally defined as:

$$r^{\text{glob}}(s_t, a_t) = \begin{cases} +1, & \text{if slot } t \text{ ends with a successful transmission,} \\ -1, & \text{if slot } t \text{ ends with a collision,} \\ 0, & \text{otherwise (idle or ACK slots).} \end{cases}$$

The local component $r_i^{\text{delay}}(s_t)$ is a function $g(d_t^i)$ based on the agent's internal state (D2LT and buffer size), as conceptually described in the main text. This structure separates a global throughput-oriented term and a local delay penalty, and is used consistently across all experiments.